Workshop Proceedings

5th International Workshop on Knowledge Discovery in Healthcare Data (KDH)

edited by

Kerstin Bach, Razvan Bunescu, Cindy Marling,

& Nirmalie Wiratunga

Co-located with Digital ECAI 2020

August 29 - 30th, 2020

Preface
The 5th International Workshop on Knowledge Discovery in Healthcare Data (KDH)

Introduction

The Knowledge Discovery in Healthcare Data (KDH) workshop series was established in 2016 to bring together AI and clinical researchers, fostering collaborative discussions and presenting AI research efforts to solve pressing problems in health care. This fifth edition of the workshop was held in conjunction with the 24th European Conference on Artificial Intelligence, Digital ECAI 2020, which was hosted in Santiago de Compostela, Spain, but conducted virtually. The focus of the workshop was on learning health care systems. For the second time, this workshop featured a challenge: The Blood Glucose Level Prediction (BGLP) Challenge.

The notion of the learning health care system has been put forward to denote the translation of routinely collected data into knowledge that drives the continual improvement of medical care. This notion has been described in many forms, but each follows a similar cycle of assembling, analyzing and interpreting data from multiple sources (clinical records, guidelines, patient-provided data including wearables, omic data, etc.), followed by feeding the acquired knowledge back into clinical practice. This framework aims to provide personalized recommendations and decision support tools to aid both patients and care providers, to improve outcomes and personalize care.

This framework also extends the range of actions possible in response to patient monitoring data, for example, alerting patients or automatically adjusting insulin doses when blood glucose levels are predicted to go out of range. Blood glucose level prediction is a challenging task for AI researchers with the potential to improve the health and well-being of people with diabetes. In the Blood Glucose Level Prediction (BGLP) Challenge, researchers came together to compare the efficacy of different machine learning (ML) prediction approaches on a standard set of real patient data.

The workshop received 35 submissions, each of which was peer-reviewed by three reviewers. Based on the reviews, 10 technical papers and 16 BGLP Challenge papers were accepted for presentation at the workshop. Among the accepted papers, the current trend of applying deep learning (DL) is strongly represented, while other methods used are case-based reasoning (CBR), natural language processing, and time series analysis. Another evident trend was the need for open data sets that can drive the field forward and promote building on each other's work. This topic was addressed by the invited talk as well as by the included BGLP Challenge.

Keynote Speaker: Kerstin Bach, NTNU, Norway

Bio: Kerstin Bach is an Associate Professor of Computer Science and Artificial Intelligence in the Department of Computer Science at the Norwegian University of Science and Technology (NTNU). She has been at NTNU since 2017, where she is currently deputy head of the Data and Artificial Intelligence group and a core member of the Norwegian Open AI Lab. Bach received her doctorate summa cum laude from the Department of Mathematics, Natural Sciences, Economics and Computer Science of the Hildesheim University, Germany, in 2012.

Kerstin Bach has broad experience building industrial strength AI applications as well as leading and collaborating on interdisciplinary teams. While working at Verdande Technology, she worked on a platform delivering AI services for the Oil and Gas, Finance and Healthcare sector. Further, she has headed the myCBR open source project since 2010 and has conducted research projects leveraging CBR and other AI methods for over 13 years. She is currently focused on two Horizon 2020 projects, selfBACK and AI4EU. She is the project manager of the selfBACK project, responsible for the technical integration of selfBACK into Back-UP, where she leads the Machine Learning tasks. In the AI4IoT pilot of AI4EU, she co-leads the efforts to develop AI showcases for the platform featuring Air Quality measurements. Bach is active in communicating AI research internationally. She is the chair of the German Special Interest Group on Knowledge Management and a board member of the Norwegian AI Society.

Title: The Potential for AI in Public Health: Lessons Learned from Developing and Testing a Patient-Centered Mobile App

Abstract: This talk provides an overview of how Artificial Intelligence and Machine Learning have been used to develop a mobile app that facilitates self-management of low back pain patients. It covers the development of the decision support system for patients using case-based reasoning as well as system evaluation via a randomized controlled trial testing the effectiveness of the app. This talk focuses on the development of the selfBACK system [24], but the approaches and methodologies employed can also be applied to the development of systems for other chronic diseases benefiting from self-management.

Accepted Papers

Main Track Papers

Main track technical papers present original research work across a broad range of KDH topics and domains. Given the current Covid-19 pandemic, this proceedings features three papers addressing the use of AI for detecting anomalies in X-ray scans. Paper [16] presents an approach for quantifying the uncertainty of deep neural networks (DNN) for the task of chest X-ray image classification, with results showing that utilizing uncertainty information may improve DNN performance for some metrics and observations. Paper [10] presents a study and a concrete tool based on machine learning to predict the prognosis of hospitalized patients with Covid-19. Paper [12] proposes a two-stage segmentation method which is capable of improving the accuracy of detection and segmentation of lung nodules from 2D CT images, achieving promising results that put the method among the top lung nodule segmentation methods.

The second group of papers focuses on how AI-based explanation and visualization can help patients and clinicians use the vast amount of information available to improve diagnosis, knowledge discovery and care. Paper [25] presents InterVENE, an approach that visualizes neural embeddings and interactively explains this visualization, aiming for knowledge extraction and network interpretation. Paper [7] makes use of the graphical representation capabilities of Formal Concept Analysis (FCA) and use graph databases as a visualization method for knowledge patterns. The authors exemplify their approach on a particular medical dataset, highlighting a 3D representation of conceptual hierarchies by using virtual reality. Paper [4] is a position paper, in which the authors analyze the cause-effect relationships for determining the causal status among a set of events. They argue that causal knowledge graphs can improve the accuracy and reliability of existing ML/DL-based diagnosis methods, by producing transparent justifications and explanations of the output. Paper [23] presents initial findings towards assessing how computer vision, natural language processing and other systems could be correctly embedded in the clinicians' pathway to better aid in fracture detection.

A third group of papers addresses the use of machine learning for blood glucose level prediction (BGLP) and diabetes management. Paper [22] compares the effectiveness of several BGLP models and found that Lasso regression performed best out of the algorithms used for both the 30-minute and 60-minute prediction horizons. Paper [1] presents a generic neural architecture previously used for BGLP in a what-if scenario that can be adapted and leveraged to make either carbohydrate or bolus recommendations. Paper [17] addresses the problem of missing sensor readings in glucose monitoring data of artificial pancreas (AP) systems. It uses data from virtual patients and a state-of-the-art AP controller simulating various scenarios.

BGLP Challenge Papers

The BGLP Challenge papers describe blood glucose (BG) level prediction approaches and experimental evaluations on the newly updated OhioT1DM dataset [20]. Of the 16 systems with papers that were accepted for publication, 8 systems had results that conformed to The BGLP Challenge Rules[1]. These 8 systems were all evaluated using the exact same test points for each of 6 data contributors in the OhioT1DM dataset. Results were reported as the root mean squared error (RMSE) and the mean absolute error (MAE) scores for the 30 minute and 60 minute prediction horizons. The 4 scores were added together to compute an overall score, and the 8 systems were ranked in increasing order of this total score. Table 1 shows the official ranking of the 8 systems, based on this overall score. Additional rankings, e.g. based on each of the 4 measures separately, as well as links to the source code for all 16 systems, are available on The BGLP Results [2] page.

Gated versions (LSTMs [13], GRUs [6]) of recurrent neural networks (RNNs) were predominant, used either at the core of the forecasting model [2, 3, 5, 11, 21], or as a component in a larger model [26, 29]. Other types of neural architectures that were frequently used were convolutional RNNs (CRNNs) [3, 8, 9] and fully connected networks (FCNs) [2, 26, 28]. Generative Adversarial Networks (GANs) were used in [32], wherein the GRU-based generator uses real data as input and its BG predictions are pitted against the true BG values in a discriminator implemented using one-dimensional convolutional neural networks (CNNs). The recently proposed Neural Basis Expansion for Interpretable Time-Series Forecasting (N-BEATS) architecture [27] served as the basis for the winning entry [29]. In this top-performing model, the fully connected block structure of N-BEATS was replaced with LSTMs, additional losses were used to provide more supervision, and secondary, sparse variables such as meals and bolus insulin were used as input while still backcasting only on the primary forecasting variable, blood glucose. A number of non-neural approaches were proposed as well, such as Genetic Programming (GP) for symbolic regression in [14], Random Forests in [14, 28], multivariate Latent Variable (LV) based models in [30], and Partial Least Squares Regression (PLSR) with stacking in [15, 26].

	30 minutes		60 minutes		
Paper	RMSE	MAE	RMSE	MAE	Overall
[29]	18.22	12.83	31.66	23.60	86.31
[11]	19.21	13.08	31.77	23.09	87.15
[32]	18.34	13.37	32.21	24.20	88.12
[31]	19.05	13.50	32.03	23.83	88.41
[2]	18.23	14.37	31.10	25.75	89.45
[30]	19.37	13.76	32.59	24.64	90.36
[14]	19.60	14.25	34.12	25.99	93.96
[19]	20.03	14.52	34.89	26.41	95.85

Table 1: BGLP Challenge overall ranking.

The LSTM-based approach from [5] was notable for its interpretability analysis, wherein the SHAP (SHapley Additive exPlanations) method [18] was used to assess the impact that each feature has on the model predictions. Also of special interest were the "what-if" evaluations from [14], where future values of basal and bolus insulin were assumed to be controlled within the prediction horizon and leveraged with good results in some of the proposed GP-based models. Overall, the participating systems were trained or fine-tuned for each patient (personalized), with the exception of [2] where a single LSTM model was trained to make predictions for all patients (non-personalized).

We very much appreciate the support of the Digital ECAI 2020 workshop chairs, Magdalena Ortiz and Amparo Alonso, as well as this year's general chair Jérôme Lang. Further, we would like to thank Jernej Masnec, of Underline.io, the digital platform provider, for technical support.

We sincerely hope that the participants enjoyed this year's workshop program and that this collection of papers will inspire and encourage more AI-related research for and within healthcare in the future.

Kerstin Bach, Razvan Bunescu,
Cindy Marling and Nirmalie Wiratunga

Santiago de Compostela, virtually, August 2020

[1] http://smarthealth.cs.ohio.edu/bglp/bglp-rules.html
[2] http://smarthealth.cs.ohio.edu/bglp/bglp-results.html

REFERENCES

[1] Jeremy Beauchamp, Razvan Bunescu, and Cindy Marling, 'A general neural architecture for carbohydrate and bolus recommendations in type 1 diabetes management', in *this volume*, (August 2020).

[2] Robert Bevan and Frans Coenen, 'Experiments in non-personalized future blood glucose level prediction', in *this volume*, (August 2020).

[3] Ananth Reddy Bhimireddy, Priyanshu Sinha, Bolu Oluwalade, Judy Wawira Gichoya, and Saptarshi Purkayastha, 'Blood glucose level prediction as time-series modeling using sequence-to-sequence neural networks', in *this volume*, (August 2020).

[4] Eva Blomqvist, Marjan Alirezaie, and Marina Santini, 'Towards causal knowledge graphs - position paper', in *this volume*, (August 2020).

[5] Giacomo Cappon, Lorenzo Meneghetti, Francesco Prendin, Jacopo Pavan, Giovanni Sparacino, Simone Del Favero, and Andrea Facchinetti, 'A personalized and interpretable deep learning based approach to predict blood glucose concentration in type 1 diabetes', in *this volume*, (August 2020).

[6] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio, 'Learning phrase representations using RNN encoder-decoder for statistical machine translation', in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1724–1734, (2014).

[7] Diana Cristea, Christian Sacarea, and Diana Şotropa, 'Knowledge discovery and visualization in healthcare datasets using formal concept analysis and graph databases', in *this volume*, (August 2020).

[8] John Daniels, Pau Herrero, and Pantelis Georgiou, 'Personalised glucose prediction via deep multitask networks', in *this volume*, (August 2020).

[9] Jonas Freiburghaus, Aïcha Rizzotti-Kaddouri, and Fabrizio Albertetti, 'A deep learning approach for blood glucose prediction and monitoring of type 1 diabetes patients', in *this volume*, (August 2020).

[10] Alfonso Emilio Gerevini, Roberto Maroldi, Matteo Olivato, Luca Putelli, and Ivan Serina, 'Prognosis prediction in Covid-19 patients from lab tests and X-ray data through randomized decision trees', in *this volume*, (August 2020).

[11] Hadia Hameed and Samantha Kleinberg, 'Investigating potentials and pitfalls of knowledge distillation across datasets for blood glucose forecasting', in *this volume*, (August 2020).

[12] Mohammad Hesam Hesamian, Wenjing Jia, Sean He, and Paul Kennedy, 'Region proposal network for lung nodule detection and segmentation', in *this volume*, (August 2020).

[13] Sepp Hochreiter and Jürgen Schmidhuber, 'Long Short-Term Memory', *Neural computation*, **9**(8), 1735–1780, (1997).

[14] David Joedicke, Oscar Garnica, Gabriel Kronberger, José Manuel Colmenar, Stephan Winkler, Jose Manuel Velasco, Sergio Contador, and Ignacio Hidalgo, 'Analysis of the performance of genetic programming on the blood glucose level prediction challenge 2020', in *this volume*, (August 2020).

[15] Heydar Khadem, Hoda Nemat, Jackie Elliott, and Mohammed Benaissa, 'Multi-lag stacking for blood glucose level prediction', in *this volume*, (August 2020).

[16] Yumin Liu, Claire Zhao, and Jonathan Rubin, 'Uncertainty quantification in chest X-ray image classification using Bayesian deep neural networks', in *this volume*, (August 2020).

[17] Yunjie (Lisa) Lu, Abigail Koay, and Michael Mayo, 'In silico comparison of continuous glucose monitor failure mode strategies for an artificial pancreas', in *this volume*, (August 2020).

[18] Scott M. Lundberg and Su-In Lee, 'A unified approach to interpreting model predictions', in *Advances in Neural Information Processing Systems 30*, eds., I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, 4765–4774, Curran Associates, Inc., (2017).

[19] Ning Ma, Yuhang Zhao, Shuang Wen, Tao Yang, Ruikun Wu, Rui Tao, Xia Yu, and Hongru Li, 'Online blood glucose prediction using autoregressive moving average model with residual compensation network', in *this volume*, (August 2020).

[20] Cindy Marling and Razvan Bunescu, 'The OhioT1DM dataset for blood glucose level prediction: Update 2020', in *this volume*, (August 2020).

[21] Michael Mayo and Tomas Koutny, 'Neural multi-class classification approach to blood glucose level forecasting with prediction uncertainty visualisation', in *this volume*, (August 2020).

[22] Richard McShinsky and Brandon Marshall, 'Comparison of forecasting algorithms for type 1 diabetic glucose prediction on 30 and 60-minute prediction horizons', in *this volume*, (August 2020).

[23] Carlos Francisco Moreno-Garca, Truong Dang, Kyle Martin, Manish Patel, Andrew Thompson, Lesley Leishman, and Nirmalie Wiratunga, 'Assessing the clinicians' pathway to embed artificial intelligence for assisted diagnostics of fracture detection', in *this volume*, (August 2020).

[24] Paul Jarle Mork and Kerstin Bach, 'A decision support system to enhance self-management of low back pain: Protocol for the selfBACK project', *JMIR Res Protoc*, **7**(7), e167, (Jul 2018).

[25] Meike Nauta, Michel van Putten, Marleen C. Tjepkema-Cloostermans, Jeroen Bos, Maurice van Keulen, and Christin Seifert, 'Interactive explanations of internal representations of neural network layers: An exploratory study on outcome prediction of comatose patients', in *this volume*, (August 2020).

[26] Hoda Nemat, Heydar Khadem, Jackie Elliott, and Mohammed Benaissa, 'Data fusion of activity and CGM for predicting blood glucose levels', in *this volume*, (August 2020).

[27] B.N. Oreshkin, D. Carpov, N. Chapados, and Y. Bengio, 'N-BEATS: Neural basis expansion analysis for interpretable time series forecasting', in *ICLR*, (2020).

[28] Jacopo Pavan, Francesco Prendin, Lorenzo Meneghetti, Giacomo Cappon, Giovanni Sparacino, Andrea Facchinetti, and Simone Del Favero, 'Personalized machine learning algorithm based on shallow network and error imputation module for an improved blood glucose prediction', in *this volume*, (August 2020).

[29] Harry Rubin-Falcone, Ian Fox, and Jenna Wiens, 'Deep residual time-series forecasting: Application to blood glucose prediction', in *this volume*, (August 2020).

[30] Xiaoyu Sun, Mudassir Rashid, Mert Sevil, Nicole Hobbs, Rachel Brandt, Mohammad Reza Askari, Andrew Shahidehpour, and Ali Cinar, 'Prediction of blood glucose levels for people with type 1 diabetes using latent-variable-based model', in *this volume*, (August 2020).

[31] Tao Yang, Ruikun Wu, Rui Tao, Shuang Wen, Ning Ma, Yuhang Zhao, Xia Yu, and Hongru Li, 'Multi-scale long short-term memory network with multi-lag structure for blood glucose prediction', in *this volume*, (August 2020).

[32] Taiyu Zhu, Xi Yao, Kezhi Li, Pau Herrero, and Pantelis Georgiou, 'Blood glucose prediction for type 1 diabetes using generative adversarial networks', in *this volume*, (August 2020).

KDH Workshop Co-Chairs

- Kerstin Bach, Norwegian University of Science and Technology
- Cindy Marling, Ohio University
- Nirmalie Wiratunga, The Robert Gordon University

BGLP Challenge Co-Chairs

- Razvan Bunescu, Ohio University
- Cindy Marling, Ohio University

Steering Committee

- Kerstin Bach, Norwegian University of Science and Technology
- Sadid Hasan, Philips Research North America
- Zina Ibrahim, King's College London
- Cindy Marling, Ohio University
- Jonathan Rubin, Philips Research North America
- Nirmalie Wiratunga, The Robert Gordon University
- Honghan Wu, University of Edinburgh

Program Committee

- Isabelle Bichindaritz, State University of New York at Oswego
- Ali Cinar, Illinois Institute of Technology
- José Manuel Colmenar, Universidad Rey Juan Carlos
- Alexandra Constantin, Bigfoot Biomedical
- Iván Contreras, Universidad Complutense de Madrid
- Vivek V Datla, Philips Research North America
- Antonio Della Cioppa, University of Salerno
- Spiros Denaxas, University College London
- Franck Dernoncourt, Massachusetts Institute of Technology
- Andrea Facchinetti, University of Padova
- Ian Fox, University of Michigan
- Pau Herrero, Imperial College London
- J. Ignacio Hidalgo, Universidad Complutense de Madrid
- Fernando Koch, IBM Global Services
- Kezhi Li, University College London
- Stewart Massie, Robert Gordon University
- Stefania Montani, University of Piemonte Oriental
- Stavroula Mougiakakou, University of Bern
- Tristan Naumann, Microsoft
- Alexander Schliep, Gothenburg University
- Thomas Searle, Kings College London
- Giovanni Sparacino, University of Padova
- Shawn Stapleton, Philips Research North America
- Lukas Stappen, University of Augsburg
- Josep Vehi, University of Girona
- Anjana Wijekoon, The Robert Gordon University

Accepted Papers

Interactive Explanations of Internal Representations of Neural Network Layers: An Exploratory Study on Outcome Prediction of Comatose Patients

Meike Nauta[1] and **Michel J.A.M. van Putten**[1,2] and **Marleen C. Tjepkema-Cloostermans**[2] and **Jeroen Peter Bos**[1] and **Maurice van Keulen**[1] and **Christin Seifert**[1]

Abstract. Supervised machine learning models have impressive predictive capabilities, making them useful to support human decision-making. However, most advanced machine learning techniques, such as Artificial Neural Networks (ANNs), are black boxes and therefore not interpretable for humans. A way of explaining an ANN is visualizing the internal feature representations of its hidden layers (neural embeddings). However, interpreting these visualizations is still difficult. We therefore present InterVENE: an approach that visualizes neural embeddings and interactively explains this visualization, aiming for knowledge extraction and network interpretation. We project neural embeddings in a 2-dimensional scatter plot, where users can interactively select two subsets of data instances in this visualization. Subsequently, a personalized decision tree is trained to distinguish these two sets, thus explaining the difference between the two sets. We apply InterVENE to a medical case study where interpretability of decision support is critical: outcome prediction of comatose patients. Our experiments confirm that InterVENE can successfully extract knowledge from an ANN, and give both domain experts and machine learning experts insight into the behaviour of an ANN. Furthermore, InterVENE's explanations about outcome prediction of comatose patients seem plausible when compared to existing neurological domain knowledge.

1 Introduction

Most advanced artificial intelligence techniques, such as Artificial Neural Networks (ANNs), are black boxes and therefore not interpretable for humans. As argued by [22], it depends on the application to what extent this is a concern. Interpretability is critical in the case of this paper: outcome prediction of comatose patients, where AI is meant to support the physician in (high-stakes) decision making. "Explainable AI" (XAI) is essential for medical professionals to understand the how and why of the AI's decision, for example, to be able to confirm that the system is right for the right reasons [23]. Moreover, interpretable algorithms (i.e. algorithms that explain or present their decision to a human in understandable terms [5]) could appropriately enhance users' trust in future AI systems [22]. Since a poor predicted diagnosis may cause care to be reduced, getting insights in the algorithm may reveal predictions that cannot be trusted, thus allowing to save a patient's life. On the other hand, AI might discover patterns that were not known to the medical profession before, leading to knowledge discovery for healthcare improvement.

One way of explaining an ANN is visualizing the internal feature representations of its hidden layers (called *neural embeddings*) [26]. Solely relying on such visualization techniques is, however, insufficient: the resulting projection does not explicitly show the relations between projected points and the original input features, making it challenging to understand why data points are placed far apart or close together. Interviews revealed that data analysts try to map the synthetic data dimensions to original input features, and that they try to name and verify clusters [3]. In this paper, we therefore *explain a visualization*.

Contributions Since visualizations of neural embeddings are too complex to readily grasp, we explain a visualization with an easy-to-understand and effective interactive approach, suitable for both domain and machine learning experts. After visualizing neural embeddings in a scatter plot with dimensionality-reduction, a user can interact with the visualization by selecting two sets of data points in the visualization. We explain the difference between these subsets by training a decision tree (or another interpretable model) that distinguishes between the user-selected subsets in the visualization in terms of the original input features. Allowing manual selection of data points in the visualization results in a personalized explanation that gives meaning to clusters seen in the visualization where the user had specific interest in. An overview of the overall process is shown in Figure 1.

Our approach serves two goals: (i) knowledge discovery and knowledge validation: extracting knowledge from the neural network allows the domain expert to observe and validate patterns learnt by the neural network, and (ii) network interpretation: understanding the neural network allows the machine learning expert to analyse errors, behaviour over training time and contributions of single layers.

We implemented our approach in a tool called InterVENE (Interactively Visualizing and Explaining Neural Embeddings)[3]. We apply InterVENE to the medical domain where interpretability of a decision support system is critical: outcome prediction for postanoxic coma patients, based on a structured dataset containing features of EEG recordings of 518 comatose patients.

[1] University of Twente, the Netherlands, email addresses: {m.nauta, m.j.a.m.vanputten, m.vankeulen, c.seifert}@utwente.nl, j.p.bos@student.utwente.nl

[2] Department of Neurology and Clinical Neurophysiology, Medisch Spectrum Twente, the Netherlands, email addresses: {m.vanputten, m.tjepkema-cloostermans}@mst.nl

[3] InterVENE is open-sourced at `https://github.com/M-Nauta/InterVENE`

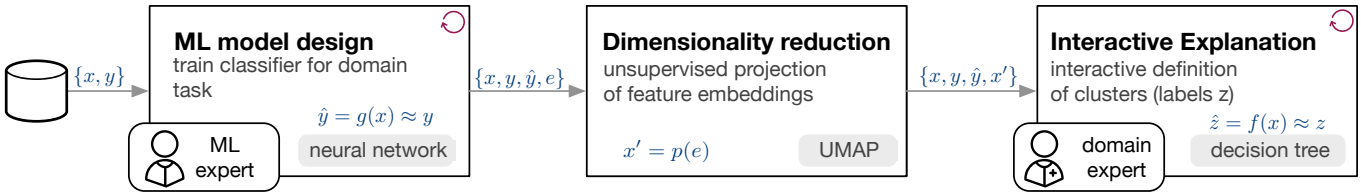


Figure 1. Overview of the approach. A machine learning model (e.g. an artificial neural network) is trained, the feature vectors of its hidden layers (embeddings) are projected into 2D. Domain experts then can interactively explore the visualizations and select groups of interest, whose difference is explained by an interpretable discriminative model (e.g. a decision tree).

2 Related Work

Explanation approaches of machine learning models address different stakeholders and explain different aspects of the model. Three general approaches have emerged towards providing explanations [7]. First, explanations can be model-based by showing the operational procedure of the complete model. Some machine learning models are considered inherently interpretable, e.g. decision trees or decision rules [6] or can be extended to be inherently interpretable, such as Deep Neural Decision Trees [31]. More complex models, such as deep neural networks can be locally or globally approximated by interpretable models (e.g., [8]). Other explanations approaches show similar cases (e.g., [20]), or the contributions of features for a decision (e.g., [12]). In this paper, we use decision trees as interpretable models to locally explain subgroups that emerge from the implicit feature representations of end-to-end machine learning models.

Dimensionality reduction techniques are suitable for visualizing high-dimensional data by projecting it on a low-dimensional space while preserving as much of the original data structure. Although many projection techniques have been proposed (we refer the reader to [14] for a review), t-SNE [15] is arguably the best known and most applied technique, due to its capability of capturing both the local and global structure of the high-dimensional data. It provides a 2- or 3-dimensional feature representation that can be visualized in a plot. Recently, the Uniform Manifold Approximation and Projection (UMAP) algorithm was introduced [16], which preserves as much of the local and more of the global data structure than t-SNE with faster run times [2]. UMAP uses the nearest neighbour descent algorithm [4] to construct a weighted k-neighbour graph that approximates the representation of the high dimensional data. It then optimizes this low dimensional layout via probabilistic edge sampling and negative sampling [17]. It has been shown that UMAP provides meaningful visualizations for biological data [2], materials science [13] and image classification [16].

Dimensionality reduction can also be applied to **derived features**, such as embeddings. An embedding is a vector containing the post-activation values in a hidden layer of an Artificial Neural Network (ANN). The learned, continuous embedding is therefore a representation of the input data. Visualizing the embeddings of each hidden layer provides a general overview of the inner behavior of the ANN [25]. Most existing work on visualizing embeddings was created for image data, using e.g. heat maps or pixel displays [25]. Dimension-reduction techniques such as t-SNE and UMAP however, can be used for visualizing the embeddings of any type of data. Data instances with similar representations in a network layer will be close in the projection space. It has already been shown that t-SNE projections of neural embeddings (both after training and during training) can aid the understanding and improving of ANNs [21], since it allows users to view global geometry and to discover clusters [26]. The embedding projector of [26], using t-SNE or Principal Component Analysis, is therefore integrated into the Tensorflow platform [1].

Although dimension-reduction is considered an *explainable* method because data is represented in a lower-dimensional space, users often have difficulty **interpreting the dimensions of the visualization** in a meaningful way [14]. It is therefore needed to *explain the visualization*. For example, [27] introduced *probing*, a tool to understand projections by exploring the dimensionality-reduced data and to interact with the visualization to examine errors. [11] explains a visualization by creating just-in-time descriptions to automatically identify and annotate visual features, such as clusters and outliers. In contrast, our approach provides personalized explanations by explaining the difference between user-chosen clusters with an interpretable model. Our approach can be applied to any dimensionality reduction method that visualizes embeddings, such as t-SNE [15], UMAP [16], DarkSight [30] or the existing embedding projector of [27].

Reliable **prediction of neurological outcome in comatose patients** after cardiac arrest is challenging. It may prevent futile care in patients with a poor prognosis, and allow timely communication with family members about the neurological condition. Early EEG recordings have been shown to allow reliable prognostication within 24 hours after arrest in a significant fraction of patients [9]. However, visual analysis by the neurologist is time-consuming and allows reliable prediction of neurological outcome of approximately 50% of patients, only. This motivates the need for techniques that assist or even replace the human expert, as resources are limited, and hold promise to increase the diagnostic yield. Deep neural networks have recently been used to predict neurological outcome [29], showing a significant improvement in classification accuracy. A limitation of these approaches, however, is the lack of interpretability, which is what we address in this paper.

3 Approach

Our approach consists of three general steps, integrating two stakeholders: the machine learning expert and the domain expert. As shown in Figure 1, the machine learning expert first **trains a predictive model** for the task at hand on the labeled data $\{x, y\}$, with x being the feature vector and y the respective label. This task is iterative and usually includes model selection and hyper-parameter optimisation steps. The output is the machine learning model, and its predictions $\hat{y}$ on the data, which ideally should match the ground-truth labels y. For the remainder of this paper, we assume that these models generate an implicit feature representation to solve their prediction task. In end-to-end (deep) learning scenarios these feature representations are the activations in the hidden layers of the neural networks. We refer to these representations as embeddings e. There can be more than one embedding for one training sample, e.g. the embeddings of the first hidden layer, of the second and so on. Em-

beddings can also be collected for different training epochs, e.g. after training the network for 10 epochs versus training for 50 epochs.

The embeddings of a layer of a certain epoch are then **projected into a 2-dimensional feature space**, resulting in one visualization for each layer. More specifically, the projection function p takes embeddings e from feature vector x and generates a low-dimensional representation x'. In principle, any projection of dimensionality reduction method can be used in this step. We choose a 2-dimensional scatter plot as visualization since this is easy to interpret. In our implementation called InterVENE, we use UMAP as dimension-reduction technique, because of its good run time performance and data structure preservation (cf. Section 2). An example of the visualizations is shown in Figure 4.

The user can select one of the visualizations (i.e. one of the layers at a certain training epoch), to inspect this projection in more detail. The selected projection x' is then used in an **interactive visualization** to show the patterns the machine learning model implicitly learned to best solve the prediction task. We visualize the following information about the machine learning model: i) an approximation of the learned feature space by the model x' (using 2d-position as visual channel), ii) the prediction of the model $\hat{y}$ (using color hue as visual channel), iii) whether the decision is a true positive, a false positive, a true negative or a false negative (using shape as visual channel). As shown in Figure 2, the visualization allows for interactive selection of subgroups of interest by either lasso selection with a mouse and/or choosing pre-selected categories, such as true and false positives or positives. The advantage of manual selection compared to automated clustering is that users can generate explanations about subgroups of data points where they have specific interest in. Data instances in these subgroups implicitly get assigned labels z indicating to which selection they belong. The *difference between the selected subgroups* is then explained by training an interpretable machine learning model to approximate the subgroup labels z based on the input features x. InterVENE uses a decision tree that classifies the selected data points to one of the selected subgroups. An example of this visualization and the explanation is shown in Figure 3.

4 Case Study and Data Set

InterVENE can be applied to any machine learning model that generates embeddings trained on any labeled dataset, since UMAP has no computational restrictions on embedding dimension [16] and the underlying techniques of InterVENE are generally applicable. To show the importance and benefits of InterVENE, we perform an exploratory study on a medical case where interpretability is critical. We use InterVENE to better understand a neural network predicting the neurological outcome of comatose patients after cardiac arrest. These predictions may prevent futile care in patients with a poor prognosis, as discussed in Section 2. Our structured dataset contains features from continuous EEG recordings of 518 prospectively collected adult patients who were comatose after cardiac arrest (Glasgow Coma Scale score <8) and admitted to the ICU of the Medisch Spectrum Twente (June 2010-May 2017) or Rijnstate hospital (June 2012-April 2017). Details have been described previously by Hofmeijer et al. [9].

The primary outcome measure was the Glasgow-Pittsburgh Cerebral Performance Category (CPC) score at 6 months, dichotomized as good (CPC 1 or 2, no or moderate neurological deficits with independence in activities of daily living) and poor (CPC score $\geq$ 3, major disability, coma, or death). The CPC scores were obtained prospectively by telephone follow-up with the patient or patient's legal representative. Part of the data is used in work of [28] and [19].

EEG features We extracted 42 qEEG features from each 10 second EEG segment at 24 hours after arrest to quantify 5 minute EEG epochs, broadly grouped into three domains: (i) time domain features capturing time varying amplitude information of the EEG signal; (ii) frequency domain features that capture key EEG patterns in different sub-bands in the spectrogram; and (iii) entropy domain features providing measures of complexity and randomness of the EEG signal. The median values of features across all channels were averaged for each 5 minute EEG epoch resulting in 42 features per patient. We used the same features as described by [19], except for 2 entropy features due to long calculation times. The data is scaled to have zero mean and unit variance, such that it can be used as input for a neural network. The dataset was complemented by a smaller set of features as described in [28] which are easier to interpret by the neurologist. These features are only used in an extra explanation, as discussed in Section 6.1.

5 Experimental Setup

Neural Network Architecture Since InterVENE can be used on any neural network architecture, the aim of this paper is not to train the best neural network for postanoxic coma classification, but to find a reasonable well-performing model which we want to explain. To find such a model, we optimized hyperparameters using grid search and stratified 5-fold cross validation on a training set (80% of the dataset) for a simple feedforward ANN. The following sets of hyperparameters were investigated by grid-search: #hidden layers $\{1, 2, 3\}$, #nodes in a hidden layer[4]: $\{5, 10, 20\}$, dropout: $\{$yes with $p = 0.5$, no$\}$, learning rate: $\{0.1, 0.01\}$, weight initialization: $\{\mathcal{N}(0, 1), \mathcal{N}(0, 0.1)\}$. To limit the number of networks to train, we fixed the activation function to ReLu (and Sigmoid in the output layer) and used the Adam optimizer. The most accurate network was a network with 2 hidden layers, with 20 hidden nodes in each hidden layer, dropout with $p = 0.5$, learning rate of 0.01 and weight initialization of $\mathcal{N}(0, 0.1)$. Using a hold out set, we decided to stop training after 100 epochs. Our network had an accuracy of 0.79 when using a threshold of 0.5 and AUC = 0.88. Because of a very high cost of error, prediction performance for poor outcome is in medical literature usually measured in recall (sensitivity) at 100% specificity. Our network has a recall of 0.34 for poor outcome at 100% specificity, outperforming the 0.32 of [28] and lower than state-of-the-art 0.44 [19], showing that our network is reasonably good.

Hyperparameters InterVENE For UMAP, we mainly use the default settings of the UMAP python package (v0.3). However, we set the number of neighbours n to 10 (default is 15) to more accurately catch the detailed manifold structure [16], and tuned the visualization by setting the distance metric to 'correlation'. For the decision tree, we use the default parameters of the scikit-learn decision tree package (v0.20.03). To improve interpretability and prevent overfitting, we set the maximum depth to 3 (but this parameter can be changed by the user), require a minimum number of 5 samples in each leaf and prune the tree such that a node is not split when all leaves have the same class label.

[4] with the restriction that the number of hidden nodes in a layer cannot exceed the number of hidden nodes in the previous layer

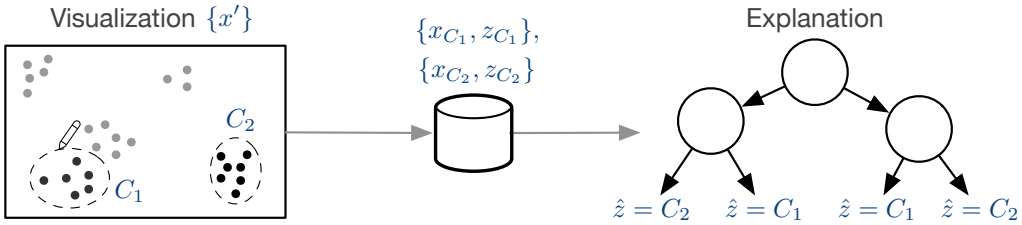


Figure 2. Overview of the interactive explanation. A user selects two groups of interest in visualization x', getting labels $z = C_1$ and $z = C_2$. Based on the input features x_{C_1}, x_{C_2}, a decision tree is trained to predict the corresponding clusters $\hat{z}$. The learnt decision tree thus explains the difference between the two selected clusters.

6 Experiments

We performed two experiments: (i) we elicited which domain knowledge can be obtained using InterVENE (see Section 6.1) and (ii) investigated which understanding about the neural networks in terms of error type and training process can be gained (see Section 6.2). We used the *constructive interaction* evaluation protocol in which two participants naturally communicate and collaborate in trying to solve predetermined tasks [18]. One participant in our study is a machine learning expert knowledgable about the projection method and neural networks. The other participant is a neurologist from Medisch Spectrum Twente with a full understanding of the medical dataset. Furthermore, we compare whether the discovered knowledge is in correspondence with existing medical literature (extracted knowledge validation).

6.1 Experiment 1: Neural Visualization and Explanation for Knowledge Discovery

We let the neurologist compare its domain knowledge with the results from our visualization and explanations. For this, the neurologist selected the visualization of the embedding of the last hidden layer of the final ANN shown in Figure 3(a). Tasks for the constructive interaction study consist of two components: 1) Interpreting the visualization, based on shape and colors; 2) Interpreting the explanation: selecting clusters and comparing the learnt decision tree with domain knowledge.

When looking at the visualization in Figure 3(a), both participants clearly see that the ANN is able to identify between comatose patients with a predicted good neurological outcome (yellow-green, 'cluster 2') and a predicted poor outcome (purple, 'cluster 1'). To get more insight, the neurologist uses InterVENE to manually select these two clusters, after which a decision tree is trained to explain the difference between these clusters. The learnt decision tree (not shown here), with a depth of 3, classifies each instance in the dataset as one of the two clusters with an accuracy of 0.913. The top feature in the tree is 'Hilbert burst'. The neurologist finds this decision tree plausible, since burst suppression (an EEG pattern in which neural activity with high amplitudes alternates with quiescence) is characteristic for an inactivated brain [10]. To evaluate the results, the neurologist visually inspects a few EEG recordings from each cluster. The neurologist clearly sees differences between poor and good outcome and would have made the same prediction as our neural network did.

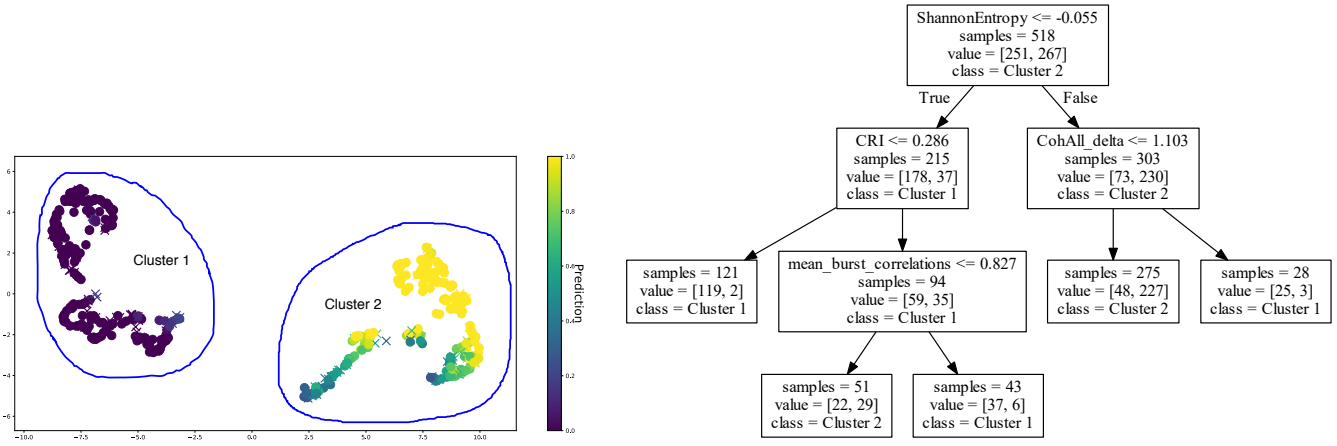
Based on the visualization in Figure 3(a), the neurologist is interested in studying the two sub-clusters within 'cluster 1', indicating that the ANN has learnt two different types of comatose patients that are expected to have a poor outcome. To explain this difference, the participants use the lasso selector of InterVENE to manually select these clusters. The resulting decision tree (not shown here) has a depth of 2 and an accuracy of 0.912 to classify an instance to one of the two purple clusters, with 'skewness' as top node. For evaluation of the result, a few EEGs from each subcluster are selected and analysed to evaluate whether the neurologist could see this same difference. Differentiation within 'cluster 1' was beyond visual assessment since skewness is a feature that cannot directly be read from an EEG; it can only be calculated.

Since manual classification is done by visual inspection of the EEG, features that cannot be directly read from an EEG are not used by neurologists in practice (although experiments like these might change this). This makes interpreting the features used in the decision tree more difficult. To solve this issue, we trained a second decision tree that only uses features that are easily interpretable by domain experts. Thus, while the visualizations are still based on the embeddings of an ANN trained on the 42 original features, the decision tree is only learnt from a dataset with 11 easy-to-interpret features (from the same patients). The decision tree shown in Figure 3 shows that Shannon entropy is the most important feature to distinguish between poor outcome ('cluster 1') and good outcome ('cluster 2'). This corresponds with existing literature which showed that Shannon entropy has the highest individual feature contribution for comatose patients 24 hours after cardiac arrest [28].

6.2 Experiment 2: Neural Visualization for Network Interpretation

In this experiment, we evaluate to what extent our approach aids neural network interpretation. We consider the visualizations of both hidden layers, and compare the visualizations of embeddings trained over time (after 10, 50 and 100 epochs). An overview with all the visualizations is shown in Figure 4. We defined the following tasks:
(i) See how the the neural network trains over time by comparing the visualizations of various epochs.
(ii) Interpret the relevance of hidden layers by comparing the visualization of deeper layers with visualizations of earlier layers.
(iii) Understand where the neural networks makes mistakes by analysing and comparing True Positives/Negatives with False Positives/Negatives.

Visualizations over time InterVENE shows how the artificial neural network (ANN) learns over time to give users a better understanding of the training process. From the top images in Figure 4, the two clusters indicate that the ANN is already able to distinguish between two groups of patients after training for 10 epochs. However, the ANN still makes many classification mistakes when trained for only 10 epochs. The lack of yellow and the presence of purple seem to indicate that the network quickly learns to recognize poor outcome, but is not confident yet about good outcomes. This is improved after 50 epochs, when the visualization clearly shows a yellow cluster



(a) Interactive Visualization (b) Explanation

Figure 3. UMAP visualization of the embedding of the 2nd hidden layer of the ANN, trained for 100 epochs. Each data point is a comatose patient. Color indicates the prediction. A circle indicates a correct classification; a cross indicates an incorrect classification (with prediction threshold 0.5). Users can interactively select 2 clusters of interest in the visualization using a lasso selector (blue lines). The difference between the selected clusters is explained by a decision tree, using only interpretable features (accuracy of 0.844). Each splitting node shows the binary expression, total number of samples, division of samples and class label.

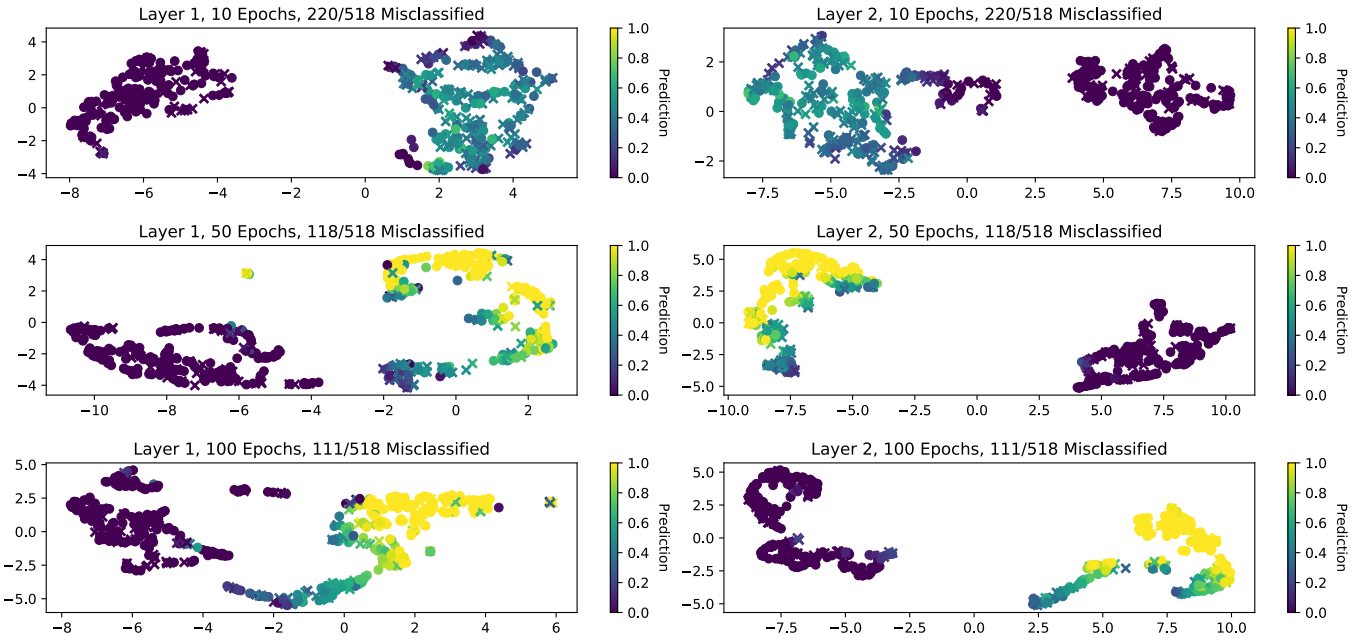


Figure 4. Visualized embeddings of two hidden layers over time. The left column shows the embedding of the first hidden layer, after the ANN is trained for 10, 50 resp. 100 epochs. The right column shows the embedding of the second hidden layer after 10, 50 resp. 100 epochs. The number of misclassifications indicates the number of incorrect classifications when a threshold of 0.5 is used to label the output of the ANN.

and a purple cluster. After 100 epochs, the second layer can clearly distinguish between good and poor outcome. Furthermore, it can better distinguish between yellow (confident about good outcome) and green (not confident about good outcome). Both the machine learning expert and domain expert found InterVENE useful to better understand the learning process of the ANN.

Relevance of hidden layers The participants noticed during the constructive interaction study that the visualizations of the first and second hidden layer do not differ substantially. Having comparable visualizations from different layers could indicate redundancy. Our grid-search found that a 2-layered model performed best but didn't tell how worse a 1-layer model would have been. To test whether the second layer is (almost) redundant, we trained an ANN with only one hidden layer, and compared its accuracy with the original 2-layered ANN. Whereas our ANN with 2 hidden layers misclassified 111 patients, the ANN with 1 hidden layer misclassified 120 patients. Since the increase is rather small, it confirms the hypothesis that the added value of the second layer is positive, but limited. This shows that InterVENE could be used for neural network pruning.

Analysing problematic instances InterVENE can also act as a starting point to further explore the training data and more quickly identify problematic training instances. As shown in Figure 5, users

can select a data subset such as true positives or false negatives using buttons. This gives more insights in the mistakes the ANN makes. When looking at the false negatives (Fig. 5), it is surprising to see that the ANN is very confident that some patients will have a poor outcome (purple) although in reality they had a good outcome. The neurologist is interested in this incorrect purple cluster, since a neural network incorrectly predicting a poor outcome (e.g. meaning that treatment can stop) can have severe consequences. InterVENE can explain the difference between e.g. true negatives and false negatives by learning a decision tree to distinguish between these two clusters. However, in our experiments the decision tree algorithm did not produce a decision tree that could distinguish between false negatives and true negatives. This means that patients with similar feature values in the dataset have different outcomes, which would explain why the ANN had difficulty to predict the correct outcome. This relevant information shows that some patients are not distinguishable, indicating that we might need to group them in a new class 'no safe prediction can be made'. We leave this 3-class prediction problem for future work.

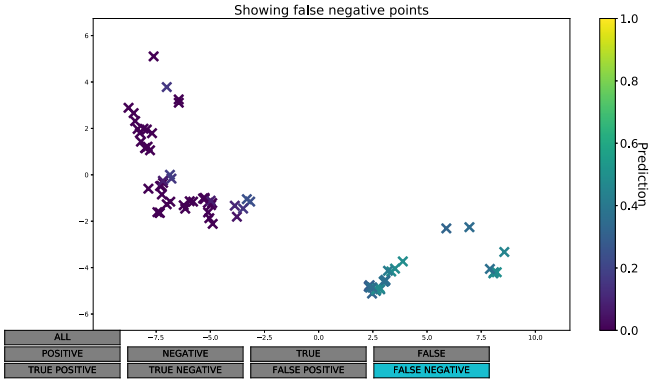


Figure 5. Screenshot of InterVENE. Button is used to only visualize false negatives, after which subgroup(s) can be selected by the user.

7 Discussion

InterVENE allows its users to interactively select groups of data points. However, literature raises concerns for clustering with t-SNE, which are salient for clustering the results of UMAP. In both methods, nearest neighbours are mostly preserved but distances between clusters and densities are not preserved well [24]. We therefore do not train an explainable method to predict the projected coordinates of an instance, but rather require it to only predict to which cluster a data instance belongs. However, users should still take these concerns into account when interpreting a UMAP visualization. Furthermore, although we applied InterVENE to only one dataset, we expect that InterVENE will perform well on other datasets. Since InterVENE let users select clusters instead of single datapoints, the quality of the visualization should not be impacted by the number of instances in the dataset. However, the number of features in a dataset might influence the quality of the explanation. Parameter tuning (either manual or automated) could ensure that an explanation is both accurate and interpretable (e.g. setting a max depth of the decision tree). Besides, InterVENE currently only allows 2-dimensional visualizations. This implementation can however easily be adapted, since UMAP supports higher dimensional visualizations.

8 Conclusion

Various projection techniques exist to visualize neural network embeddings. Since these visualizations are difficult to interpret, we presented an approach to *explain* these visualizations to aid knowledge discovery and network interpretation. We showed with a case study that users can get more insight in such a visualization by interactively selecting two subsets and comparing them with an interpretable, predictive model. Our implementation, called InterVENE, was applied to a structured dataset containing features about EEGs of comatose patients and is evaluated by a machine learning expert and domain expert (neurologist). An artificial neural network was trained to predict whether a patient would have a good outcome (e.g. wake up) or a poor outcome (e.g. further treatment is futile). After visualizing the neural embeddings, user can interact with InterVENE to generate a personalized decision tree which explains the difference between two user-selected subsets from the visualization. Our case study on comatose patients confirmed that InterVENE can successfully extract knowledge from a neural network. This knowledge was valuable for the neurologist and the explanations seemed plausible when compared with existing neurological domain knowledge. InterVENE also visualizes neural embeddings while the network is trained over time. Our experiments showed that this gives both domain experts and machine learning experts an idea of how the neural network is learning. Moreover, we showed that InterVENE can be used to judge the relevance of a hidden layer, by comparing the visualized embeddings of different hidden layers. For future work, we would like to apply InterVENE to other case studies, and train a network on raw EEG data and extract the learnt patterns from the network by learning a decision tree with hand-made features.

ACKNOWLEDGEMENTS

The authors would like to thank Duc Lê Trân Anh Dúc for improving the implementation of InterVENE.

REFERENCES

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al., 'Tensorflow: Large-scale machine learning on heterogeneous distributed systems', *arXiv preprint arXiv:1603.04467*, (2016).

[2] Etienne Becht, Leland McInnes, John Healy, Charles-Antoine Dutertre, Immanuel WH Kwok, Lai Guan Ng, Florent Ginhoux, and Evan W Newell, 'Dimensionality reduction for visualizing single-cell data using UMAP', *Nature Biotechnology*, **37**(1), 38, (2019).

[3] Matthew Brehmer, Michael Sedlmair, Stephen Ingram, and Tamara Munzner, 'Visualizing dimensionally-reduced data: Interviews with analysts and a characterization of task sequences', in *Proc. Workshop on Beyond Time and Errors: Novel Evaluation Methods for Visualization*, pp. 1–8. ACM, (2014).

[4] Wei Dong, Charikar Moses, and Kai Li, 'Efficient k-nearest neighbor graph construction for generic similarity measures', in *Proceedings of the 20th international conference on World wide web*, pp. 577–586. ACM, (2011).

[5] Finale Doshi-Velez and Been Kim, 'Towards a rigorous science of interpretable machine learning', *arXiv preprint arXiv:1702.08608*, (2017).

[6] L. H. Gilpin, D. Bau, B. Z. Yuan, A. Bajwa, M. Specter, and L. Kagal, 'Explaining Explanations: An Approach to Evaluating Interpretability of Machine Learning', *ArXiv e-prints*, (May 2018).

[7] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi, 'A survey of methods for explaining black box models', *ACM Comput. Surv.*, **51**(5), 93:1–93:42, (2018).

[8] G. Hinton, O. Vinyals, and J. Dean, 'Distilling the Knowledge in a Neural Network', *ArXiv e-prints*, (March 2015).

[9] J Hofmeijer, T.M.J. Beernink, F.H. Bosch, A Beishuizen, M.C. Tjepkema-Cloostermans, and M.J.A.M. van Putten, 'Early EEG contributes to multimodal outcome prediction of postanoxic coma', *Neurology*, **85**, 1–7, (2015).

[10] Jeannette Hofmeijer, Marleen C Tjepkema-Cloostermans, and Michel JAM van Putten, 'Burst-suppression with identical bursts: a distinct eeg pattern with poor outcome in postanoxic coma', *Clinical Neurophysiology*, **125**(5), 947–954, (2014).

[11] Eser Kandogan, 'Just-in-time annotation of clusters, outliers, and trends in point-based data visualizations', in *Proc. IEEE VAST*, pp. 73–82. IEEE, (2012).

[12] Klas Leino, Linyi Li, Shayak Sen, Anupam Datta, and Matt Fredrikson, 'Influence-directed explanations for deep convolutional networks', *CoRR*, **abs/1802.03788**, (2018).

[13] Xin Li, Ondrej E Dyck, Mark P Oxley, Andrew R Lupini, Leland McInnes, John Healy, Stephen Jesse, and Sergei V Kalinin, 'Manifold learning of four-dimensional scanning transmission electron microscopy', *npj Computational Materials*, **5**(1), 5, (2019).

[14] S. Liu, D. Maljovec, B. Wang, P. Bremer, and V. Pascucci, 'Visualizing high-dimensional data: Advances in the past decade', *IEEE Trans Vis Comput Graph*, **23**(3), 1249–1268, (March 2017).

[15] Laurens van der Maaten and Geoffrey Hinton, 'Visualizing data using t-SNE', *J. Mach. Learn. Res*, **9**(Nov), 2579–2605, (2008).

[16] Leland McInnes, John Healy, and James Melville, 'Umap: Uniform manifold approximation and projection for dimension reduction', *arXiv:1802.03426*, (2018).

[17] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean, 'Distributed representations of words and phrases and their compositionality', in *Advances in neural information processing systems*, pp. 3111–3119, (2013).

[18] Naomi Miyake, 'Constructive interaction and the iterative process of understanding', *Cognitive Science*, **10**(2), 151–177, (1986).

[19] Sunil B. Nagaraj, Marleen C. Tjepkema-Cloostermans, Barry J. Ruijter, Jeannette Hofmeijer, and Michel J.A.M. van Putten, 'The revised Cerebral Recovery Index improves predictions of neurological outcome after cardiac arrest', *Clinical Neurophysiology*, **129**(12), 2557–2566, (2018).

[20] Nicolas Papernot and Patrick D. McDaniel, 'Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning', *CoRR*, **abs/1803.04765**, (2018).

[21] Paulo E Rauber, Samuel G Fadel, Alexandre X Falcao, and Alexandru C Telea, 'Visualizing the hidden activity of artificial neural networks', *IEEE Trans Vis Comput Graph*, **23**(1), 101–110, (2017).

[22] Ariella Richardson and Avi Rosenfeld, 'A survey of interpretability and explainability in human-agent systems', *Proc. Workshop on Explainable Artificial Intelligence, IJCAI*, 137–143, (2018).

[23] Andrew Slavin Ross, Michael C. Hughes, and Finale Doshi-Velez, 'Right for the right reasons: Training differentiable models by constraining their explanations', in *Proc. of the 26th Int. Joint Conf. on Artificial Intelligence, IJCAI-17*, pp. 2662–2670, (2017).

[24] Erich Schubert and Michael Gertz, 'Intrinsic t-stochastic neighbor embedding for visualization and outlier detection', in *Proc. Conf. Similarity Search and Applications*, pp. 188–203. Springer, (2017).

[25] Christin Seifert, Aisha Aamir, Aparna Balagopalan, Dhruv Jain, Abhinav Sharma, Sebastian Grottel, and Stefan Gumhold, 'Visualizations of deep neural networks in computer vision: A survey', in *Transparent Data Mining for Big and Small Data*, 123–144, Springer, Cham, (2017).

[26] Daniel Smilkov, Nikhil Thorat, Charles Nicholson, Emily Reif, Fernanda B Viégas, and Martin Wattenberg, 'Embedding projector: Interactive visualization and interpretation of embeddings', *NIPS 2016 Workshop on Interpretable Machine Learning in Complex Systems*, (2016).

[27] Julian Stahnke, Marian Dörk, Boris Müller, and Andreas Thom, 'Probing projections: Interaction techniques for interpreting arrangements and errors of dimensionality reductions', *IEEE Trans Vis Comput Graph*, **22**(1), 629–638, (2016).

[28] Marleen Tjepkema-Cloostermans, Jeannette Hofmeijer, Albertus Beishuizen, Harold W Hom, Michiel J Blans, Frank H Bosch, and Michel JAM Van Putten, 'Cerebral recovery index: reliable help for prediction of neurologic outcome after cardiac arrest', *Critical care medicine*, **45**(8), e789–e797, (2017).

[29] Marleen Tjepkema-Cloostermans, Catarina Lourence, B.J. Ruijter, A. Beishuizen, Gea Drost, Frank H. Bosch, Francois H Kornips, J. Hofmeijer, and M.J.A.M. van Putten, 'Outcome prediction in postanoxic coma with deep learning', *Critical Care Medicine*, in press, (2019).

[30] Kai Xu, Dae Hoon Park, Chang Yi, and Charles Sutton, 'Interpreting deep classifier by visual distillation of dark knowledge', *arXiv preprint arXiv:1803.04042*, (2018).

[31] Yongxin Yang, Irene Garcia Morillo, and Timothy M. Hospedales, 'Deep neural decision trees', *CoRR*, **abs/1806.06988**, (2018).

Comparison of Forecasting Algorithms for Type 1 Diabetic Glucose Prediction on 30 and 60-Minute Prediction Horizons

Richard McShinsky[1] and **Brandon Marshall**[2]

Abstract. *Control of blood glucose (BG) levels is essential for diabetes management, especially for long term health improvement. Predicting both hypoglycemic events (BG $<$ 70 mg/dl) and hyperglycemic events (BG $>$ 180 mg/dl) is essential in helping diabetics control their long term health. In this paper we attempt to forecast future blood glucose levels, as well as analyze the efficiency of detecting both hypoglycemic events and hyperglycemic events. We do so by comparing Auto-Regressive Integrated Moving-Average, Vector Auto-Regression, Kalman Filter, Unscented Kalman Filter, Ordinary Least Squares, Support Vector Machines, Random Forests, Gradient Boosted Trees, XGBoosted Trees, Adaptive Neuro-Fuzzy Inference System (ANFIS), and Multi-Layer Perceptron in terms of Root Mean Squared Error, Mean Absolute Error, Coefficient of Determination, Matthews Correlation Coefficient, and Clarke Error Grid to compare their effectiveness in predicting future blood glucose levels, as well as predicting both hypoglycemic and hyperglycemic events.*

1 Introduction

Blood glucose prediction has been an ongoing challenge within the medical field due to the near unpredictable variability of the many underlying factors influencing an individual's glucose levels. There has been a strong drive recently to create an artificial pancreas using artificial intelligence, which has necessitated the need to predict future blood glucose levels as well as the ability to accurately predict the onset of both hypoglycemic (BG $<$ 70 mg/dl) and hyperglycemic (BG $>$ 180 mg/dl) events [11].

Most predictive models for blood glucose encompass a physiological profile that includes a person's insulin, meal absorption, and past blood glucose levels [13]. Various machine learning methods that have been attempted to predict future blood glucose levels with regards to this profile include Auto-Regressive Integrated Moving-Average (ARIMA, see [3], [4], [13], and [15]), Support Vector Machines and Kernel Regression (SVM, see [3], [12], [13], and [15]), Random Forests (RF, see [8], [12], [13], and [15]), Gradient Boosted Trees (see [8] and [15]), and Artificial Neural Networks (see REFERENCES).

Comparing papers on the results, accuracy, and effectiveness of the models is near impossible due to different data sets being used between them. This paper seeks to offer a comparison of as many models as possible on a single data set.

In this paper, we compare the effectiveness of several models, namely ARIMA, Vector Auto-Regression Moving-Average with Exogenous Regressor (VAR), Ordinary Least Squares (OLS), K-Nearest Neighbors (KNN), SVM, RF, Gradient Boosting, XGBoosting, Adaptive Neuro-Fuzzy Inference System (ANFIS), and Multi-Layer Perceptron. Additionally we attempt to use both the Kalman Filter and the Unscented Kalman Filter (UKF) to predict future blood glucose values. The Unscented Kalman Filter was chosen over the Extended Kalman Filter due to its ability to use state-space models to predict nonlinear functions. In comparing each of these model's effectiveness we use RMSE, MAE, the Matthew Correlation Coefficient (A commonly used metric for checking hypoglycemic and hyperglycemic events that roughly measures the quality of binary classifications) [4], and the Clarke Error Grid.

[1] Brigham Young University, USA, email: richard.mcshinsky@byu.net
[2] Brigham Young University, USA, email: brandon.marshall@byu.net

2 Data

2.1 OHIO T1DM

The data used for this comparison was the OhioT1DM data set, which was obtained as part of the second Blood Glucose Level Prediction Challenge [5]. This data set contains eight weeks worth of data for 12 people with type 1 diabetes. All contributors were on insulin pump therapy with continuous blood glucose monitoring (CGM). All pumps were of one of two brands, all life event data was reported via a custom smartphone app, and all psychological data was provided from a fitness band. The features themselves provided in the data set are: Date, Glucose Level, Finger Stick, Basal (Insulin), Basal Temperature, Bolus (Insulin), Meal (Carbohydrate Estimate), Sleep, Work, Stressors, Hypoglycemic Event, Illness, Exercise, Basis Heart Rate, Basis GSR, Basis Skin Temperature, Basis Air Temperature, Basis Steps, Basis Sleep, and Acceleration [5].

The train and test splits were given as part of the second Blood Glucose Level Prediction Challenge (see [5] for more details).

2.2 Preprocessing

The glucose readings are in about 5-minute increments while other reading are every minute. Other readings reported by the patient are at arbitrary times not aligned with the glucose readings. To combine them into one data frame to use for predicting glucose, the most important predictor, glucose levels, was made the main index. All other values were merged to the closest glucose values within the previous 4 minutes. For values that were not in this tolerance they were dropped from the data frame.

Most of these values that were dropped were due to missing data. There are many gaps where the meter was not recording glucose values. This could be times between taking it off and putting it on, the

hour or more it takes for the meter to get set up, or a day where the user just did not put it on. Leaving these gaps often resulted in large jumps in the training and testing data. These discontinuities would be a problem in training the models. To fill them we couldn't use interpolation methods as we are unable to know the future while predicting these values. Therefore, our method to extrapolate values for these times was to use a moving average. For example, for the first extrapolated missing value, we would use the mean of the previous 2 values. For the second we would use the mean of the previous 4 values. For the tenth we would use the mean of the previous 20 values, including the ten we had just extrapolated before that. This would happen in five minute increments until we reach the next actual value in the data frame. The last predicted value would be dropped and the data frame would continue as normal until a difference of more than 6 minutes between values was detected and this rolling average would extrapolate the missing values. The rolling average would eventually converge to the average value of all the data, but maintains the nature of the recent data. For example, if the person has had high blood glucose levels for the day, the filled data would stay high, but eventually move towards the mean of the person when using several days for large gaps. This was done since after a few hours, guessing where the person's data was going to start is nearly random guessing. Since the actual glucose values are essentially normally distributed, it is better to guess more towards the mean of the glucose levels. Meanwhile, the discontinuities were reduced by maintaining the local rolling mean. This resulted in many of the extrapolations ending very close to where the data continues from the discontinuity for this data.

3 Methods

We intend to compare many methods used for classical and regressive time series analysis. Thus, even though some methods are known to not perform well with blood glucose levels for this type of problem, they give a baseline to compare each successive method. In addition to the classical models, we used some models described in other papers about predicting glucose levels for comparison and potentially better parameter choices. Further, we chose some methods like VAR and ANFIS in order to compare methods not seen in the research found. The following subsections explain choices in why specific methods, parameters, and architecture were chosen.

3.1 Classical Methods

3.1.1 ARIMA

Even though ARIMA itself is a linear combination of a trend component, a seasonal component, and a residual component, we chose to use this model due to its classical use within time series analysis. Additionally, ARIMA was chosen due to its ability to allow us to choose the order of p and q for both the AR and MA parts of the model. These hyperparameters p and q were chosen using stats.models.orderselect, from which we found that p=2 and q=2 gave the lowest error. It should be noted that the data is nearly stationary to start, so a lag of 0 was used (as larger lags resulted in a worse error). The only data features used were the previous p blood glucose levels and the q corresponding error terms.

3.1.2 VAR

VAR is a vectored version of an AR model. This allows for more types of inputs to influence the prediction, rather than just simply using the previous p blood glucose values. VAR used the same parameters used in the ARIMA model described above.

3.1.3 Unscented Kalman Filter (UKF)

Whilst the Extended Kalman Filter (EKF) works well for linear projections, blood glucose levels are nonlinear in nature. Generally EKF can be thought of as the extension of a Gaussian Random Variable (GRV) through a linear system [14]. In the nonlinear case however, the EKF produces approximations to the values x_k, y_k, and K_k (the state, observation, and covariance for the system) [14]. In other words, the Extended Kalman Filter propagates a GRV through a first-order linearization of the nonlinear system [14].

The Unscented Kalman Filter also uses a Gaussian Random Variable, but instead uses a minimal set of carefully chosen sample points for which to propagate this GRV [14]. This is done by applying the unscented transformation to the selected sample points and then propagating these carefully chosen points through the system. Doing so allows for approximations that are accurate to the third order of a Taylor series expansion [14].

To summarize, the Unscented Kalman Filter selects carefully chosen points, applies the unscented transformation to these points, then performs the time update and measurement update as is standard in the Kalman Filter [14].

3.2 Regression and Ensemble Methods

Since the OhioT1DM data set is time series based, regular regression methods are not immediately available for us to use when forecasting data. However, we can transform the data into a regression problem by first redefining how the data is presented. Instead of each row in the data representing a single time step of the nineteen features, we instead redefine the data on the last six rows of data (we used the last 30 minutes of known information of data). Thus each row in the new reformatted data set now contains the last six known time steps with the labels being the future blood glucose values we wish to predict at each time step. Each label is the next six or twelve blood glucose values following the current time step in the OhioT1DM data set for the 30-minute and 60-minute prediction horizons respectively. In summary, each time step is reformatted to have a 6x19 feature space with each label having 6 or 12 values. With the data reformatted the following algorithms can be run.

3.2.1 Ordinary Least Squares

While the data is nonlinear in nature, it is possible that within a sufficiently small subset of the data (that is, for a sufficiently small time interval), the data may be quasi-linear. As with ODEs (where one can essentially linearize a nonlinear system) we seek to do something similar by attempting to fit affine functions to a sufficiently small time domain. Ordinary Least Squares (OLS) seeks to do this, fit an affine function (with a constant and error term), to the data set. In addition to regular OLS, we also run OLS with regularization terms, namely Lasso (L1 regularization), Ridge (L2 regularization), and Elastic Net (L1 and L2 regularization) all with α values of 1 for the regularization terms. We note that Lasso regularization gives us the advantage of feature reduction, allowing us to analyze which lags are most important in determining future blood glucose levels.

3.2.2 Support Vector Machines

We believe Support Vector Machine regression may be a useful method due to its ability to alter the kernel being used, thus allowing us to alter our definition of distance with regards to the data. Support Vector Machine (SVM) regression seeks to fit a hyperplane to the data with an ϵ-margin. Points that fall within this ϵ-margin are known as support vectors and are used to help define the hyperplane used in the regression. Notions of distance to this hyperplane are defined using a kernel. We attempt to use an RBF-kernel (with a scaling γ value) and a Polynomial Kernel (with a scaling γ value, a constant term of 0 and a power of 3) in our regressions. Each SVM had an ϵ-margin of 0.1. The results for each of the SVMs are reported under RBF, Poly, and Sig respectively.

3.2.3 K-Nearest Neighbors

It is likely that previous patterns in the lags of blood glucose (and other features) may be similar to the current pattern in the lags of features, we believe KNN regression may also be a useful regression method. KNN uses a voting method to form the regression. Using a defined metric of distance, KNN regression finds the K closest neighbors to the given data point and then returns the average of the labels. We use five neighbors, along with Euclidean distance for this algorithm. The results for this algorithm are reported under KNN.

3.2.4 Random Forest Regression

Random Forest Regression is an ensemble method that combines weak decision-tree regressors to form a strong group regressor, Random Forests allow us to create a regressor that branches based on the features. This is included here due to its use in other papers attempting blood glucose prediction (see [8], [12], [13], and [15]). To limit run-time to a reasonable length, a max-depth of four was imposed on each forest.

3.2.5 Gradient Boosting

Another ensemble method that combines weak decision-tree regressors to form a strong group regressor, Gradient Boosting instead seeks to optimize the gradient of the loss function for each regressor. As this can perform well with the correct hyperparameters, we include this to see if the algorithm can outperform any of the aforementioned algorithms. In addition to using regular Gradient Boosted Trees, we also use an optimized version of this algorithm known as Extreme Gradient Boosted Trees (XGB). For Gradient Boosting a least-squares loss function, along with a learning rate of 0.1, and 100 estimators were used. For XBG a grid search was performed to find the optimal hyperparameters. Respectively, the results for these algorithms are reported under Grad and XGB.

3.3 Neural Networks

Much work has already been done implementing neural networks in many different forms, including CNN, CRNN, DCNN, LSTM, Jump neural Networks, and Echo State (see [1], [2], [3], [4], [6], [8], and [15]). Much of this work came from the Blood Glucose Level Prediction Challenge (BGLP) in 2018 using the OHIO T1DM data set.

3.3.1 ANFIS

ANFIS is a neural network that includes fuzzy logic principles. Fuzzy logic is about partial truths. Most neural networks have a true/false form in selections. Fuzzy logic models uncertainties. Some examples of this are what one considers warm/cold, fast/medium/slow, or high/low. Rather than just picking one or the other, a draw from a distribution can give a weighted random nature to the choices. ANFIS is designed to approximate nonlinear functions like glucose values. This was chosen due to the extremely accurate predictions in the referenced paper on chaotic systems. [9]

3.3.2 Multi-Layer Perceptron (MLP)

The Multi-Layer Perceptron (MLP) is a fully-connected, feed-forward neural network. This neural network can often find higher-order terms without having to create these higher-order terms. This reduces feature engineering of the data. Our MLP consists of three hidden layers, each with 100 nodes, and ReLu activation functions. The output layer for the regression is merely the output of the last affine function. Results are reported under MLP.

4 Metrics

The following metrics were used when evaluating the efficiency and accuracy of the algorithms:

4.1 Root Mean Square Error

The root mean square error (RMSE) is defined as $\sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y_i} - y_i)^2}$ where $\hat{y_i}$ is the predicted value and y_i is the actual value. RMSE has the advantage of an easily defined gradient, easy interpretability, and taking the square root of the squares transforms the error back to the original function space (that is, the RMSE value is in the same units as our label). This is the first metric used in evaluating the accuracy of the regression models.

4.2 Mean Absolute Error

The mean absolute error (MAE) is defined as $\frac{1}{n}\sum_{i=1}^{n} \mid \hat{y_i} - y_i \mid$. This error function is easy to define, is fairly robust against outliers, and will be in the same units as our label. However, the gradient is not always easy to define (and may not exist). This is the second metric used in evaluating the accuracy of the regression models.

4.3 Coefficient of Determination

The coefficient of determination (R^2) is defined as

$$1 - \frac{\sum_{i=1}^{n} \epsilon_i^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

where y_i is the actual value, $\hat{y_i}$ is the predicted value, $\epsilon_i = y_i - \hat{y_i}$ and is defined as the i^{th} residual, and $\bar{y}$ is the sample mean. The coefficient of determination gives a measure of how much variance is explained by the model. Values near 1 indicate nearly all variance is explained by the model, while values near 0 indicate the variance may be caused by other factors. We note that negative values are possible, and for this paper indicate poor performance from the model.

4.4 Matthews Correlation Coefficient

The Matthews Correlation Coefficient (MCC) is defined as $\frac{(TP*TN)-(FP*FN)}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$ where TP, FP, FN, TN stand for the true positive, false positive, false negative, and true negative rates respectively [4]. This metric gives a general idea of how well an algorithm does in predicting glycemic events. Values near 1 show the predictions correlate with the actual glycemic events. Values near 0 indicate the algorithm does no better than random guessing. Values near -1 indicate negative correlation (that is the predictions correlate with the opposite of the glycemic event). This metric is commonly used by many articles that attempt to predict blood glucose levels (see [4] for one such example), and as such is used here.

4.5 Clarke Error Grid

The Clarke Error Grid plots the actual blood glucose values against the predicted blood glucose values and is used as an indication of the potential results that may occur for a given prediction. The grid is split into 5 zones A-E. Predictions in Zone A and B are generally considered safe predictions and would not result in any negative effects on the patient. Predictions in Zone C would result in unnecessary treatment. Predictions in Zone D indicate a potentially dangerous failure to detect a glycemic event. Predictions in zone E would confuse treatment of hypoglycemia for hyperglycemia and vice versa (see [1]). Points in Zone E are considered extremely dangerous, as treatment due to these results could result in the patient's death. For this paper, in addition to MCC we use the percentage of points within each zone to evaluate the accuracy of a model's predictions.

5 Results

The following tables describe the average of the metric scores from the 6 patients. Each of these metrics are described above, namely RMSE, MAE, MCCs, and R^2. The abbreviation definitions and explanations can be found in the Methods section above.

Table 1. Metric Averages for 30-minute Prediction Horizon

Method	RMSE	MAE	MCC_l	MCC_h	R^2
OLS	20.53	14.14	0.34	0.79	0.86
Lasso	20.58	14.22	0.32	0.79	0.85
Ridge	20.52	14.13	0.35	0.79	0.86
Elastic	20.56	14.20	0.31	0.79	0.86
RBF	24.89	16.96	0.14	0.74	0.79
Poly	31.73	22.51	-0.00	0.70	0.66
KNN	24.57	17.07	0.30	0.73	0.79
RF	23.00	16.27	0.16	0.76	0.82
Grad	21.37	14.87	0.17	0.78	0.84
XGB	24.62	17.29	0.34	0.74	0.79
Kalman	24.08	24.08	0.40	0.74	0.78
UKF	29.88	20.65	0.30	0.67	0.69
ARIMA	23.73	16.68	0.12	0.75	0.81
VAR	25.25	17.05	0.36	0.74	0.79
ANFIS	24.56	16.52	0.26	0.76	0.80
MLP	20.85	14.30	0.30	0.78	0.85

Table 2. Metric Averages for 60-minute Prediction Horizon

Method	RMSE	MAE	MCC_l	MCC_h	R^2
OLS	33.42	24.65	0.02	0.61	0.62
Lasso	33.41	24.67	0.02	0.61	0.62
Ridge	33.41	24.65	0.02	0.61	0.62
Elastic	33.40	24.67	0.02	0.61	0.62
RBF	36.76	26.53	-0.00	0.55	0.54
Poly	39.16	29.31	-0.00	0.53	0.48
KNN	38.11	28.01	0.15	0.53	0.50
RF	35.20	26.08	0.09	0.58	0.58
Grad	33.98	24.96	0.08	0.58	0.61
XGB	39.78	26.97	0.15	0.53	0.46
Kalman	22.77	15.28	0.41	0.75	0.81
UKF	29.78	20.65	0.30	0.66	0.69
ARIMA	36.39	26.93	0.01	0.56	0.54
VAR	35.06	19.56	0.16	0.70	0.54
ANFIS	36.87	26.53	0.12	0.59	0.56
MLP	35.59	25.81	0.06	0.59	0.57

6 Analysis

In an attempt to first analyze the accuracy of these predictions we first analyze the RMSE and MAE for both the 30-minute and 60-minute prediction horizons (Tables 1 and 2). As a general guideline we will first analyze which model we believe is performing best among the patients. Once this is done we will then analyze general trends we have noticed while analyzing this data.

6.1 30-Minute Prediction

We note that in terms of the above defined metrics OLS, Lasso, Ridge, and Elastic Net Regression perform nearly identical. Thus, since the differences between OLS, Ridge, Lasso, and Elastic Net regression yield minimally different results, we consider Lasso to be the best model for the 30-minute blood glucose predictions. Lasso regression offers a natural form of feature selection which allows us to analyze which lags are most important for predicting future blood glucose levels. A further analysis of the feature relevancy can be found under section 6.4.

Even though we have identified Lasso regression as the best performing algorithm among those tested for the 30-minute prediction horizon, this means little if this "best" algorithm still yields subpar results. As such, we analyze Lasso regression both in terms of MCC and the Clarke Error Grid to determine if these results are "sufficiently adequate" for blood glucose prediction. To see general trends for the prediction we analyze the results for actual and predicted values across time for patients 540 and 584.

Note the Clarke Error Grid for patients 540 and 584 for the 30-minute prediction horizon (figure 2). The closer the points fall onto the bottom left to top right diagonal the better the predictions are considered. Analyzing these plots visually does not raise any immediate concerns for the predictions. Most values appear to fall within zones A, B, and C. Analyzing the zones percentages (table 3) shows that Lasso has 96% accuracy for patient 540 and about 99% accuracy for patient 584. The major concern however is that the rest of these predictions fall within zones D-E, indicating these predictions may result in potentially dangerous care if acted on for the patient. Considering the high accuracy for each patient though, these results are considered "sufficiently accurate" for the 30-minute prediction horizon.

Analyzing the MCC for Lasso regression for the 30 minute horizon shows that the MCC tends to be about twice as high for hyper-

glycemic events than for hypoglycemic events. Given that the data tends to have many more values in the hyperglycemic range than the hypoglycemic this reflects more on the class imbalance more than the algorithm. This is seen due to all the algorithms having this trend. Further, this bias is reflected in the algorithm's predictions, as valleys in the predictions do not reach as low as the valleys in the actual data (see figure 1). Because of this, we note that the algorithms are less likely to predict hypoglycemic events as they are hyperglycemic events, a result that occurs due to the higher number of blood glucose values in the data.

6.2 60-Minute Prediction

Looking at the results for the 60-minute prediction horizon for the RMSE and MAE we find the surprising result that the Kalman Filter (not the Unscented Kalman Filter), performs best out of all the algorithms. Several explanations are possible as to why this occurs. One of these is that the Kalman filter seemed to dampen the predictions. Most of the other algorithms would keep predicting upwards for the hour predictions if the trend was going up beforehand. The Kalman filter seems to mainly shift the prediction horizon over (so the difference between the last known glucose value and the prediction for an hour later is minimal). Since it keeps the results in the typical ranges of glucose values it may avoid the poor scores from unusually strong spikes of predicted values. The scores may be the best, but they may still be very poor predictors for an hour out.

Considering the aforementioned problems with the Kalman filter, we analyze the "second" best algorithm. Since the general trends discussed in the 30-minute prediction horizon section still hold for the 60-minute prediction horizon (when we disregard the Kalman Filter), we conclude Lasso regression to be the next best algorithm to use. However, analyzing the difference between the 30-minute prediction horizon and the 60-minute prediction horizon raises several concerns with using Lasso regression for the 60-minute prediction horizon.

We noted earlier that Lasso regression tends to underfit with regards to hypoglycemic events. This problem is only exacerbated when the prediction horizon is extended to 60 minutes (see table 2). Here we notice the hypoglycemic MCC has reduced to near 0, indicating that Lasso prediction does no better than random guessing as to whether a hypoglycemic event is occurring. This is far from ideal for any diabetic patient. As well, we note that for the 60-minute prediction horizon, the accuracy of safe predictions degrades by about 2-3% (see table 3). While 94-97% accuracy is still fairly good, given that this reduction in accuracy results in 2-3% more dangerous predictions, and considering the fact that Lasso regression is unable to predict hypoglycemic events better than random guessing, we do **not** consider these predictions to be "sufficiently accurate" for the 60-minute prediction horizon. As such, our recommendation is to use the 30-minute prediction horizon.

6.3 Overall Trends

The biggest trend that we notice is that the models tend to underfit in regards to hypoglycemic events. That is, the predicted values do not reach as low as the actual blood glucose values do. This is noted in the hypoglycemic MCC for the 30-minute prediction horizon (see table 1) which gives on average a score at about 0.3. This indicates a general correlation in predicting hypoglycemic events, but not a strong one. Given that the average blood glucose levels on the test data were 159.42 mg/dl, 158.51 mg/dl, 134.92 mg/dl, 143.41 mg/dl, 172.71 mg/dl, and 148.23 mg/dl for patients 540, 544, 552, 567, 584, and 596 respectively the most likely reason that the MCC for hypoglycemic events is so low is due to class imbalance within the glucose levels. Since most glucose levels are generally high for the patients, the model overfits for higher glucose levels, and as such struggles to predict hypoglycemic events. A potential solution could be to upsample by "jittering" the smaller imbalanced class (adding small random perturbations to the existing smaller imbalanced class in order to create for data). See [7] and [10] for such an example.

6.4 Feature Relevancy

As stated earlier, one important benefit of Lasso regression is the ability to identify features important to glucose prediction. As seen in Table 4: glucose level, bolus, meal, and exercise are significant in predicting glucose levels (finger sticks are potentially significant, but they may be linearly dependent on glucose level). The Weights column is the sum of all 6 people's weight scores. The problem with the weights is the huge variability in the number of recorded data points. In an attempt to normalize the data, we created an Adjusted Weight. This is made by dividing the weights of each person by the number of recorded values for each person and summing all 6 of them together. This was multiplied by 1000 so the values would be about the same magnitude as the original weights. The lack of enough data for exercise is demonstrated here. Only 3 of the 6 people had values for exercise and one of them had only 4 values. This person in the Adjusted Weights had a score of 32 while the other two were about 1.5 and 2. More data points for these other categories would reduce the variance and more clearly identify what features are important.

7 Conclusion

We found that Lasso regression performed best out of the algorithms used for both the 30-minute prediction horizon and the 60-minute prediction horizon. While the results were adequate for the 30-minute prediction horizon, these quickly degraded for the 60-minute horizon. We found in general that the regression algorithms perform fairly well for predicting hyperglycemic events, but struggle for predicting hypoglycemic events. It is our opinion that further research should be done with regards to improving the prediction horizon for blood glucose prediction. Specifically, further research should be investigated into the effects of the volume of data on the prediction horizon. If an artificial pancreas is to become a reality, stable prediction horizons beyond 30-minutes are needed.

Furthermore, analyzing the coefficients of the Lasso model shows that glucose level, bolus, meal, and exercise are the most relevant features in producing forecasts for blood glucose levels. However, problems with sparsity among certain features reduce the relevancy of these features. As such, future research should include handling sparse features in a more robust way.

8 Additional Material

For those wishing to compare or reproduce work found in this paper, the related code can be found at `https://github.com/marshallb95/BloodGlucosePrediction/blob/master/Master.ipynb`.

REFERENCES

[1] A. Aliberti, I. Pupillo, S. Terna, E. Macii, S. Di Cataldo, E. Patti, and A. Acquaviva, 'A multi-patient data-driven approach to blood glucose prediction', *IEEE Access*, **7**, 69311–69325, (2019).

[2] J. Chen, K. Li, P. Herror, T. Zhu, and G Pantelis. Dilated recurrent neural network for short-time prediction of glucose concentration. Paper presented at the Third International Workshop on Knowledge Discovery in Healthcare Data at the 27th International Joint Conference on Artificial Intelligence and the 23rd European Conference on Artificial Intelligence, 2018.

[3] S. Fiorini, C. Martini, D. Malpassi, R. Cordera, D. Maggi, A. Verri, and A. Barla. Data-driven strategies for robust forecast of continuous glucose monitoring time-series. Paper presented at the 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), 2017.

[4] K. Li, J. Daniels, C. Liu, P. Herrero, and P. Georgiou, 'Convolutional recurrent neural networks for glucose prediction', *IEEE Journal of Biomedical and Health Informatics*, **24**, 603–613, (2019).

[5] C. Marling and R. Bunescu. The ohiot1dm dataset for blood glucose level prediction: Update 2020. In The 5th International Workshop on Knowledge Discovery in Healthcare Data, Santiago de Compostela, Spain, June, 2020, 2020.

[6] J. Martinsson, A. Schliep, B. Eliasson, C. Meijner, S. Persson, and O Mogren. Automatic blood glucose prediction with confidence using recurrent neural networks. Paper presented at the Third International Workshop on Knowledge Discovery in Healthcare Data at the 27th International Joint Conference on Artificial Intelligence and the 23rd European Conference on Artificial Intelligence, 2018.

[7] M. Mayo, L. Chepulis, and R. Paul, 'Glycemic-aware metrics and oversampling techniques for predicting blood glucose levels using machine learning', *PLoS ONE*, **14**, 0225613–0225632, (2019).

[8] C. Midroni, P. J. Leimbigler, G. Baruah, M. Kolla, A. J. Whitehead, and Y. Fossat. Predicting glycemia in type 1 diabetes patients: Experiments with xgboost. Paper presented at the Third International Workshop on Knowledge Discovery in Healthcare Data at the 27th International Joint Conference on Artificial Intelligence and the 23rd European Conference on Artificial Intelligence, 2018.

[9] A. Miranian and M. Abdollahzade, 'Developing a local least-squares support vector machines-based neuro-fuzzy model for nonlinear and chaotic time series prediction,', *IEEE Transactions on Neural Networks and Learning Systems,*, **24**, 207–218, (2013).

[10] N. Nnamoko and I. Korkontzelos, 'Efficient treatment of outliers and class imbalance for diabetes prediction', *Artificial Intelligence in Medicine*, **104**, 101805–101817, (2020).

[11] S. M. Pappada, M. H. Owais, B. D. Cameron, J. C. Jaume, A. Mavarez-Martinez, R. S. Tripathi, and T. J. Papadimos, 'An artificial neural network-based predictive model to support optimization of inpatient glycemic control', *Diabetes Technology & Therapeutics*, **22**, 1–12, (2020).

[12] I. Rodriguez-Rodriguez, J.V. Rodriguez, I. Chatzigiannakis, and M.A. Zamora, 'On the possibility of predicting glycaemia 'on the fly' with constrained iot devices in type 1 diabetes mellitus patients', *Sensors*, **19**, 4482–4496, (2019).

[13] I. Rodríguez-Rodríguez, I. Chatzigiannakis, J.V. Rodríguez, M. Maranghi, M. Gentili, and M.A. Zamora, 'Utility of big data in predicting short-term blood glucose levels in type 1 diabetes mellitus through machine learning techniques', *Sensors*, **19**, 4538–4557, (2019).

[14] E. A. Wan and R. V. D. Merwe. The unscented kalman filter for nonlinear estimation. Adaptive Systems for Signal Processing, Communications, and Control Symposium, 2000.

[15] J. Xie and Q. Wang. Benchmark machine learning approaches with classical time series approaches on the blood glucose level prediction challenge. Paper presented at the Third International Workshop on Knowledge Discovery in Healthcare Data at the 27th International Joint Conference on Artificial Intelligence and the 23rd European Conference on Artificial Intelligence, 2018.

Table 3. Clarke Error Grid percentages

	30 min		60 min	
Zone	540	584	540	584
Zones A-B	0.96	0.99	0.935	0.97
Zone C	0.0	0.00	0.001	0.01
Zones D-E	0.04	0.01	0.064	0.02

Table 4. Lasso Significant Values Totals

Feature	Number Recorded	Weights	Adjusted Weights
glucose_level	77563	15.4654	1.2062
basis gsr	39542	0.2272	0.03560
skin temperature	39540	0.2418	0.0295
acceleration	39542	0	0
finger stick	1669	0.54	2.4504
basal	428	0	0
temp basal	208	0	0
bolus	1994	9.4944	23.4776
meal	957	3.5682	31.6974
stressors	2	0	0
exercise	65	0.2312	36.2337

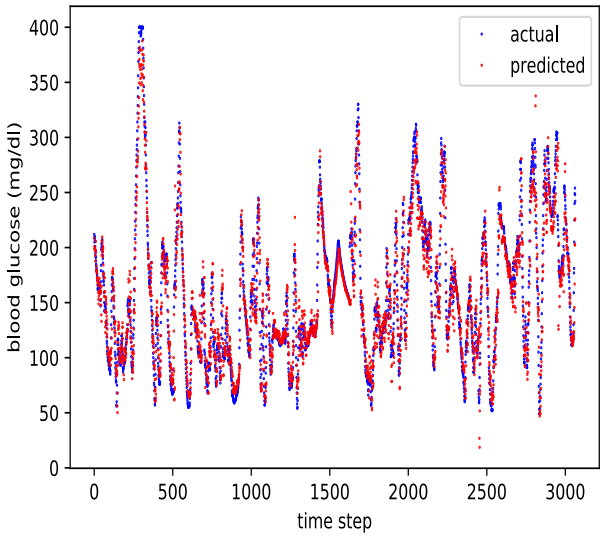


Figure 1. Patient 540 prediction results for 30 min PH with Lasso regression

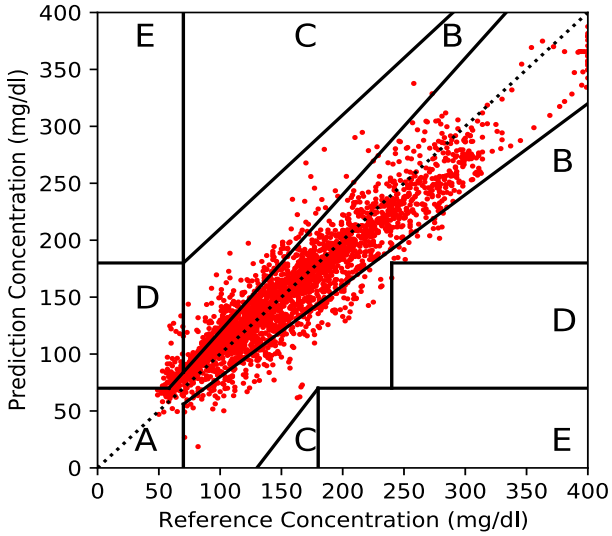


Figure 2. Patient 540 Clarke Error Grid for 30 min PH with Lasso regression

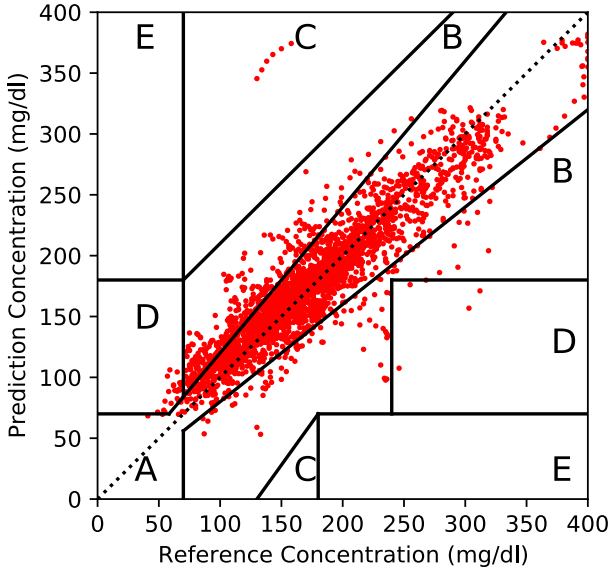


Figure 3. Patient 584 Clarke Error Grid for 30 min PH with Lasso regression

Uncertainty Quantification in Chest X-Ray Image Classification using Bayesian Deep Neural Networks

Yumin Liu[1] and **Claire Zhao**[2] and **Jonathan Rubin**[3]

Abstract. Deep neural networks (DNNs) have proven their effectiveness on numerous tasks. However, research into the reliability of DNNs falls behind their successful applications and remains to be further investigated. In addition to prediction, it is also important to evaluate how confident a DNN is about its predictions, especially when those predictions are being used within medical applications. In this paper, we quantify the uncertainty of DNNs for the task of Chest X-Ray (CXR) image classification. We investigate uncertainties of several commonly used DNN architectures including ResNet, ResNeXt, DenseNet and SENet. We then propose an uncertainty-based evaluation strategy that retains subsets of held-out test data ordered via uncertainty quantification. We analyze the impact of this strategy on the classifier performance. In addition, we also examine the impact of setting uncertainty thresholds on the performance. Results show that utilizing uncertainty information may improve DNN performance for some metrics and observations.

1 INTRODUCTION

Neural networks have been very successful in many fields such as natural language processing [41, 23], computer vision [18, 8], speech recognition [15, 5], machine translation [6], control system [36], auto driving [4] and so on. However, there is much less research available on how reliable neural network predictions are. A common criticism of neural networks is that they are a black box that can perform very well for many tasks, yet lacking interpretability. On the other hand, it is very important to ensure the reliability of a system involved in high risk fields, including stock-market analysis, self-driving cars and medical imaging [28]. As the rapid development of machine learning and artificial intelligence especially deep learning, they are getting more and more applications in health areas including disease diagnosis [9, 10], drug discovery [25, 30] and medical imaging [7, 16, 33]. Rather than just being told a final result by an machine learning algorithm, shareholders (doctors, physicians, radiologists, etc) would like to know how "confident" a neural network model is, so that they can take different actions according to different confidence levels. For example, in a medical image classification scenario, a neural network model is applied to detect whether a patient has a certain type of lung pathology by classifying his/her chest X-ray images. An ideal situation would be that physicians can trust the result of the neural network, if it is highly confident (low uncertainty) about its prediction. On the contrary, if the neural network gives a prediction with low confidence (or high uncertainty), then the prediction could not be trusted and the patient's scan should be further examined by a radiologist. Applying this mechanism is beneficial since there are lots of X-ray images everyday but there are limited radiologist resources. It can help prioritize X-ray images for radiologists to examine, require more attention to low confidence instances and support treatment recommendations for highly confident instances.

Neural network-based deep learning algorithms are also getting popular for medical X-ray image processing [27, 1, 35]. It is necessary to examine the uncertainty of neural network models in medical X-ray image processing. The confidence of a prediction by a machine learning method can be measured by the uncertainty of the method outputs. A typical way to estimate uncertainty is through Bayesian learning [2], which regards the parameters of methods as random variables and attempts to get the posterior distribution of the parameters during training while marginalizing out the parameters to get the distribution of the prediction during inference. Bayesian learning is well developed in traditional non-neural network machine learning framework [2]

2 RELATED WORKS

In recent years Bayesian learning and estimation of prediction uncertainty have gained more and more attention in neural networks context due to the wide application of deep neural networks in many areas [11, 3, 12, 13, 22, 14, 32, 24, 40, 26, 12, 31, 32].

The authors in [3] introduced a method called "Bayes By Backprop" to learn the posterior distribution on the weights of neural networks and get weight uncertainty. Essentially this method assumes the weights come from a multivariate Gaussian distribution and updates the mean and covariance of the Gaussian instead of the weight samples during training. During inference the network weights are drawn from the learned distribution. This method is mathematically grounded, backpropagation-compatible and can learn the distribution of network weights directly, but it cannot utilize pre-trained model and has to build the corresponding model for every neural network architecture. [13] reformulated dropout in neural networks as approximate Bayesian inference in deep Gaussian processes and thus can estimate uncertainty in neural networks with dropout layers. This method requires dropout layers applied before every weight layer. During inference, the dropout layers with random 0-1s drawn from Bernoulli distribution mask out some weights and only use a subset of the weights learned during training phase to make a prediction. In [22], the authors further proposed that there are two types of uncertainties and they showed the benefits of explicitly formulating these two uncertainties separately. The first type is called *aleatoric* uncertainty (or data uncertainty), which is due to the noise in the data and cannot be eliminated, while the other type is called *epistemic* uncer-

[1] Northeastern University, USA, email: yuminliu@ece.neu.edu
[2] Philips Research North America, USA, email: claire.zhao@philips.com
[3] Philips Research North America, USA, email: jonathan.rubin@philips.com

tainty (or model uncertainty), which accounts for uncertainty in the model and can be eliminated given enough data. The network architectures have to be modified to add extra outputs in order to model these uncertainties. [24] adopted this typing of uncertainty, but modified the formulation of aleatoric and epistemic uncertainty to avoid the requirement of extra outputs.

[26] proposed a method called "Stochastic Weight Averaging Gaussian (SWAG)" to approximate the posterior distribution over the weights of neural networks as a Gaussian distribution by utilizing information in Stochastic Gradient Descent (SGD). This method has an advantage in that it can be applied to almost all existing neural networks without modifying their original architectures and can directly leverage pre-trained models. [34] also decomposed predictive uncertainty in deep learning into two components and modeled them separately. They shown that quantifying the uncertainty can help to improve the predictive performance in medical image super resolution. [39] investigated the relationship between uncertain labels in CheXpert [21] and Chest X-ray14 [37] data sets and the estimated uncertainty for corresponding instances using Bayesian neural network and suggested that utilizing uncertain labels helped prevent over-confident for ambiguous instances.

Despite the above works in Bayesian deep neural network learning and uncertainty quantification, there are few works on evaluating the effects of uncertainty-based evaluation strategies for medical image classification. To the best of our knowledge, we are the first to apply uncertainty quantification strategies for chest X-ray image classification using deep neural networks and evaluate their impacts on performances. The main contributions of this paper are:

- We apply uncertainty quantification to five deep neural network models for chest X-ray image classification and analyze their performances.
- We investigate the impact that uncertainty information has on classification task performance by evaluating subsets of held-out test data ordered via uncertainty quantification.

3 METHOD

In this section, we will introduce the basic ideas of Bayesian Neural Networks and one of its approximations – SWAG [26], which is used in this paper. We also describe the uncertainty quantification method used in this paper.

3.1 Bayesian Neural Network

In the ordinary deterministic neural networks, we get point estimation of the network weights $\boldsymbol{w}$ which are regarded as fixed values and will not be changed after training. During inference, for each input $\boldsymbol{x}_i$ we get one deterministic prediction $p(y_i|\boldsymbol{x}_i) = p(y_i|\boldsymbol{x}_i, \boldsymbol{w})$ without getting the uncertainty information.

In the Bayesian neural network settings, in addition to the target prediction, we also want to get the uncertainty for the prediction. To do so we regard the neural network weights as random variables that subject to some form of distribution and try to estimate the posterior distribution of the network weights given the training data during training. We then integrated out the weights and get the distribution over the prediction during inference. From the prediction distribution we can further calculate the prediction output and corresponding uncertainty. More specifically, let $D = \{(\boldsymbol{X}, \boldsymbol{Y})\}$ and $\boldsymbol{w}$ be the training data and weights of a neural network, respectively. The ordinary deterministic neural network methods try to get a point estimate of $\boldsymbol{w}$ by either maximum likelihood estimator (MLE) $\boldsymbol{w}^* = \arg\max_{\boldsymbol{w}} p(D|\boldsymbol{w})$ or maximum a posterior (MAP): $\boldsymbol{w}^* = \arg\max_{\boldsymbol{w}} p(\boldsymbol{w}|D)$ where $p(\boldsymbol{w}|D) = \frac{p(\boldsymbol{w})p(D|\boldsymbol{w})}{p(D)} \propto p(\boldsymbol{w})p(D|\boldsymbol{w})$. The $\boldsymbol{w}^*$ are fixed after training and used for inference for the new data. In Bayesian learning, we estimate the posterior distribution $p(\boldsymbol{w}|D)$ during training and marginalize out $\boldsymbol{w}$ during the inference to get a probability distribution of the prediction.

$$p(\boldsymbol{y}|\boldsymbol{x}, D) = \mathbb{E}_{\boldsymbol{w}\sim p(\boldsymbol{w}|D)}[p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{w})] = \int p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{w})p(\boldsymbol{w}|D)d\boldsymbol{w} \quad (1)$$

After getting the $p(\boldsymbol{y}|\boldsymbol{x})$, we can calculate the statistical moments of the predicted variable and regard the first and second moment (i.e., mean and variance) as the prediction and uncertainty, respectively.

However, in practice there are two major difficulties. The first one is that $p(D) = \int p(\boldsymbol{w})p(D|\boldsymbol{w})d\boldsymbol{w}$ is usually intractable and thus we cannot get exact $p(\boldsymbol{w}|D)$. The second lies in that Eq. (1) is also usually intractable for neural networks. One common approach to deal with the first difficulty is to use a simpler form of distribution $q(\boldsymbol{w}|\theta)$ with hyperparameters θ to approximate $p(\boldsymbol{w}|D)$ by minimizing the Kullback-Leibler (KL) divergence between $q(\boldsymbol{w}|\theta)$ and $p(\boldsymbol{w}|D)$. This turns the problem into an easier optimization problem:

$$\begin{aligned}\theta^* &= \arg\min_{\theta} KL[q(\boldsymbol{w}|\theta)||p(\boldsymbol{w}|D)] \\ &= \arg\min_{\theta} \int q(\boldsymbol{w}|\theta) log \frac{q(\boldsymbol{w}|\theta)}{p(\boldsymbol{w}|D)} d\boldsymbol{w}\end{aligned} \quad (2)$$

For the second difficulty, the usual approach is to use sampling to estimate Eq. (1), and it becomes

$$p(\boldsymbol{y}|\boldsymbol{x}) \approx \mathbb{E}_{\boldsymbol{w}\sim q(\boldsymbol{w}|\theta^*)}[p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{w})] \approx \frac{1}{T}\sum_{i=1}^{T} p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{w}^{(i)}) \quad (3)$$

where $\boldsymbol{w}^{(i)} \sim q(\boldsymbol{w}|\theta^*)$.

People had proposed different methods to approximate the posterior $p(\boldsymbol{w}|\theta)$ or to get the samples of $\boldsymbol{w}$ [26, 3, 12, 13].

3.2 Stochastic Weight Averaging Gaussian (SWAG)

The basic idea of SWAG [26] is to regard the weights of the neural networks as random variables and get their statistical moments through training with SGD. Then use these moments to fit a multivariate Gaussian to get the posterior distribution of the weights. After the original training process in which we get the optimal weights, we continue to train the model using the same training data with SGD and get T samples of the weights $\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_t, \cdots, \boldsymbol{w}_T$. The mean of those samples is $\overline{\boldsymbol{w}} = \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{w}_t$. The mean of the square is $\overline{\boldsymbol{w}^2} = \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{w}_t^2$ and we define a diagonal matrix $\boldsymbol{\Sigma}_{diag} = diag(\overline{\boldsymbol{w}^2} - \overline{\boldsymbol{w}}^2)$ and a deviation matrix $\boldsymbol{R} = [\boldsymbol{R}_1, \cdots, \boldsymbol{R}_t, \cdots, \boldsymbol{R}_T]$ whose columns $\boldsymbol{R}_t = \boldsymbol{w}_t - \overline{\boldsymbol{w}}_t$, where $\overline{\boldsymbol{w}}_t$ is the running average of the first t weights samples $\overline{\boldsymbol{w}}_t = \frac{1}{t}\sum_{j=1}^{t} \boldsymbol{w}_j$. In the original paper, the authors used the last K columns of $\boldsymbol{R}$ to get the low rank approximation of $\boldsymbol{R}$. The K-rank approximation is $\widehat{\boldsymbol{R}} = [\boldsymbol{R}_{T-K+1}, \cdots, \boldsymbol{R}_T]$. Then the mean and covariance matrix for the fitted Gaussian are given by:

$$\boldsymbol{w}_{SWA} = \overline{\boldsymbol{w}} \quad (4)$$

$$\boldsymbol{\Sigma}_{SWA} = \frac{1}{2}\boldsymbol{\Sigma}_{diag} + \frac{1}{2(K-1)}\widehat{\boldsymbol{R}}\widehat{\boldsymbol{R}}^T \quad (5)$$

During inference, for each input (image) $\boldsymbol{x}_i$, sample the weights from the Gaussian $\boldsymbol{w}_s \sim N(\boldsymbol{w}_{SWA}, \boldsymbol{\Sigma}_{SWA})$ then update the batch norm statistics by performing one epoch of forward pass, and then

the sample prediction is given by $p(\hat{y}_{is}|\boldsymbol{x}_i) = p(y_i|\boldsymbol{x}_i, \boldsymbol{w}_s)$. Repeat the precedure for S times and we get S predictions $\hat{y}_{i1}, \hat{y}_{i2}, \cdots, \hat{y}_{is}, \cdots, \hat{y}_{iS}$ for the same input $\boldsymbol{x}_i$. By using these S predictions we can get the final prediction and uncertainty. For regression problem, the final prediction will be $\hat{y}_i = \frac{1}{S}\sum_{s=1}^{S} \hat{y}_{is}$.

3.3 Uncertainty Quantification

Some methods had been proposed to quantify the uncertainty in classification [24, 22]. Here we adopt the method proposed by [24] since it does not require extra output and does not need to modify the network architectures.

For a classification problem, suppose there are C classes, denote $\boldsymbol{p}_s \triangleq [p_{s1}, p_{s2}, \cdots, p_{sc}] = p(\boldsymbol{y}|\boldsymbol{x}, \theta_s)$, $s \in \{1, 2, \cdots, S\}$ as the softmax (or sigmoid in binary case if $C = 2$) output of the neural network for a same repeated input $\boldsymbol{x}$ for S times, then the predicted "probability" is the average of those S sample outputs $\overline{\boldsymbol{p}} = \frac{1}{S}\sum_{s=1}^{S} \boldsymbol{p}_s$ The predicted class label index is $\hat{\boldsymbol{y}} = \arg\max_c \overline{\boldsymbol{p}}$. The aleatoric uncertainty U_a and the epistemic uncertainty U_e are $U_a = \frac{1}{S}\sum_{s=1}^{S}[diag(\boldsymbol{p}_s) - \boldsymbol{p}_s\boldsymbol{p}_s^T], U_e = \frac{1}{S}\sum_{s=1}^{S}(\boldsymbol{p}_s - \overline{\boldsymbol{p}})(\boldsymbol{p}_s - \overline{\boldsymbol{p}})^T$ The total uncertainty is $U_{total} = U_a + U_e$. For binary classification, the sigmoid output is a scalar and the uncertainty equations are reduced to

$$U_a = \frac{1}{S}\sum_{s=1}^{S} p_s(1 - p_s) \tag{6}$$

$$U_e = \frac{1}{S}\sum_{s=1}^{S}(p_s - \overline{p})^2 \tag{7}$$

where $\overline{p} = \frac{1}{S}\sum_{s=1}^{S} p_s$ and $p_s = p(y = 1|\boldsymbol{x}, \theta_s) = 1 - p(y = 0|\boldsymbol{x}, \theta_s)$. The predicted label is:

$$\hat{y} = \begin{cases} 1 & \overline{p} \geq 0.5 \\ 0 & \overline{p} < 0.5 \end{cases} \tag{8}$$

In this way, we can get uncertainties for all the instances.

3.4 Transfer Learning

Transfer learning is a widely used technique to help improve performance for deep neural networks in image classification. Here we can also benefit from transfer learning by loading pre-trained neural network models trained by ImageNet (`http://image-net.org`) dataset. The SWAG method has one advantageous characteristic that it does not require to modify any architecture of the original neural networks and therefore we can fully utilize pre-trained models trained by ImageNet dataset to speed up training process and get better predictions. In the initialization stage, we download the pre-trained model parameters and use them to initialize our models to be trained.

3.5 Procedure

Basically we follow the method in [26] to approximate the Bayesian neural network and the formulas in [24] to quantify uncertainty of the models. The overall algorithm for SWAG and uncertainty quantification is shown in Algorithm 1. We initialize the model with corresponding pre-trained model, and then fine-tune it by training using chest X-ray images and observation labels. After that we perform SWAG algorithm by continuing training using Stochastic Gradient Descent for T epochs and calculate statistics $\overline{\boldsymbol{w}}$, $\overline{\boldsymbol{w}^2}$, $\boldsymbol{\Sigma}_{diag}$ and $\widehat{\boldsymbol{R}}$,

Algorithm 1 Uncertainty Quantification

1: **Input:**
$D = \{(\boldsymbol{X}, \boldsymbol{Y})\}$ / $\boldsymbol{X}_i$: training / evaluating chest X-ray images and corresponding observation labels
2: **Initialization:**
load pre-trained neural network (NN) models by ImageNet
3: **Training:**
Fine-tune NN models using cheXpert dataset
4: **Perform SWAG:**
Continue training with SGD
i) train NN models using SGD for some epochs with D
ii) save statistics of the weights for those epochs
iii) calculate $\boldsymbol{w}_{SWA}$ and $\boldsymbol{\Sigma}_{SWA}$ using Eq. 4 and 5
vi) fit a Gaussian using $\boldsymbol{w}_{SWA}$ as mean and $\boldsymbol{\Sigma}_{SWA}$ as covariance
Prediction
for s from 1 to S
draw weights $\boldsymbol{w}_s \sim N(\boldsymbol{w}_{SWA}|\boldsymbol{\Sigma}_{SWA})$
update batch norm statistics using D
$p(y_{is}|\boldsymbol{X}_i) = p(y_{is}|\boldsymbol{X}_i, \boldsymbol{w}_s)$
end for
5: **Calculate Outputs:**
$\overline{p}(y_i|\boldsymbol{X}_i) = \frac{1}{S}\sum_{s=1}^{S} p(y_{is}|\boldsymbol{X}_i)$
Calculate $\hat{y}_i$, U_a and U_e using Eq. (8), (6) and (7).
$U_{total} = U_a + U_e$
6: **Return:**
$\hat{y}_i, U_a, U_e, U_{total}$

from which we can get $\boldsymbol{w}_{SWA}$ and $\boldsymbol{\Sigma}_{SWA}$ using Eq. 4 and 5. Then we fit a multivariate Gaussian using $\boldsymbol{w}_{SWA}$ as mean and $\boldsymbol{\Sigma}_{SWA}$ as covariance and get an approximated distribution for the neural network weights. When doing a prediction, an input chest X-ray image is repeatedly fed into the network for S times, each time with a new set of weights sampled from the Gaussian distribution. The S output probabilities are used to calculate the final predicted label $\hat{y}_i$ and uncertainty $U_{total} = U_a + U_e$. It is worthwhile to note that, after drawing sample weights the network batch norm statistics need to be updated for the models that use batch normalization. It can be achieved by running one epoch with partial or full training set D. More detailed justification for the necessity was given in the original paper [26].

4 DATASET

We perform experiments using the CheXpert data set [21]. CheXpert is a large chest X-ray dataset released by researchers at Stanford University. This dataset consists of 224,316 chest radiographs of 65,240 patients. Each data instance contains a chest X-ray image and a vector label describing the presence of 14 observations (pathologies) as positive, negative, or uncertain. The labels were extracted from radiology reports using natural language processing approaches. For our experiments we focus on 5 observations, namely Cardiomegaly, Edema, Atelectasis, Consolidation and Pleural Effusion. As [21] had pointed out, these 5 observations were selected based on their clinical importance and prevalence in this dataset. In their experiment they also used these 5 observations to evaluate the labeling approaches. A sample image for each observation is shown in Figure 1.

The original dataset consists of training set and validation set and we do not have access to test set. The labels for the training set were generated by automated rule-based labeler which extract informa-

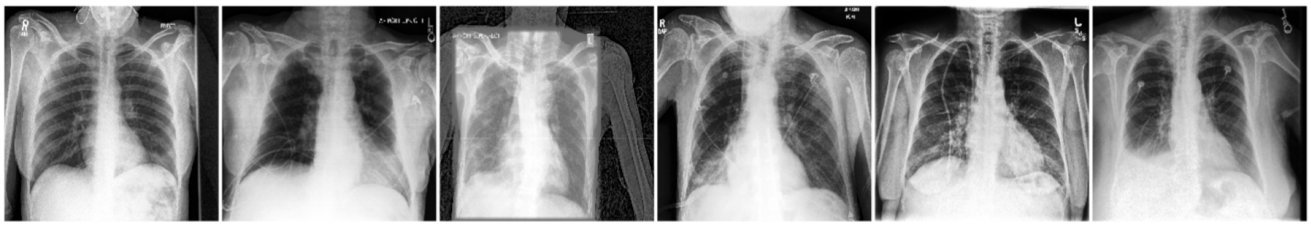
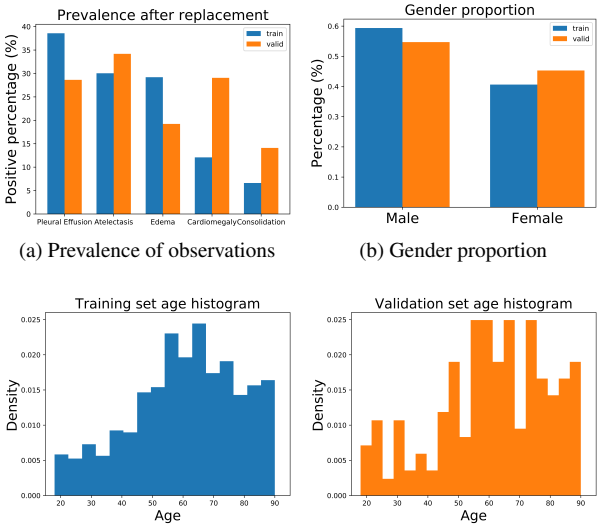


Figure 1: Sample image for each observation. From left to right: no finding (all negative), cardiomegaly, edema, consolidation, atelectasis and pleural effusion

tion from radiology reports. This was done by the Stanford research group who released the dataset. There are three possible values for the label of an instance for a given observation, i.e., 1, 0 and -1. 1 means the observation is positive (or exists), 0 means negative (or not exists), and -1 means not certain about whether the observation exists. The labels for the validation set were determined by the majority vote from three board-certified radiologists and only contains positive (1) or negative (0) values. The original paper [21] investigated several different ways to deal with the uncertain labels (-1), such as regarding them as positive (1), negative (0), the same with the majority class, or a separate class. They found out that for different observations, the optimal ways to deal with the uncertain labels are different, and they gave the replacement for 5 observations mentioned above. Based on the results from [21] and for simplicity, we replace the uncertain labels with 0 or 1 for different observations.

Specifically, the uncertain labels of cardiomegaly, consolidation and pleural effusion are replaced with 0, while edema and atelectasis with 1. Therefore the problem becomes a multi-label binary image classification problem. The predicted result is a five dimensional vector with element value being 1 or 0, where 1 means that the network predicts existence for the corresponding observation while 0 means the network predicts not existence of the corresponding observation. We follow the official training set / validation set split given by the data set provider. After removing invalid instances, we get a total number of 223,414 instances for training and 234 instances for validation. We first initialize the neural network's parameters with corresponding downloaded pre-trained model parameters, and then train the neural network using the training set and test their performance on the validation set. We will use the original training set as the training set and original validation set as the evaluation set in our experiments.

In Figure 2 we show the patient statistics of the 5 observations after replacing the uncertain labels in the training set. The prevalence is the ratio of the number of positive instances over the total number of instances. From the figure we can see that all five observations are imbalance as the prevalence being under 50%. Besides, there is a gap in the prevalence for the training and evaluation sets in all observations, which will probably affect the performance of the neural network models.



(a) Prevalence of observations (b) Gender proportion

(c) Training set age histogram (d) Validation set age histogram

Figure 2: Patient statistics

5 EXPERIMENT

In this section, we perform experiments and present the investigation results of uncertainty quantification and strategy on five different neural network models using *PyTorch* implementation. These neural networks are DenseNet [20] with 121 layers (denote as *DenseNet121*), DenseNet with 201 layers (denote as *DenseNet201*), ResNet [17] with 152 layers (denote as *ResNet152*), ResNeXt [38] with 101 layers (denote as *ResNeXt101*) and Squeeze-and-Excitation network [19] with 154 layers (denote as *SENet154*). ResNet uses skip connections to mitigrate the gradient vanishment problem and was the winner of ILSVRC 2015 [29] and COCO 2015 (`http://cocodataset.org`) competition. ResNeXt is a variant of ResNet and won the 2nd place in ILSVRC 2016 classification task. DenseNet further utilizes the concept of skip connections by connecting previous layer output to all its subsequent layers and forming "dense" skip connections. DenseNet further alleviates vanishing gradient problem, reduce number of parameters and reuses intermediate features, and is widely used since it was proposed. SENet uses squeeze-and-excitation block to model image channel interdependencies and won the ILSVRC 2017 competition for classification task.

All networks are trained as binary classifiers for multi-label classification instead of training separate models for each class.

The pipeline of the experiment is shown in Figure 4. We use PyTorch implementation. The neural network models and pre-trained parameters are from torchvision (except SENet154 which is from *pretrainedmodels*, `https://github.com/Cadene/pretrained-models.pytorch`).

In our experiment we set the number of sample weights $T = 5$, the number of columns of the deviation matrix $K = 10$ and the number of repeated prediction samples $S = 10$. During training, we use Adam optimizer with weight decay regularizer and *ReduceLROnPlateau* learning rate scheduler. The the initial learning rate is 1×10^{-5} and weight decay coefficient is 0.005. The maximum number of fine-tuning epoch is 50 epochs. The original chest X-ray images are resized and randomly cropped to 256×256 (except for SENet154 which has a fixed input size 224×224). We stop fine-

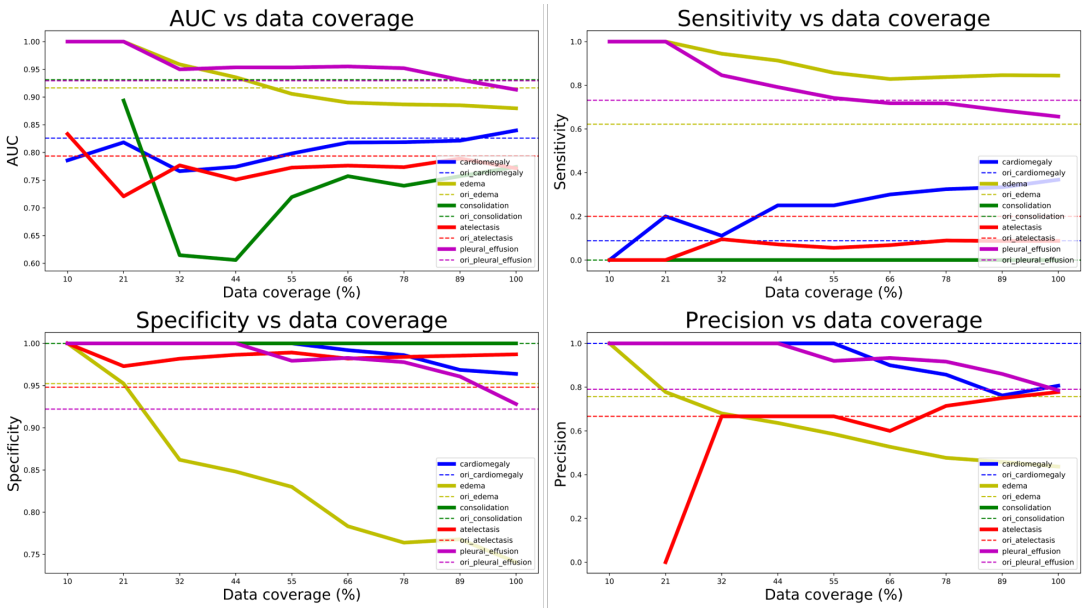


Figure 3: Comparison of performance between original deterministic network and Bayesian neural network with uncertainty strategy. The neural network is DenseNet with 201 layers.

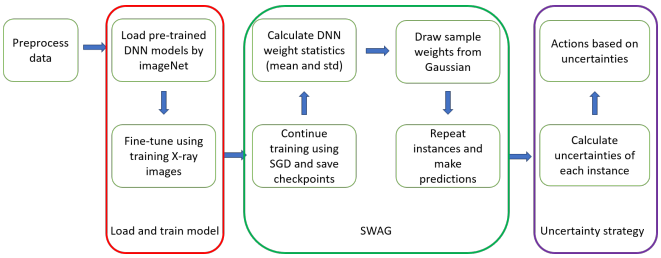


Figure 4: Pipeline of the experiment

tuning the model when the AUC (explained below) does not increase for consecutive 10 epochs and save the model with the best AUC as the optimal trained model.

We use four metrics to evaluate the network classification performance: Area under curve (AUC), Sensitivity, Specifity and Precision. Those metrics are widely used for machine learning and medicine community. The AUC is often used to measure the quality of a classifier and is defined as the area under the Receiver Operating Characteristic (ROC) curve which plots the sensitivity against the false positive rate. The sensitivity (or true positive rate or recall) is defined as the ratio of the number of correctly predicted positive instances over the number of total positive instances. The specificity is defined as the ratio of the number of correctly predicted negative instances over the total number of negative instances. And the precision is defined as the ratio of the number of correctly predicted positive instances over the number of instances that are predicted as positive.

5.1 Without Strategy

First we compare the AUC of the original ordinary deterministic neural networks with the AUC corresponding neural networks after performing SWAG but before applying any uncertainty strategies. The results are shown in Table 1. The "Average" column is the average over all 5 observations. The bold font indicates better performance. For edema and pleural effusion, the original neural network performs better than SWAG for most of the networks. For cardimegaly, consolidation and atelectasis, the performances are mixed. This maybe because edema and pleural effusion are harder to detect and more sensitive to network weights perturbation. On the whole the SWAG algorithm does not outperform the original neural network. These might be accountable because SWAG uses a Gaussian to approximate the distribution over the optimal weights and then draws sample weights from the approximated Gaussian distribution, and may deviate from the optimal weights if the approximation is inaccurate. Therefore we need to adopt some strategy to prevent the performance from deterioration. The benefit lies in that we can get the uncertainty estimation for each prediction while keeping similar or even better prediction results.

Table 1: Original AUC vs SWAG AUC

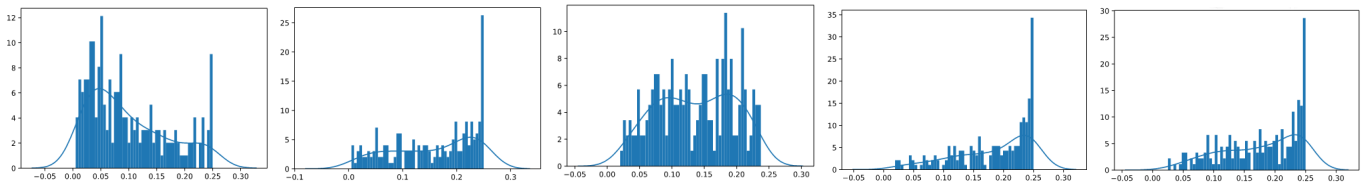
AUC / Networks	Average		Cardiomegaly		Edema		Consolidation		Atelectasis		Pleural Effusion	
	Original	SWAG	Original	SWAG	Original	SWAG	Original	SWAG	Original	SWAG	Original	SWAG
Resnet152	**0.8831**	0.8786	**0.8376**	0.8149	**0.9123**	0.8713	0.8927	**0.9234**	**0.8543**	0.8537	0.9184	**0.9298**
ResNext101	**0.8807**	0.8726	0.8013	**0.8339**	**0.9212**	0.8748	0.9250	**0.9311**	**0.8246**	0.8162	**0.9314**	0.9071
SEnet154	**0.8794**	0.8695	**0.8203**	0.8040	**0.9195**	0.8702	**0.9216**	0.9187	0.8056	**0.8553**	**0.9301**	0.8992
Densenet121	0.8842	**0.8942**	0.8436	**0.8752**	**0.9264**	0.8940	0.9139	**0.9512**	0.8153	**0.8489**	**0.9220**	0.9016
Densenet201	**0.8793**	0.8356	0.8259	**0.8397**	**0.9165**	0.8796	**0.9313**	0.7739	**0.7936**	0.7714	**0.9294**	0.9132

5.2 With Coverage Strategy

Next we utilize the uncertainty quantification information to determine if the performances can be improved. One strategy is to sort instances according to uncertainty in an ascending order, and then take those instances with less uncertainty into consideration and discard the rest. In clinical practice, the discarded instances could be flagged for further evaluation by a physician.

Ideally we would expect a decreasing trend for the metrics when data coverage increase as shown in Figure 6. The horizontal axis "Data coverage" is the percentage of instances being considered. For example, a data coverage of 20% means that only the top twenty percent of the least uncertain (or the most confident) instances are taken into consideration and the rest are discarded.

Figure 3 shows the comparison of performances with regard to the foure metrics (AUC, sensitivity, specificity and precision) between



(a) Cardiomegaly (b) Edema (c) Consolidation (d) Atelectasis (e) Pleural effusion

Figure 5: Estimated total uncertainty (aleatoric + epistemic) histogram for each observation

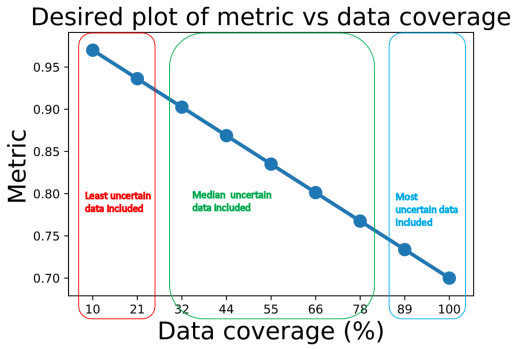


Figure 6: Expected ideal performance. The metric decreases as data coverage increases.

the original deterministic networks and Bayesian neural networks with uncertainty strategy. The solid lines are the Bayesian neural network with uncertainty strategy, while the dashed lines are the original ordinary deterministic networks without any uncertainty strategy. Different colors represent different observations.

From Figure 3 we can see that for edema and pleural effusion, the AUC decreases as the coverage increases, and are above the corresponding original AUC until around 45% and 90% coverage, respectively. This means that applying the uncertainty strategy can improve AUC for these two observations. The highest AUC gain can be 8% and 6% for edema and pleural effusion, respectively. We also observe similar trend in sensitivity, specificity and precision for both edema and pleural effusion. Three observations (cardiomegaly, atelectasis and consolidation) have low sensitivity as most of the predictions are negative. On the contrary the specificity is high.

The highest gains for applying the uncertainty strategy are shown in the Table 2. The effect of the uncertainty strategy over the five observations with the model DenseNet201 can be summarized as in the Table 3. The symbols $\surd$, $\times$, $\circ$ and $-$ represents *helpful*, *not helpful*, *mixed behavior* and *missing value*, respectively. For edema and pleural effusion, applying uncertainty strategy is beneficial for improving all four metrics. However, for other observations, it does not show benefits or only limited benefits for some metrics. The reason why it show varied behavior may be interesting and needs further investigation. Similarly, we summarize the effect of applying uncertainty strategy for different neural network architectures and the results are shown in Table 4 to Table 7. From the tables we can see that applying uncertainty strategy will help to improve some performance metrics for all four neural network models.

Table 2: Perfomance gain for edema and pleural effusion. The values are the absolute and relative gains

Gain (% Gain)	AUC	Sensitivity	Specificity	Precision
Edema	0.0835(9.11%)	0.3778(60.71%)	0.0476(5.00%)	0.2432(32.14%)
Pleural effusion	0.0706(7.60%)	0.2687(36.73%)	0.0778(8.44%)	0.2097(26.53%)

Table 3: Effect of uncertainty strategy for DenseNet201.

Densenet201	AUC	Sens.	Spec.	Prec.
Cardiomegaly	$\times$	$\times$	$\surd$	$\circ$
Edema	$\surd$	$\surd$	$\surd$	$\surd$
Consolidation	$\times$	$\times$	$\circ$	-
Atelectasis	$\circ$	$\times$	$\circ$	$\times$
Pleural effusion	$\surd$	$\surd$	$\surd$	$\surd$

$\surd$: helpful; $\times$: not helpful; $\circ$: mixed behavior; -: missing value

Table 4: Effect of uncertainty strategy for different networks

ResNet152	AUC	Sens.	Spec.	Prec.
Cardiomegaly	$\surd$	$\times$	-	-
Edema	$\times$	$\times$	$\surd$	$\surd$
Consolidation	$\times$	$\times$	-	-
Atelectasis	$\times$	$\times$	$\surd$	$\surd$
Pleural effusion	$\surd$	$\surd$	$\surd$	$\surd$

$\surd$: helpful; $\times$: not helpful; $\circ$: mixed behavior; -: missing value

Table 5: Effect of uncertainty strategy for different networks

SENet154	AUC	Sens.	Spec.	Prec.
Cardiomegaly	$\surd$	-	-	-
Edema	$\times$	$\times$	$\surd$	$\times$
Consolidation	$\surd$	-	-	-
Atelectasis	$\circ$	-	$\times$	-
Pleural effusion	$\surd$	$\times$	$\surd$	$\surd$

$\surd$: helpful; $\times$: not helpful; $\circ$: mixed behavior; -: missing value

Despite that for some observations (e.g., pleural effusion), several metrics performance benefit a lot from applying the uncertainty strategy, we should also notice that the strategy does not help to improve performance for some other observations with regard to these metrics, and in some cases even degrade the performance. The reasons behind might be varied and needs more investigation. For example, this may be that the neural network weight distribution approximated by the SWAG algorithm does not capture the true distribution, or even the uncertainty quantification formulas are inappropriate.

5.3 With Absolute Threshold Strategy

We also plot the total uncertainty distribution for each observation, as shown in Figure 5. From the figure we can see that for cardiomegaly, the estimated uncertainty tends to be smaller, while for edema, atelectasis and plueral effusion, the proportion of larger estimated uncertainty is higher. Consolidation has a relatively even distribution for estimated uncertainty. This suggest that edema, atelectasis and

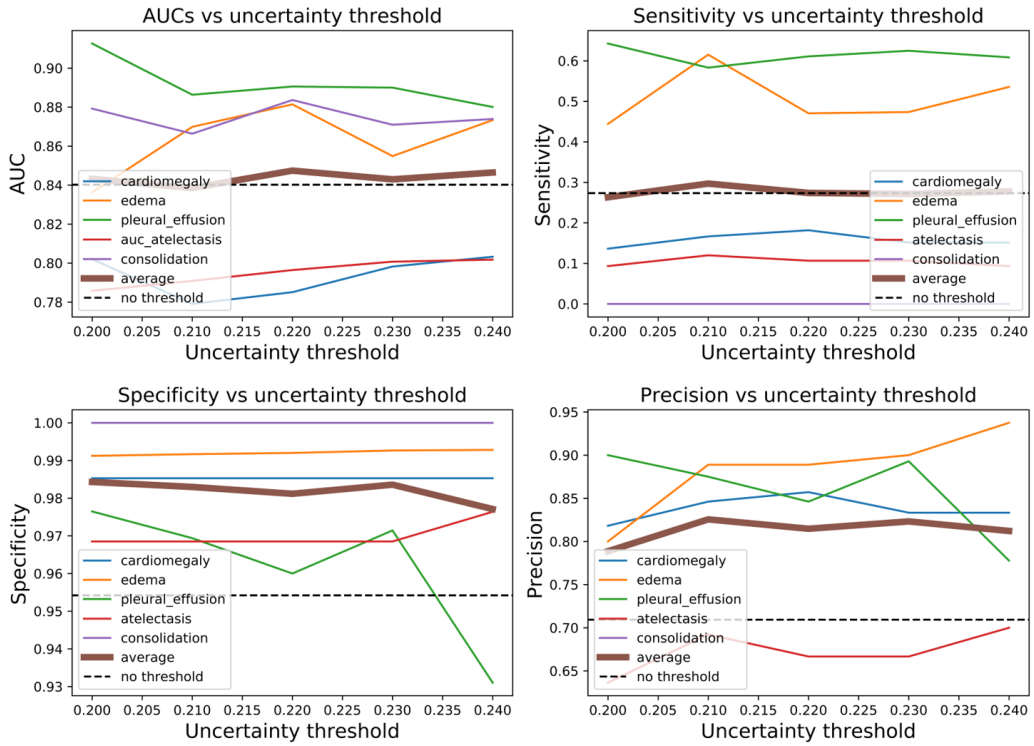


Figure 7: Comparison of performance between original deterministic network and Bayesian neural network with uncertainty threshold.

Table 6: Effect of uncertainty strategy for different networks

ResNext101	AUC	Sens.	Spec.	Prec.
Cardiomegaly	√	×	×	×
Edema	×	×	√	√
Consolidation	√	×	√	-
Atelectasis	○	√	×	×
Pleural effusion	√	√	√	√

√: helpful; ×: not helpful; ○: mixed behavior; -: missing value

Table 7: Effect of uncertainty strategy for different networks

DenseNet121	AUC	Sens.	Spec.	Prec.
Cardiomegaly	×	×	-	○
Edema	×	×	√	√
Consolidation	√	-	-	-
Atelectasis	√	○	√	√
Pleural effusion	√	√	√	√

√: helpful; ×: not helpful; ○: mixed behavior; -: missing value

pleural effusion are more prone to be affected by setting an uncertainty threshold. Combining this finding with the results in Table 2, we set thresholds for both edema and pleural effusion to check the influence on metric performance. We only consider the instances whose estimated uncertainty is smaller than the threshold to compute the performance metrics. We vary the threshold from 0.2 to 0.24 by a step of 0.01 and the results are shown in Figure 7. The black dashed line is the average metric values of the original deterministic neural network, while the solid color thin lines are metric values for each observation, and the thick brown line is the average metric values of all five observation after applying threshold only to edema and pleural effusion. Comparing the thick brown line with the dash black line, we can see that the average specificity and precision have been improved while the average AUC and sensitivity roughly keep the same. This means that applying uncertainty threshold to edema and pleural effusion is beneficial.

6 CONCLUSION

In this paper we investigate uncertainty quantification in medical image classification using Bayesian deep neural networks. We train five different deep neural network models on the CheXpert X-ray image data for five clinical observations and quantify the model uncertainty. Then we analyze the performance of the network for situations with and without applying uncertainty strategy. The results show that the uncertainty quantification and strategy improve several performance metrics for some observations. This suggests that uncertainty quantification is helpful in medical image classification using neural networks. However, the results also show that in some cases the strategy is not helpful, or can even deteriorate the performance. Further analysis may be needed to examine this phenomenon.

REFERENCES

[1] Yaniv Bar, Idit Diamant, Lior Wolf, Sivan Lieberman, Eli Konen, and Hayit Greenspan, 'Chest pathology detection using deep learning with non-medical training', in *2015 IEEE 12th International Symposium on Biomedical Imaging (ISBI)*, pp. 294–297. IEEE, (2015).

[2] Christopher M Bishop, *Pattern recognition and machine learning*, springer, 2006.

[3] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra, 'Weight uncertainty in neural networks', *arXiv preprint arXiv:1505.05424*, (2015).

[4] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al., 'End to end learning for self-driving cars', *arXiv preprint arXiv:1604.07316*, (2016).

[5] William Chan, Navdeep Jaitly, Quoc Le, and Oriol Vinyals, 'Listen, attend and spell: A neural network for large vocabulary conversational speech recognition', in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4960–4964. IEEE, (2016).

[6] Junyoung Chung, Kyunghyun Cho, and Yoshua Bengio, 'A character-level decoder without explicit segmentation for neural machine translation', *arXiv preprint arXiv:1603.06147*, (2016).

[7] Marleen de Bruijne. Machine learning approaches in medical image analysis: From detection to diagnosis, 2016.

[8] Chao Dong, Chen Change Loy, and Xiaoou Tang, 'Accelerating the super-resolution convolutional neural network', in *European conference on computer vision*, pp. 391–407. Springer, (2016).

[9] Meherwar Fatima and Maruf Pasha, 'Survey of machine learning algorithms for disease diagnostic', *Journal of Intelligent Learning Systems and Applications*, **9**(01), 1, (2017).

[10] Konstantinos P Ferentinos, 'Deep learning models for plant disease detection and diagnosis', *Computers and Electronics in Agriculture*, **145**, 311–318, (2018).

[11] Yarin Gal, *Uncertainty in deep learning*, Ph.D. dissertation, PhD thesis, University of Cambridge, 2016.

[12] Yarin Gal and Zoubin Ghahramani, 'Bayesian convolutional neural networks with bernoulli approximate variational inference', *arXiv preprint arXiv:1506.02158*, (2015).

[13] Yarin Gal and Zoubin Ghahramani, 'Dropout as a bayesian approximation: Representing model uncertainty in deep learning', in *international conference on machine learning*, pp. 1050–1059, (2016).

[14] Yarin Gal, Riashat Islam, and Zoubin Ghahramani, 'Deep bayesian active learning with image data', in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1183–1192. JMLR. org, (2017).

[15] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton, 'Speech recognition with deep recurrent neural networks', in *2013 IEEE international conference on acoustics, speech and signal processing*, pp. 6645–6649. IEEE, (2013).

[16] Hayit Greenspan, Bram Van Ginneken, and Ronald M Summers, 'Guest editorial deep learning in medical imaging: Overview and future promise of an exciting new technique', *IEEE Transactions on Medical Imaging*, **35**(5), 1153–1159, (2016).

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, 'Deep residual learning for image recognition', in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, (2016).

[18] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam, 'Mobilenets: Efficient convolutional neural networks for mobile vision applications', *arXiv preprint arXiv:1704.04861*, (2017).

[19] Jie Hu, Li Shen, and Gang Sun, 'Squeeze-and-excitation networks', in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7132–7141, (2018).

[20] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger, 'Densely connected convolutional networks', in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700–4708, (2017).

[21] Jeremy Irvin, Pranav Rajpurkar, Michael Ko, Yifan Yu, Silviana Ciurea-Ilcus, Chris Chute, Henrik Marklund, Behzad Haghgoo, Robyn Ball, Katie Shpanskaya, et al., 'Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison', *arXiv preprint arXiv:1901.07031*, (2019).

[22] Alex Kendall and Yarin Gal, 'What uncertainties do we need in bayesian deep learning for computer vision?', in *Advances in neural information processing systems*, pp. 5574–5584, (2017).

[23] Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher, 'Ask me anything: Dynamic memory networks for natural language processing', in *International conference on machine learning*, pp. 1378–1387, (2016).

[24] Yongchan Kwon, Joong-Ho Won, Beom Joon Kim, and Myunghee Cho Paik, 'Uncertainty quantification using bayesian neural networks in classification: Application to ischemic stroke lesion segmentation', *Medical Imaging with Deep Learning, 2018*, (2018).

[25] Antonio Lavecchia, 'Machine-learning approaches in drug discovery: methods and applications', *Drug discovery today*, **20**(3), 318–331, (2015).

[26] Wesley Maddox, Timur Garipov, Pavel Izmailov, Dmitry Vetrov, and Andrew Gordon Wilson, 'A simple baseline for bayesian uncertainty in deep learning', *arXiv preprint arXiv:1902.02476*, (2019).

[27] Pranav Rajpurkar, Jeremy Irvin, Kaylie Zhu, Brandon Yang, Hershel Mehta, Tony Duan, Daisy Ding, Aarti Bagul, Curtis Langlotz, Katie Shpanskaya, et al., 'Chexnet: Radiologist-level pneumonia detection on chest x-rays with deep learning', *arXiv preprint arXiv:1711.05225*, (2017).

[28] Muhammad Imran Razzak, Saeeda Naz, and Ahmad Zaib, 'Deep learning for medical image processing: Overview, challenges and the future', in *Classification in BioApps*, 323–350, Springer, (2018).

[29] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei, 'ImageNet Large Scale Visual Recognition Challenge', *International Journal of Computer Vision (IJCV)*, **115**(3), 211–252, (2015).

[30] Benjamin Sanchez-Lengeling and Alán Aspuru-Guzik, 'Inverse molecular design using machine learning: Generative models for matter engineering', *Science*, **361**(6400), 360–365, (2018).

[31] Peter Schulam and Suchi Saria, 'Can you trust this prediction? auditing pointwise reliability after learning', in *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 1022–1031, (2019).

[32] Murat Sensoy, Lance Kaplan, and Melih Kandemir, 'Evidential deep learning to quantify classification uncertainty', in *Advances in Neural Information Processing Systems*, pp. 3179–3189, (2018).

[33] Kenji Suzuki, 'Overview of deep learning in medical imaging', *Radiological physics and technology*, **10**(3), 257–273, (2017).

[34] Ryutaro Tanno, Daniel Worrall, Enrico Kaden, Aurobrata Ghosh, Francesco Grussu, Alberto Bizzi, Stamatios N Sotiropoulos, Antonio Criminisi, and Daniel C Alexander, 'Uncertainty quantification in deep learning for safer neuroimage enhancement', *arXiv preprint arXiv:1907.13418*, (2019).

[35] Demetri Terzopoulos et al., 'Semi-supervised multi-task learning with chest x-ray images', in *International Workshop on Machine Learning in Medical Imaging*, pp. 151–159. Springer, (2019).

[36] Huanqing Wang, Peter Xiaoping Liu, Shuai Li, and Ding Wang, 'Adaptive neural output-feedback control for a class of nonlower triangular nonlinear systems with unmodeled dynamics', *IEEE transactions on neural networks and learning systems*, **29**(8), 3658–3668, (2017).

[37] Xiaosong Wang, Yifan Peng, Le Lu, Zhiyong Lu, Mohammadhadi Bagheri, and Ronald M Summers, 'Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases', in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2097–2106, (2017).

[38] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He, 'Aggregated residual transformations for deep neural networks', in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1492–1500, (2017).

[39] Hao-Yu Yang, Junling Yang, Yue Pan, Kunlin Cao, Qi Song, Feng Gao, and Youbing Yin, 'Learn to be uncertain: Leveraging uncertain labels in chest x-rays with bayesian neural networks', in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 5–8, (2019).

[40] Jiayu Yao, Weiwei Pan, Soumya Ghosh, and Finale Doshi-Velez, 'Quality of uncertainty quantification for bayesian neural network inference', *arXiv preprint arXiv:1906.09686*, (2019).

[41] Tom Young, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria, 'Recent trends in deep learning based natural language processing', *ieee Computational intelligenCe magazine*, **13**(3), 55–75, (2018).

Prognosis Prediction in Covid-19 Patients from Lab Tests and X-ray Data through Randomized Decision Trees

Alfonso E. Gerevini*, Roberto Maroldi*†, Matteo Olivato*, Luca Putelli*, Ivan Serina*
***Università degli Studi di Brescia, †ASST Spedali Civili di Brescia**
{alfonso.gerevini, roberto.maroldi, ivan.serina, m.olivato, l.putelli002}@unibs.it

Abstract. AI and Machine Learning can offer powerful tools to help in the fight against Covid-19. In this paper we present a study and a concrete tool based on machine learning to predict the prognosis of hospitalised patients with Covid-19. In particular we address the task of predicting the risk of death of a patient at different times of the hospitalisation, on the base of some demographic information, chest X-ray scores and several laboratory findings. Our machine learning models use ensembles of decision trees trained and tested using data from more than 2000 patients. An experimental evaluation of the models shows good performance in solving the addressed task.

1 Introduction

The fight against Covid-19 is a new important challenge for the world that AI and machine learning can help facing at various levels [15, 28, 29]. In March 2020, at the time of the coronavirus emergency in Italy, we started working in strict collaboration with one of the hospitals that had more Covid-19 patients in Italy, *Spedali Civili di Brescia*, to help predicting the prognosis of hospitalised patients. Our work was focused on the task of predicting the risk of death of a patient at different times of the hospitalisation. As discussed in [28], predicting if a patient is at risk of decease or adverse events can help the hospital, for instance, to organize the allocation of limited health resources in a more efficient way.

Our predictive models are built on the base of demographic information (sex and age), the values of ten laboratory tests and the chest X-ray score(s), which is an innovative measure developed and used at *Spedali Civili di Brescia* to assess the severity of the pulmonary conditions [3]. Other important information, such us the patient comorbidities or the time and duration of the symptoms related to Covid-19, were not used because not available to us.

Using raw data from more than 2000 patients, we built some data sets describing the "clinical history" of each patient during the hospitalisation. In particular, each dataset contains a "snapshot" of the infection conditions of every considered patient at a certain day after the start of the hospitalisation. For each dataset, we built a different predictor, allowing to make progressive predictions over time that take into account the evolution of the disease severity in a patient, which helps the formulation of a personalized prediction of the prognosis. A change of the predicted risk over time for a patient could also hint a link between specific events or treatments and the increase or decrease of the risk for the patient. As snapshot times for a patient, in our experiments we considered the 2nd, 4th, 6th, 8th and 10th hospitalization day, and the day before the end of the hospitalisation.

Our datasets were engineered to cope with a number of practical issues, including missing values and feature values categorization, and to add some helpful artificial features. We also addressed the "concept drift" issue [6, 23], since we observed that the risk of death was clearly sensitive to the time period when the patient was hospitalised; the risk was significantly higher during the earlier period of the emergency (March 2020), when in northern Italy the spread of the virus infection was very high and many people were hospitalised. Moreover, given the very sensitive nature of our task, we introduced a threshold to discharge the model predictions that have a low estimated probability. Such a threshold is a parameter that is automatically calculated and optimised during the training phase.

We considered several machine learning algorithms. A first experimental comparison of their performance on our data sets showed that methods based on forests of trees have more promising performance, and so we decided to focus on this approach. The obtained prediction models have good performance over a randomly chosen test set of 200 patients for each considered period, in terms of both F2 and ROC-AUC scores. In particular, overall the system makes very few errors in predicting patient survival, i.e., the specificity of the prediction is very high.

In the following, after discussing related work, we describe our data sets, we present our prediction models and their experimental evaluation, and finally we give conclusions and mention future work.

2 Related work

Artificial Intelligence and Machine Learning techniques can be used for tackling the Covid-19 pandemic in different aspects. However, given that the pandemic has started only few months ago, most works are still preliminary, and there isn't a clear description of the developed techniques and of their results (often only pre-printed and not properly peer-reviewed).

A preliminary study is presented in [15]. Given a set of only 53 patients with mild symptoms and their lab tests, comorbidities and treatment, the authors train several machine learning models (Logistic Regression, Decision Trees, Random Forests, Support Vector Machines, KNN) to predict if a patient will be subject to more sever symptoms, obtaining a prediction accuracy score of up to 0.8 using 10-fold cross validation. The generalizability and strength of these results are questionable, given the very small set of considered patients.

Another example is the pre-printed work by Li Yan *et al.* [29] that uses lab tests for predicting the mortality risk; the proposed model is a very simple decision tree based on the three most important fea-

tures. While the performance seems promising, the test set used for evaluation was very small (29 patients).

Various AI and machine learning techniques have been developed for prognosis and disease progression prediction [7] in the context of diseases different from Covid-19 [20, 21, 22]. In particular, in the last few years, several works about predicting mortality risk or adverse events and on the use of AI in critical care [19] have been published. The survey in [1] presents a review of statistical and ML systems for predicting the mortality risk, the need of beds in intense care units [30] or the length of the patient hospitalization. In particular, it is worth to mention the work by Harutyunyan et al. [11] which uses LSTM Neural Networks for predicting both the mortality risk and the length of the hospitalisation.

An overview of the issues and challenges for applying ML in a critical-care context is available in [16]. This work stresses the need to deal with corrupted data, like missing values, imprecision, and errors that can increase the complexity of prediction tasks.

Lab test findings and their variation over time are the main focus of the work by Hyland et al. [14], which describes a system that processes these data to generate an alarm predicting that a patient will have a circulatory failure 2 hours in advance.

3 Available Data Sources

During the Covid-19 outbreak, from February to April 2020 in hospital *Spedali Civili di Brescia* more than two thousand patients were hospitalised. During their hospitalisation, the medical staff performed several exams to them in order to monitor their conditions, checking the response to some treatments, verifying the need to transfer a patient to the ICU, etc. We had data from a total of 2015 hospitalised patients; for each of these patients, the specific data that were made available to us are:

- the age and sex;
- the values and dates of several lab tests (see Table 1);
- the scores (each one from 0 to 18), assigned by the physicians, assessing the severity of the pulmonary conditions resulting from the X-ray exams [3];
- the values and dates of the throat-swab exams for Covid-19;
- the final outcome of the hospitalisation at the end of the stay, which is the classification value of our application (either in-hospital death, released survivor, or transferred to another hospital or rehabilitation center).

Table 1 specifies the considered lab tests, their normal range of values, and their median values in our set of patients. We had no further information about symptoms, their timing, comorbidities, generic health conditions or clinical treatment. Moreover, we have no CT images or text reports associated with the X-ray exams. The available information about whether a patient was or had been in ICU was not clear enough to be used. Finally, of course, also the names of the patient and of the involved medical staff names were not provided.

3.1 Data Quality Issues

When applying machine learning to raw real-world data, there are some non-trivial practical issues to deal with, such as the quality of the available data and related aspects, that in biomedical applications are especially important given the very sensitive domain [12].

In our case, one of such issues is that the length of the hospitalisation period can sensibly differ from one patient to another (from few days to two months), due to different reasons including the novelty and the characteristics of the disease, its high contagiousness or the absence of an effective treatment. Therefore, the number of performed lab tests and relative findings significantly varies among the considered set of patients (from only three to hundreds).

Lab test	Normal Range	Median Value
C-Reactive Protein (PCR)	≤ 10	34.3
Lactate dehydrogenase (LDH)	[80, 300]	280
Ferritin (Male)	[30, 400]	1030
Ferritin (Female)	[13, 150]	497
Troponin-T	≤ 14	19
White blood cell (WBC)	[4, 11]	7.1
D-dimer	≤ 250	553
Fibrinogen	[180, 430]	442
Lymphocite (over 18 years old patients)	[20, 45]	1.0
Neutrophils/Lymphocites	[0.8, 3.5]	4.9
Chest XRay-Score (RX)	< 7	8

Table 1: Lab tests performed during the hospitalisation. In the second column, we show the range which is considered clinically normal for a specific exam. In the third column, we show the median value extracted considering the lab test findings for our set of 2015 patients.

Moreover, the lab tests and X-ray exams are not performed at a regular frequency due, e.g., to the different kinds and timing of the relative procedures, the need of different resources (X-Ray machines, lab equipments, technical staff, etc.), or to the different severity of the health conditions of the patients. For example, in our data we see that a patient can be tested for PCR everyday and not be subject to a Ferritin exam for two weeks. This leads to the need of handling the issues *missing values* and *outdated values*. When we consider a snapshot of a patient at a certain day, we have a missing value for a lab test (or X-ray) feature if that test (X-ray) has not been performed. We have an outdated value for a feature if the corresponding lab test (X-ray) was performed several days earlier: since in the meanwhile the disease has progressed, the findings of the lab test could be inconsistent with the current conditions of the patient, and so they could mislead the prediction.

Data quality issues arise especially patients hospitalised in the period of the highest emergency, when several hundreds of patients were in the hospital at the same time.

3.2 Concept Drift

An examination of the data available for our cohort of patients revealed that their prognostic risk is influenced by multiple factors, such as the number of the patients currently hospitalised and the consequent availability of ICU beds or other resources, the experimentation of new therapies, and the increase of the clinical knowledge.

In machine learning, this change of data distribution is known as *concept drift* [6, 23]. A classical method to deal with this problem is training the algorithm using only a subset of samples, depending on the data distribution that we are considering [6, 24].

For this reason, we divided the considered set of patients into two groups: the **High Contagion Phase (HCP)** group of patients, which is composed by the patients admitted during the last weeks of February and the first weeks of March (the most critical period of the pandemic outbreak in Italy) and the **Moderate Contagion Phase (MCP)** group of patients, which is composed by the patients admitted from the last decade of March to the end of April.

The main differences between these groups of patients are:

1. the mortality rate of the HCP patients is about twice the mortality rate of the MCP patients;

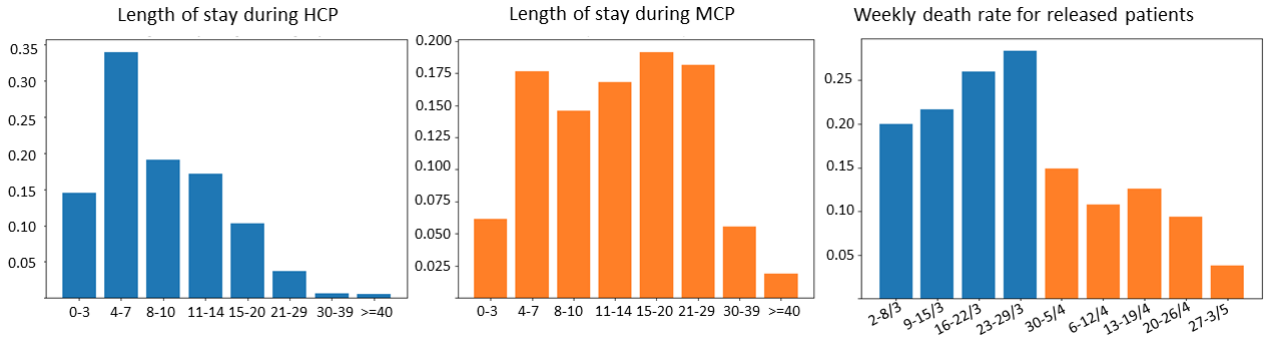


Figure 1: Length of stay in hospital (left) and weekly death rate histograms for the High Contagion Phase (in blue) and for the Moderate Contagion Phase (in orange). On the x-axis, for the length of stay we indicate the range of days, for the death rate we indicate the week when the patient was released. On the y-axis we indicate the percentage of patients.

2. in HCP patients the median value of the hospitalisation period is 8 days, while in MCP patients is 14 days. Further details are given in Figure 1;
3. for many of the considered lab test, the mortality rate associated with having values in a particular range significantly changes in the two groups. For example, in HCP patients the mortality rate for the patients which had a PCR value 10 times above the normal range is 40.1%, while in MCP patients it is 21.1%.

These differences clearly indicate that the data in the HCP and MCP groups represent different target (concept) functions; therefore predicting mortality during the high infection phase and during the moderate phase can be considered as two different tasks. If we had only the patients hospitalised during the high infection phase, using these data for training an algorithm that predicts the mortality during the moderate phase would lead to many errors.

In our case, we generated two different systems, one for each of the two groups of patients. We are currently investigating ways to automatically select the set of patients for training starting from the latest ones, and keeping the less recent ones until we find significant changes in the mortality rate or in the data distribution.

4 Datasets for Training and Testing

The main task of our work is to provide survival/death predictions at different days of the patient hospitalisation, according to the current patient conditions reflected by the available lab findings and X-ray scores. In this section we describe the specific extracted features and the (training and testing) datasets that we built for this purpose.

4.1 Pre-processing and Feature Extraction

The issues presented in Section 3.1 compel us to a robust pre-processing phase with the goal of extracting features in order to summarize the patients conditions and process them by a machine learning algorithm. The pre-processing is applied to both HCP and MCP data.

Given that we have no information about the survival or the decease of a patient after a transfer (which can be due to limited availability of beds or ICU places), we exclude from our training and test set the 142 patients which were admitted in *Spedali Civili di Brescia* and then transferred to another hospital. However, the 74 patients who were transferred to a rehabilitation center can be considered not at risk of death; therefore we include them in our datasets and consider the transferred patients as released alive.

4.1.1 *Patient Snapshot and Feature Engineering*

In order to provide a prediction for a patient at different hospitalisation times, we introduced the concept of **patient snapshot** to represent the patient health conditions at a given day.

In this snapshot, for each lab test of Table 1, we consider its most recent value. In the ideal case, we should know the lab test findings at every day. However, as explained in Section 3.1, in a real-world context the situation is very different. For example, in our data if we consider to take a snapshot of a patient 14 days after the admission into the hospital, we have cases with very recent values of PCR, LDH or WBC (obtained one or a few days before), very old values for Fibrinogen or Troponin-T (obtained the first day of the hospitalisation) and even no value for Ferritin.

Given the difficulty to set a predefined threshold that separates recent and old values of the lab tests (e.g., for Fibrinogen and Troponin-T), we choose to always use the most recent value, even if it could be outdated. In order to allow the learning algorithm to capture that a value may not be significant to represent the current status of the patient (because too old), we introduce a feature called **ageing** for each test finding. If a lab test has been performed at a day d_0, and the snapshot of a patient is taken at day d_1, the ageing is defined as the number of days between d_1 and d_0. If there is no available value for a lab test, its ageing is considered a missing value.

A patient snapshot can contain the values of the lab test findings in two forms: either **numerical**, in which we report the value itself, or **categorical**, in which the value is transformed into an integer number expressing the gravity of the test finding within a partition of the possible real values. This partition is based on the range of values for normal conditions and on how the test values are distributed over the data of all patients. For example, we divide the D-Dimer vales into 6 categories: the normal range, up to 2 times the maximum value of the normal range, up to 4 times, 6 times, 10 times and over 10 times. The categorical form could help the algorithm to have a clearer understanding of the data and improve performance.

Monitoring the conditions of a patient means knowing not only the patient status at a specific time, but also how the conditions evolve during the hospitalisation. For this purpose, we introduce a feature called **trend** that is defined as follows:

> For each lab test, if there is no available value for a lab test or if the patient has not performed the lab test at least two times, the trend is a *missing value*. Otherwise, given the values v_1 and v_2 of the findings for the lab test performed at days d_1 and d_2 and a threshold T that we set to 15% of v_1, if $v_2 > (1 + T) * v_1$, then the trend is *increasing*, while if $v_2 < (1 - T) * v_1$ the trend is

decreasing; otherwise the trend is *stable*.

We distinguish two types of trends: the **start trend**, that uses the distance between the most recent value and the first available value, and the **last trend**, that uses the distance between the last one and the penultimate one. We are currently investigating techniques for including more than two values in the trend calculation.

To summarize, for each lab test in a patient snapshot, we have the most recent finding and the relative ageing and trend, as well as the static features **age** and **sex**.

4.2 Training and Test Sets Generation

In this section we describe how we generated the training and test sets for the purpose of predicting, at different days from the start of the patient hospitalization, the final outcome of her/his stay.

First, for both the HCP and MCP sets, we used stratified sampling for selecting *80% of the patients for training* the models and *20% for testing* them. Then, we created specific training and test sets for each element in a sequence of times when the model is used to make the prediction[1]:

- **2 days** of hospitalisation. We include all the patients' snapshots containing the first values for each lab test conducted in the first two days after the hospital admission. Note that if a patient has performed a lab test more than once in the first two days, the snapshot will consider the oldest value. In fact, the purpose of the model we want to build is to provide the prediction as soon as possible, with the first information available. Furthermore, in these snapshots the ageing and trend values are not included.
- **4 days** and **6 days** of hospitalisation. In these cases, the corresponding snapshots also contain the ageing and trend features, and the lab values will be the most recent ones in the available data. Given that only a few days passed after admission, we consider the *start* trend.
- **8 days** and **10 days** of hospitalisation. The procedure of creating the corresponding snapshots is the same as for the snapshots of 4 days and 6 days cases, except that we consider the *last* trend instead of the start trend.
- **End day** (the last day before the patient is released or the patience decease). In this case, for each lab test the snapshot includes both the start trend and the last trend features.

It is important to observe, that while the datasets of the latter days will contain more information about the single patients (more lab tests findings, less missing values), the overall number of patients in the datasets decreases with the prediction day increase. This is due to the fact that more patients are released or die within longer periods of hospitalisation, and therefore such patients are not included in the corresponding datasets.

Finally, note that the splitting between training and testing of the data is done only once considering all patients. Thus if, for instance, a patient belongs to the training set of 2 days, then it does not belong to the test set of the following days.

5 Machine Learning Algorithms

In this section we briefly describe the machine learning algorithms used in our prognosis prediction system.

[1] While we chose 2, 4, 6, 8, 10 days after the hospitalisation, plus the day before the patient release, of course other sequences could be considered.

5.1 Classification Algorithms

Decision trees

Decision Trees [25] are one of the most popular learning methods for solving classification tasks. In a decision tree, the root and each internal node provides a condition for splitting the training samples into two subsets depending on whether the condition holds for a sample or not. In our context, for each numerical feature f, a candidate splitting condition is $f \leq C$, where C is called *cut point*. The final splitting condition is chosen by finding the f and C providing the best split according to one of some possible measures like Information Gain, Entropy index or Gini index.

A subset of samples at a tree node can either be split again by further feature conditions forming a new *internal node*, or form a *leaf node* labelled with a specific classification (prediction) value; in our application domain the label is either the *alive* class or the *dead* class. Let us consider a decision tree with a leaf node l and a subset S of associated *training* samples. A *test* instance X that reaches l from the root tree, is classified (predicted) y with probability

$$P(y|X) = \frac{TP}{TP + FP}$$

where TP (True Positives) is the number of training samples in S that have class value y, and FP (False Positives) is the number of samples in S that don't have class value y [5]. Given that in our task we have only two classes (y and $\overline{y}$), $P(\overline{y}|X) = 1 - P(y|X)$. The classification outcome of a decision tree for$X$ is the class value with the highest probability.

Random Forests

Random Forests (RF) [4] is an ensemble learning method [32] that builds a number of decision trees at training time. For building each individual tree of the random forest, a randomly chosen subset of the data features is used. While, in the standard implementation of random forests the final classification label is provided using the statistical mode of the class values predicted by each individual tree, in the well-known tool Scikit-Learn [18] that we used for our system implementation, the probability of the classification output is obtained by averaging the probabilities provided by all trees. Hence, given a random forest with n decision trees, a class (prediction) value y is assigned to an instance X with the following probability:

$$P(y|X) = \frac{\sum_{i=1}^{n} P_i(y|X)}{n}.$$

Extra Trees

Extremely Randomized Trees (Extra Trees or ET) [8] are another ensemble learning method based on decision trees. The main differences between Extra Trees and Random Forests are:

- In the original description of Extra Trees [8] each tree is built using the entire training dataset. However in most implementations of Extra Trees, including Scikit-Learn [18], the decision trees are built exactly as in Random Forests.
- In standard decision trees and Random Forests, the cut point is chosen by first computing the optimal cut point for each feature, and then choosing the best feature for branching the tree; while in Extra Trees, first we randomly choose k features and then, for each chosen feature f, the algorithm randomly selects a cut point C_f in the range of the possible f values. This generates a set of k

couples $\{(f_i, C_i) \mid i = 1, \ldots, k\}$. Then, the algorithm compares the splits generated by each couple (e.g., under split test $f_i \leq c_i$) to select the best one using a split quality measure such as the Gini Index or others.

The probability $P(y|X)$ of assigning a class value y to an instance X is computed as in Random Forests (see equation above).

5.2 Hyperparameter Search

Most machine learning algorithms have several hyperparameters to tune such as, for instance, in a Random Forest the number of decision trees to create and their maximum depth. Since in our application handling the missing values is an important issue, we also used a hyperparameter for this with three possible settings: a missing value is set to either the average value, the median value or a special constant (-1).

In order to find the best performing configuration of the hyperparameters, we used the Random Search optimization approach [2], which consists of the following main steps:

1. We divide our training sets into k folds, with either $k = 10$ or $k = 5$, depending on the dimension of the considered dataset.
2. For each randomly selected combination of hyperparameters, we run the learning algorithm in k-fold cross validation.
3. For each fold, we evaluate the performance of the algorithm with that configuration using the Macro F-β score metric and $\beta = 2$. The F-β score is the weighted harmonic mean of precision and recall measures. The β parameter indicates how many times the recall is more important with respect to the precision:

$$F\text{-}\beta = (1 + \beta^2) * \frac{precision + recall}{\beta^2 * precision + recall}$$

   We choose $\beta = 2$ in order to give particular importance to false negatives, i.e. those patients which our system could not identify as at death risk. Given that we can compute the F2-score both for both the *alive* class and the *dead* class, we considered the Macro F2-Score, which is the arithmetic mean of the scores for the two classes.
4. The overall evaluation score of the k-fold cross validation for a configuration of the parameters is obtained by averaging the scores obtained for each fold.
5. The hyperparameter configuration with the best overall score is selected.

5.3 Handling Prediction Uncertainty

The output for an instance X of every generate classification model is an array of two probabilities, $P(alive|X)$ and $P(dead|X)$, defined as described in Section 5.1. We can see them as "degrees of certainty" of the prediction: the higher the probability is, the more reliable the prediction is. Given the very sensitive nature of our task, the system discards potential predictions supported by a low probability. This is achieved using a *prediction threshold* under which the system considers the prediction **uncertain** (and the patient risk unpredictable). Note that if we used a threshold value that is too high, many patients could be classified uncertain, and our model would be much less useful for clinical practice. To avoid this, at training time we impose a maximum percentage of samples that can can be considered uncertain (unpredictable), and we implemented this with a parameter, called max_u, that is given in input; for our experimental analysis we used $max\_u = 25\%$.

FINDUNCERTAINTHRESHOLD**:** Algorithm for computing, during the training phase, an optimised prediction threshold under which the model labels an instance as uncertain.

Input:
- L array of labels ($alive$ or $dead$) l_i with $l[i]$ label of the sample i of the validation data (fold);
- $P = [p_i = (p_{alive}, p_{dead})_i \mid i$ is the sample index in val. set];
- max_u the maximum percentage of the samples in the validation set that can be labeled as uncertain (not predictable);
- n the maximum number of thresholds to try;
- $EvaluateScore$ the score function to maximize by dropping the uncertain samples;

Output: A pair (v, th) where v is the score function value after dropping the uncertain samples and th the optimized threshold value.

1 $L_{pred} \leftarrow$ array of labels such that $L_{pred}[i]$ is the predicted label (the label with highest probability) of the val. sample i;
2 $P_{max} \leftarrow [max(p_{alive}, p_{dead})_i \mid (p_{alive}, p_{dead})_i \in P]$;
3 $v \leftarrow EvaluateScore(L, L_{pred})$;
4 $th \leftarrow$ min value in P_{max};
5 $\delta \leftarrow$ [(max value in P_{max}) − (min value in P_{max})]$/n$;
6 **for** $i \leftarrow 0$ *to* $n - 1$ **do**
7 $th' \leftarrow$ min value in $P_{max} + i \cdot \delta$;
8 $S \leftarrow \{i \,|i$ is id sample such that $P_{max}[i] > th'\}$
9 $u \leftarrow 1 - (|S|/|P_{max}|)$;
10 **if** $u \geq max\_u$ **then return** (v, th);
11 $L' \leftarrow$ array of labels such that $L[i]$ is the label of the val. sample i and $i \in S$;
12 $L'_{pred} \leftarrow$ array of labels such that $L_{pred}[i]$ is the predicted label of the val. sample i and $i \in S$;
13 $v' \leftarrow EvaluateScore(L', L'_{pred})$;
14 **if** $v' > v$ **then**
15 $th \leftarrow th'$;
16 $v \leftarrow v'$;
17 **end**
18 **end**

Figure 2: Pseudocode of algorithm FINDUNCERTAINTHRESHOLD.

We designed an algorithm called FINDUNCERTAINTHRESHOLD that is used in the training phase to decide the threshold and optimize the prediction performance on the training samples that pass it, under the max_u constraint. The pseudocode of the algorithm is in Figure 2.

Given the original labels L of the validation samples and their prediction probabilities P derived by the learning algorithm, FINDUNCERTAINTHRESHOLD first computes: the predicted labels L_{pred} (i.e., the class values with highest probabilities) and the relative P_{max} probabilities; the original score v obtained using the input score function evaluating *all* samples; an initial value of the threshold (th) defined as to the minimum probability in P_{max}.

The next loop finds an optimal value of threshold th and computes the score function for the validation set reduced to the validation samples with predicted labels that have probabilities above th. The considered threshold values are obtained by using the δ-increments defined at lines 5 and 7. First we compute the new threshold th' increasing the current threshold by δ, and then we derive the set S of sample ids with prediction probabilities higher than th'. Next we compute the percentage u of samples that are labeled as uncertain using threshold th'. If $u \geq max\_u$, we can terminate returning the

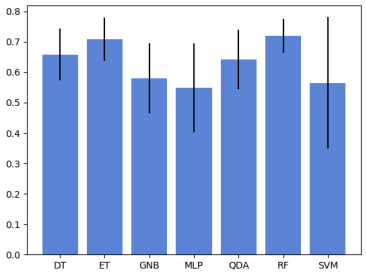


Figure 3: Average performance (F2 score) of seven machine learning algorithms for the HCP datasets. The line over the bar represents the standard deviation.

current best new score v and the corresponding threshold value th (a greater threshold value cannot lead to label as uncertain less samples than the returned th value). Otherwise ($u < max\_u$), we compute the correct sample labels L' and the predicted sample labels L'_{pred} for the samples identified by S, and we compute the new score value v' using L' and L'_{preds}. If v' is a better score than v, we update both the threshold and the score values.

FINDUNCERTAINTHRESHOLD is executed during the training phase. In particular during the hyperparameter search, for each attempted hyperparamenter configuration, we compute through FINDUNCERTAINTHRESHOLD an optimized threshold and the relative score function value. These two values are obtained by averaging the optimal thresholds and corresponding scores over all folds of the cross validation for the attempted configuration. The hyperparameter search returns the best configuration together with the relative (averaged) threshold.

6 Experimental Evaluation and Discussion

In this section, we evaluate the performance the of the machine learning models that we built. Our system was implemented using the Scikit-Learn [18] library for Python, and the experimental tests were conducted using a Intel(R) Xeon(R) Gold 6140M CPU @ 2.30GHz.

The performance of the learning algorithms with the relative optimized hyperparameters was evaluated using the test set in terms of F2 score and ROC-AUC score. The second metric is defined as the area under the Receiver Operating Characteristic curve, which plots the true positive rate against the false positive rate, and it takes also into account the probability that the predictive system produces false positives (i.e. false alarms). This metric is a standard method for evaluating medical tests and risk models [9, 10].

In a preliminary study we examined various machine learning approaches and we compared their average performances over the HCP datasets. Figure 3 shows a summary of the relative performance in terms of F2 score. We considered Decision Trees [25], ExtraTrees (ET) [8], Gaussian Naive Bayes [31], Multilayer Perceptron with two layers (MLP) [13], Quadratic Discriminant Analysis [26], Random Forests (RF) [4] and Support Vector Machines [27]. The best performance was obtained with RF and ET. NN and SVM performed much worse and with a much higher variability over the datasets, probably related to the missing values and the scarcity of data. For the MCP datasets the relative performance was similar. Given the observed better performance of RF and ET, we focused the evaluation of our system on these learning algorithms

Regarding the training time, including the hyperparamenter search over 4096 random configurations and the optimization of the uncertainty threshold, for any specific dataset (e.g., the MCP numerical dataset for 2 days), the overall training time is between 20 and 30 minutes. Therefore, we can build all the four most promising models generated by RF and ET using the numerical version (RF-N, TC-N) or the categorical version (RF-C, ET-C) of the data set in less than two hours, and then select the best performing model among them.

It is also worth to note that in our system the models for predicting the prognostic risk at different days are completely independent from each other, and so we can consider prediction tasks at different days as different tasks.

In Figure 4 and in Table 2 we show the performances of our system at each considered day for both the High Contagion Phase and the Moderate Contagion Phase. As we can see, we obtain promising results in terms of F2 score for an early evaluation of the risk during the HCP (with score 77.1% at day 2), while we encounter some problems at the 6th and 10th days. For the MCP datasets, the system performs better at the latter days, in particular for the 10th day F2 is 80.4% and ROC-AUC is 90.2%. For HCP, both RF and ET obtain good results in both the numerical and categorical versions of the datasets. Instead, for MCP using the categorical datasets does not give good performance, and we do not observe an improvement for the latter prediction days (the F2 score is always below the 70%).

In all but one case, the models using the uncertain threshold increase the performance in terms of both F2 and ROC-AUC scores. In particular, in the most problematic cases of HCP, such as for the 6-days and 10-days datasets, the prediction performance improves in terms of F2 by over than 7 points. The improvement is less significant for MCP.

Note that, while the threshold value under which the system labels an instance (patient risk) as uncertain is derived at training time imposing a maximum percentage of uncertain samples (we used 25%), there is no formal guarantee that this percentage limit is satisfied for test set. However, in most cases the percentage of uncertain test samples (indicated with % Unc in Table 2) is much below the limit imposed during training, expect for the test set of the 6th day in HCP, where the unpredicted (labelled as uncertain) patients are 26.1%. The performance for the "end" dataset is good for both HCP and MCP even without omitting the uncertain patients (F2 score 86.6% for HCP, and F2 score 86.9% for MCP).

Figure 4 gives graphical pictures comparing the performance of our system for HCP and MCP in terms of F2 and ROC-AUC. The performance behaviour over time significantly differs in the two contagion periods, reflecting the concept drift we discussed in Section 3.2. For HCP, considering the results without omitting the uncertain test instances (blue curves), the performance prediction is very good at the 2nd day and it decreases at the 6th and 10th days. Instead, for MCP the performance improves over time, reaching 90.2% in terms of ROC-AUC at the 10th day, as also reported in Table 2. This is due to several factors:

- MCP includes patients that have hospitalisation periods much longer than the patients in HCP, which can make more difficult to predict the mortality risk for some patients with only a few days of hospitalisation;
- on the contrary, in HCP half of the patients stayed in hospital for less than 8 days. This decreases significantly the size of the 8-days and 10-days training sets, which contain respectively only 431 and 339 patients. The lack of training data in these datasets is only partially compensated by the increase of the lab tests for a single patient in the datasets;
- as described in Section 3.2, the MCP patients are much more unbalanced (with only 11% deceased patients) than the HCP patients, and this increases the difficulty of learning an high performing model [17].

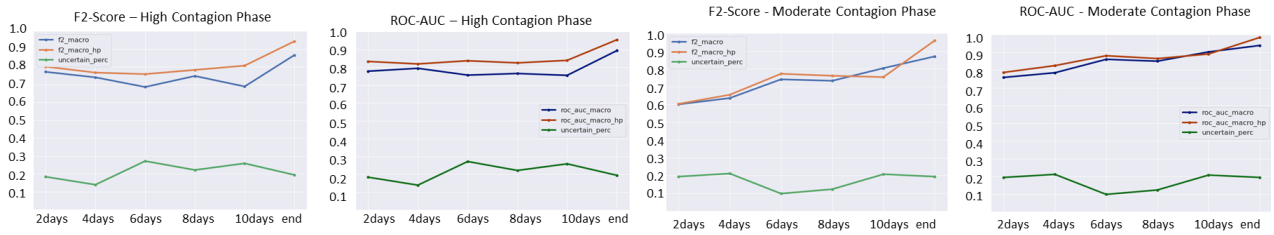


Figure 4: Graphical representation of the prediction performance (F2 and ROC-AUC scores) over hospitalisation time for HCP and MCP.

HCP data	F2	ROC	F2-U	ROC-U	% Unc	Model	MCP data	F2	ROC	F2-U	ROC-U	% Unc	Model
2days	77.1	77.8	80.1	83.3	18.3	ET-C	2days	60.0	75.4	61.0	78.1	13.9	ET-N
4days	74.1	79.4	76.7	81.9	13.8	RF-N	4days	63.5	78.5	65.4	82.4	21.1	RF-N
6days	68.7	75.6	75.9	83.6	26.1	RF-N	6days	74.1	86.0	77.2	88.1	9.8	ET-N
8days	74.8	76.5	78.2	82.5	22.1	ET-C	8days	73.2	85.0	76.1	86.5	12.3	ET-N
10days	68.9	75.5	80.6	83.9	24.8	RF-C	10days	80.4	90.2	75.3	89.0	12.7	ET-N
end	86.6	89.4	94.3	95.5	19.3	RF-C	end	86.9	93.9	95.8	98.4	19.4	RF-N

Table 2: Predictive performance for the High Contagion Phase (HCP, left) and the Moderate Contagion Phase (MCP, right) in terms of F2 and ROC-AUC scores considering all instances in the test set (columns F2 and ROC) and omitting the instances classified uncertain (columns F2-U and ROC-U). The percentages of instances that the system classifies uncertain are in the column % Unc. Column Model indicates the method selected for generating the model; ET stands for Extra Trees, RF for Random Forest, C for categorical and N for numerical.

Figure 5 shows the confusion matrices for the test sets generated using our predictive models. Above the line we have the HCP datasets and below the MCP datasets. Despite the training phase was optimised (through the use of the F2 metric) to avoid false negatives, for the HCP datasets there are several false negatives (bottom-left of the matrices). This can be explained by the scarcity of lab test and X-ray data in the HCP data that affects prediction.

However, false negatives are significantly reduced with the models that can classify a patient as uncertain. For example, at day 6, the system classifies as uncertain 4 patients who otherwise would be false negatives. Moreover, when there are less false negatives, such as at days 8 and 10, classifying some patients as uncertain helps to also avoid false positives and so to generate less false alarms.

Remarkably, especially for the MCP datsets, we have very few false negatives even at the early days, which is quite important in our application context. On the other hand, especially for days 2 and 4, our system produces many false positives. This type of error is reduced in the models with uncertain patients up to only 5 false alarms for the end dataset (e.g., at day 2 we avoid 16 false positives.)

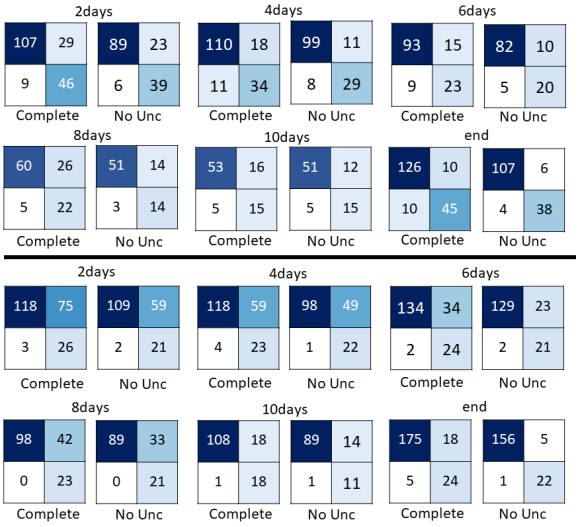


Figure 5: Confusion matrices for datasets HCP (above the line) and MCP (below the line) at different days with *dead-alive* predictions for all patients (Complete) and omitting patients classified uncertain (No Unc). For each matrix of 4 numbers, on the main diagonal we have the correct predictions (*alive* class on the top-left corner and *dead* class on the bottom-right corner); on the anti-diagonal, we have the incorrect predictions (false positives and on the top-right corner and the false negatives on the bottom-left corner).

7 Conclusions and Future Work

We have presented a system for predicting the prognosis of Covid-19 patients focusing on the death risk. We built and engineered some datasets from lab test and X-ray data of more than 2000 patients in an hospital in northern Italy that was severely hit by Covid-19. Our predictive system uses a collection of machine learning algorithms and a new method for setting, at training time, an uncertain threshold for prediction that helps to significantly reduce the prediction errors.

Overall, the experimental results are quite promising, and show that our system often obtains high ROC-AUC scores. The observed predictive performance is especially good in terms of false negatives (patients erroneously predicted survivor), that are very few. This gives a predictive test for patient survival with very good specificity in particular when the system can classify a patient as uncertain.

On the other hand, in terms of false positives, there is room for significant improvements. We are confident that the availability of more information, such as patient comorbidities or clinical treatments, will help to improve performance, reducing the number of both false positives and (few) false negatives.

For future work we plan to extend our datasets with more information (both additional features and patients), to consider further methods for dealing with the observed concept drift and to address other prediction tasks such as the duration of the hospitalisation or the need of ICU beds and critical hospital resources. Moreover, we are analyzing the importance of the features used in our models, and we intend to investigate additional learning techniques.

Acnowledgements. The work of the first author has been supported by Fondazione Garda Valley.

REFERENCES

[1] Aya Awad, Mohamed Bader–El–Den, and James McNicholas, 'Patient length of stay and mortality prediction: A survey', *Health Services Management Research*, **30**(2), 105–120, (2017). PMID: 28539083.

[2] James Bergstra and Yoshua Bengio, 'Random search for hyper-parameter optimization', *Journal of machine learning research*, **13**(Feb), 281–305, (2012).

[3] Andrea Borghesi and Roberto Maroldi, 'Covid-19 outbreak in italy: experimental chest x-ray scoring system for quantifying and monitoring disease progression', *La radiologia medica*, (05 2020).

[4] Leo Breiman, 'Random forests', *Machine learning*, **45**(1), 5–32, (2001).

[5] NV CHAWLA, 'Evaluating probability estimates from decision trees', in *Proc. AAAI Workshop on Evaluation Methods for Machine Learning, Boston, MA, 2006*, pp. 18–23, (2006).

[6] João Gama, Indrundefined Žliobaitundefined, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia, 'A survey on concept drift adaptation', *ACM Comput. Surv.*, **46**(4), (March 2014).

[7] Alfonso Emilio Gerevini, Alberto Lavelli, Alessandro Maffi, Roberto Maroldi, Anne-Lyse Minard, Ivan Serina, and Guido Squassina, 'Automatic classification of radiological reports for clinical care', in *Proceedings of the 16th Conference on Artificial Intelligence in Medicine, AIME 2017, Vienna, Austria, June 21-24, 2017*, volume 10259 of *Lecture Notes in Computer Science*, pp. 149–159. Springer, (2017).

[8] Pierre Geurts, Damien Ernst, and Louis Wehenkel, 'Extremely randomized trees', *Machine learning*, **63**(1), 3–42, (2006).

[9] Gary L Grunkemeier and Ruyun Jin. Receiver operating characteristic curve analysis of clinical risk models, 2001.

[10] Karimollah Hajian-Tilaki, 'Receiver operating characteristic (roc) curve analysis for medical diagnostic test evaluation', *Caspian journal of internal medicine*, **4**(2), 627, (2013).

[11] Hrayr Harutyunyan, Hrant Khachatrian, David C Kale, Greg Ver Steeg, and Aram Galstyan, 'Multitask learning and benchmarking with clinical time series data', *Scientific data*, **6**(1), 1–18, (2019).

[12] Sharique Hasan and Rema Padman, 'Analyzing the effect of data quality on the accuracy of clinical decision support systems: a computer simulation approach', in *AMIA annual symposium proceedings*, volume 2006, p. 324. American Medical Informatics Association, (2006).

[13] Simon Haykin, *Neural networks: a comprehensive foundation*, Prentice Hall PTR, 1994.

[14] Stephanie Hyland, Martin Faltys, Matthias Hüser, Xinrui Lyu, Thomas Gumbsch, Cristóbal Esteban, Christian Bock, Max Horn, Michael Moor, Bastian Rieck, Marc Zimmermann, Dean Bodenham, Karsten Borgwardt, Gunnar Rätsch, and Tobias Merz, 'Early prediction of circulatory failure in the intensive care unit using machine learning', *Nature Medicine*, **26**, 1–10, (03 2020).

[15] Xiangao Jiang, Megan Coffee, Anasse Bari, Junzhang Wang, Xinyue Jiang, Jianping Huang, Jichan Shi, Jianyi Dai, Jing Cai, Tianxiao Zhang, et al., 'Towards an artificial intelligence framework for data-driven prediction of coronavirus clinical severity', *CMC: Computers, Materials & Continua*, **63**, 537–51, (2020).

[16] Alistair EW Johnson, Mohammad M Ghassemi, Shamim Nemati, Katherine E Niehaus, David A Clifton, and Gari D Clifford, 'Machine learning and decision support in critical care', *Proceedings of the IEEE*, **104**(2), 444–466, (2016).

[17] Bartosz Krawczyk, 'Learning from imbalanced data: open challenges and future directions', *Progress in Artificial Intelligence*, **5**(4), 221–232, (2016).

[18] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, 'Scikit-learn: Machine learning in Python', *Journal of Machine Learning Research*, **12**, 2825–2830, (2011).

[19] Tom J Pollard and Leo Anthony Celi, 'Enabling machine learning in critical care', *ICU management & practice*, **17**(3), 198, (2017).

[20] Luca Putelli, Alfonso Gerevini, Alberto Lavelli, Matteo Olivato, and Ivan Serina, 'Deep learning for classification of radiology reports with a hierarchical schema', in *Proceedings of 24th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems*, (2020).

[21] Luca Putelli, Alfonso Gerevini, Alberto Lavelli, and Ivan Serina, 'The impact of self-interaction attention on the extraction of drug-drug interactions', in *Proceedings of the Sixth Italian Conference on Computational Linguistics*, (2019).

[22] Luca Putelli, Alfonso Emilio Gerevini, Alberto Lavelli, and Ivan Serina, 'Applying self-interaction attention for extracting drug-drug interactions', in *XVIIIth International Conference of the Italian Association for Artificial Intelligence, Rende, Italy, November 19–22, 2019, Proceedings*, (11 2019).

[23] Joaquin Quionero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D Lawrence, *Dataset shift in machine learning*, The MIT Press, 2009.

[24] Anna S Rakitianskaia and Andries Petrus Engelbrecht, 'Training feed-forward neural networks with dynamic particle swarm optimisation', *Swarm Intelligence*, **6**(3), 233–270, (2012).

[25] Lior Rokach and Oded Maimon, *Data Mining with Decision Trees: Theory and Applications*, World Scientific Publishing Co., Inc., River Edge, NJ, USA, 2008.

[26] Santosh Srivastava, Maya R Gupta, and Béla A Frigyik, 'Bayesian quadratic discriminant analysis', *Journal of Machine Learning Research*, **8**(Jun), 1277–1305, (2007).

[27] Johan AK Suykens and Joos Vandewalle, 'Least squares support vector machine classifiers', *Neural processing letters*, **9**(3), 293–300, (1999).

[28] Mihaela van der Schaar and Ahmed Alaa, 'How artificial intelligence and machine learning can help healthcare systems respond to covid-19', *https://www.vanderschaar-lab.com/covid-19/*, (2020).

[29] Li Yan, Hai-Tao Zhang, Yang Xiao, Maolin Wang, et al., 'Prediction of criticality in patients with severe covid-19 infection using three clinical features: a machine learning-based prognostic model with clinical data in wuhan', *medArxiv preprint*, (2020).

[30] Jinsung Yoon, Ahmed Alaa, Scott Hu, and Mihaela Schaar, 'Forecasticu: a prognostic decision support system for timely prediction of intensive care unit admission', in *International Conference on Machine Learning*, pp. 1680–1689, (2016).

[31] Harry Zhang, 'The optimality of naive bayes', in *Proceedings of the Seventeenth International Florida Artificial Intelligence Research Society Conference, Miami Beach, Florida, USA*, eds., Valerie Barr and Zdravko Markov, pp. 562–567. AAAI Press, (2004).

[32] Zhi-Hua Zhou, *Ensemble Methods: Foundations and Algorithms*, Chapman & Hall/CRC, 1st edn., 2012.

Knowledge Discovery and Visualization in Healthcare Datasets using Formal Concept Analysis and Graph Databases

Diana Cristea and **Christian Săcărea** and **Diana-Florina Şotropa** [1]

Abstract. Among the major advances in Artificial Intelligence we can mention Knowledge Discovery, Processing and Representation. Since in our modern society the healthcare system plays an important role and has a major impact in our daily lives, it lies at hand to apply the aforementioned methods in order to discover relevant patterns in healthcare databases and then to represent them in a way which supports reasoning, decision making, and communication. We approach this task by using two complementary directions, which are then interlinked. On the one hand we make use of the graphical representation capabilities of Formal Concept Analysis (FCA) and its powerful algorithms for conceptual knowledge discovery and processing. On the other, we use graph databases as a complementary visualization method of the extracted knowledge patterns. We exemplify this approach on a particular medical dataset, highlighting a 3D representation of conceptual hierarchies by using virtual reality (VR).

1 Introduction

Formal Concept Analysis (FCA) is a prominent field of applied mathematics which formalizes the classical philosophical understanding of a concept as a unit of thought and provides powerful algorithms for knowledge discovery, processing and representation. FCA is well known for its expressive and intuitive graphical representation of knowledge. The basic data structure is a formal context, i.e., a universe of discourse, and knowledge extraction is restricted to concepts, particular patterns which constitute building blocks of the knowledge encapsulated in the dataset. Concepts are ordered and displayed in an order diagram, called concept lattice or conceptual hierarchy. Due to its elementary yet powerful formal theory, FCA can express other methods, and therefore has the potential to unify the methodology of data analysis. Summarizing, FCA is a human-centered method to structure and analyze data, as well as a method to visualize data and its inherent structures, implications and dependencies.

How well can healthcare systems be used in order to support physicians? As researchers, we cannot stop asking what we should do in order to improve them. When trying to assemble and analyze medical data, we all have the same purpose: to aid both patients and care providers, while improving the outcomes and offering personalised care. A common approach followed in order to extract knowledge from the large amount of collected data usually starts with data preprocessing and analysis, which is then usually continued with extraction of knowledge or patterns. Once extracted, this knowledge can be used in various ways, from improvement of medical systems, to understanding and prediction issues or for learning.

This paper is presenting some current research about using FCA and graph databases to discover knowledge in healthcare databases. The data is organized and represented in conceptual landscapes of knowledge, using a methodology developed by R. Wille [22]. These conceptual landscapes of knowledge can be used, for instance to understand the way how patterns are arising from medical data, to investigate analogies between symptoms and treatments, to support communication and they can be be integrated into a decision support system that assists doctors in the process of diagnosis. One major step forward is switching from 2D to 3D using VR and establishing virtual discussion rooms where multiplayers can navigate and explore conceptual knowledge. On the other hand, graph databases are offering a different perspective. They enable us to analyze different connections between data, using a graph based approach.

The contributions of our paper include detecting and extracting knowledge patterns from healthcare data as well as presenting some visualization techniques for these, both in 2D and 3D format.

The paper is structured as follows. Section 2 describes some related work, while Section 3 contains some preliminaries, introducing the method use to extract the knowledge from the dataset, namely Formal Concept Analysis, and graph databases. Section 4 presents some experiments while trying to show how new information about medical investigations can be discovered using knowledge graphs. Section 5 concludes the work presented and highlights some future research directions.

2 Related Work

Artificial Intelligence is a wide field comprising a large set of methods and algorithms that can be applied in multiple fields. Given the nature and the importance of medicine in our lives, a large number of researchers work on applications in the medical field. A lot of the work in this field is focused on prediction models for diseases using different data mining methods. For instance, Delen et al. present a comparison of three data mining methods (logistic regression, artificial neural networks and decision trees) for predicting breast cancer survivability [2]. However, an important part of the medical system is the diagnosis of the patients. In this sub-field there is a lot of work to be done in order to build systems that can aid practitioners in their decisions. For example, one of the previous authors identifies this in a subsequent paper, where different machine learning techniques are applied to build predictive models [23]. Their conclusions are

[1] Babeş-Bolyai University, Romania, email: dianat@cs.ubbcluj.ro, csacarea@math.ubbcluj.ro, diana.halita@ubbcluj.ro

that having more information about the patients' conditions can improve models' predictive power which then can help practitioners make better diagnostic and treatment decisions.

When dealing with electronic health record (EHR) it is well known that the volume of data can easily become too large for humans to process. Therefore, the need of implementing support systems that assist clinicians in examining the data has been previously identified and acknowledged by medical experts and researchers. For instance, Fujita et al. propose in their paper to improve the user experience by limiting the visual format. They define several screen designs based on some identified principles, such as: limiting a view to a single patient data, summarizing an overview of the data and give details on demand [3]. In this approach, the advantage of having a large collection of data is lost and becomes rather a disadvantage. Such approaches lose sight of important information correlating different cases, symptoms and diagnostics, which can give an important and useful insight into the data. For that reason we believe that a better approach is to find ways of taking advantage of all the patterns contained in the data and, instead of cutting down on the data visualized, find new methods of knowledge discovery and representation techniques that allow clinicians to have an overview of the data and at the same time to be able to infer knowledge from the data, such as useful correlations and patterns.

Through FCA, medical data will be scaled so that it can be modeled as objects possessing attributes, in order to allow the discovery and to visualize of implications between them. The formal concept is the unit of measure and the central point from which the pattern mining begins. The graphical representation is given by the concept lattice containing all formal concepts.

Formal Concept Analysis can be applied in multiple fields proving that it is a suitable information retrieval technique [13]. There are also some application of FCA in the medical field. Gupta et al. use context reduction techniques along with classification rules in order to find redundancies among various medical examination tests [5]. Jay et al. use FCA for mining and interpreting patient flows within a healthcare network [7]. Pan et al. propose to use FCA in order to provide a method for modeling and designing a multidisciplinary clinical process, in which medical specialists can coordinate the treatment of specific groups of patients [12].

In our previous work, we have applied FCA techniques on different medical datasets, such as Otorhinolaryngology data, cancer registry and drug adverse reaction. For all these cases the medical datasets are considered as many-valued contexts, and they are subject to conceptual scaling in order to build knowledge landscapes. For instance, we have used several methods of conceptual knowledge processing to build a logical information system for oncological databases [1]. In some other work the scaling effort of FCA was focused on attributes describing treatment options and their results, the type and location of cancerous cells and the adverse drug reactions [16]. The databases were analyzed from different perspectives, using dyadic formal contexts as well as triadic formal contexts [6] . The triadic setting offers conditions as a third dimensions which can lead to a better understanding for instance in the case of adverse drug reactions [15].

Furthermore, we have used analogical reasoning combined with FCA in order to offer valuable support for decision making in a medical setting. The purpose of this is to improve the interaction between clinicians and electronic health record systems [19]. In some of our previous papers we started analyzing how combining different mining techniques and visualization methods, such as analogical reasoning, FCA and graph databases, can bring a fresh perspective over the medical process and improve the task of knowledge discovery in EHR systems [17, 18]. The results obtained so far highlight the fact that FCA is suitable for improving electronic health record systems.

3 Preliminaries

3.1 Formal Concept Analysis

Formal Concept Analysis (FCA) was introduced by Bernhard Ganter and Rudolf Wille in the early 1980s [4]. The theory has its mathematical basis in general lattice theory created by Garrett Birkhoff in the 1930s. One advantage of the FCA analysis techniques is that the FCA tools do not require extensive knowledge on lattice theory in order to be used and interpreted, which makes FCA a suitable and accessible method for information retrieval.

There are three kinds of relations that exist among concepts: independence, intersection and inheritance. Based on these relations, knowledge about the data can be extracted, and often causal relations can be identified [12].

We will briefly recall some definitions introduced by Rudolf Wille in [22] regarding formal concept, formal context, many-valued contexts and conceptual scaling. A formal context is a triple (G, M, I) where G and M are sets and $I \subseteq G \times M$ is a binary relation, called incidence relation. A Galois connection on the powersets of G and M respectively is defined and is used as a concept forming operator. More precisely, for $A \subseteq G$, we define $A' := \{m \in M \mid \forall a \in A, (a, m) \in I\}$, and dually for $B \subseteq M$, we define $B' := \{g \in G \mid \forall b \in B, (g, b) \in I\}$. A formal concept is a pair (A, B) with $A \subseteq G$, $B \subseteq M$, and $A' = B$, $B' = A$. Concepts are ordered by the subconcept-superconcept relation and the resulting structure is a complete lattice, called concept lattice or conceptual hierarchy, and it can be graphically represented as an order diagram. Every node of this order diagram represents a concept, while the path connecting the nodes upwards or downwards are exactly the subconcept-superconcept relation. Using a reduced labeling, only some particular concepts are labeled with the elements from G and M, respectively, more exactly those which are supremum or infimum irreducible in the lattice.

A many-valued context (G, M, W, I) consists of sets G, M, and W and a ternary relation I between G, M and W (i.e., $I \subseteq G \times M \times W$) for which it holds that $(g, m, w) \in I$ and $(g, m, v) \in I$ always implies $w = v$. The triple $(g, m, w) \in I$ is read as "the attribute m has the value w for the object g". The many-valued attributes can be regarded as partial maps from G in W. Therefore, it seems reasonable to write $m(g) = w$ instead of $(g, m, w) \in I$. In order to derive the conceptual structure of a many-valued context, we need to scale every many-valued attribute. This process is called conceptual scaling and it is always driven by the semantics of the attribute values.

A scale for the attribute m of a many-valued context is a formal context $S_m := (G_m, M_m, I_m)$ with $m(G) \subseteq G_m$. The objects of a scale are called scale values, the attributes are called scale attribute. Every context can be used as a scale. Formally there is no difference between a scale and a context. However, we will use the term "scale" only for contexts which have a clear conceptual structure and which bear meaning. The set of scales can then be used to navigate within the conceptual structure of the many-valued context (and the subsequent scaled context). Some scales are predefined (like nominally, ordinally, etc.), while for more complex views, we need to define particular scales.

3.2 Graph Databases

Graphs are data structures containing nodes with pairwise relationships between them, represented as edges. When the edge corresponds to an ordered pair of nodes, then the graph is called a directed graph, otherwise it is an undirected graph. A strongly connected component in an undirected graph is a maximal region within which each node is reachable from any other node. When defining this notion for directed graphs the direction of the edges plays an important role. Hence, we can define a strongly connected component for a directed graph as a maximal subset of nodes such that there is a directed path from any node to any other node. Strongly connected components can be very useful in an early phase of the data analysis in order to see how the graph is structured and to identify clusters of data that have similar behavior. Graph algorithms provide one of the most powerful approaches to analyzing connected data since they are relationship-oriented [11, 21].

Graphdatabases use a graph datamodel as opposed to the relational data model used in most database management systems. The main advantage of graph databases is that representing the data as a graph structure gives a more intuitive representation of the data rather than the relational structured databases or other table structures. Another reason for considering graph databases rather than the well established and widely spread relational or NoSQL data models is performance. There are use cases when a graph database is much more efficient and flexible for the implementation, mostly because a graph database can use graph-specific algorithms which in a different setting have a higher complexity [14]. The flexibility is given by the fact that the graph model is easily extensible. In contrast, when dealing with changes in a relational database one must make structural changes that can affect the existing data.

Graph databases basically consist of a labeled property graph model. In a graph database entities are represented as nodes of the graph and labels are used to express that a certain node belongs to a particular category. Nodes contain properties in form of key-value pairs. The structure of the graph is given by relationships among nodes. Relationships have a direction and a role property, i.e. a name, which together give the meaning of that relationship and show how two nodes are associated. For the implementation we used Neo4j [8] which enables us to build a knowledge graph for the analyzed dataset. Neo4j is a highly scalable and easy to manage graph database that offers an efficient query language implementation called Cypher.

4 Knowledge discovery in medical data

Medical diagnosis is regarded as an important yet difficult task that needs to be executed accurately and efficiently. Regarding accuracy, in practice, one can still find a high number of wrong diagnostics. Regarding efficiency, sometimes even if reaching the correct diagnostic, a lot of tests are performed on the patient, some of which may be irrelevant for the condition of the patient. This can be a time-consuming and costly process which can be optimized with the help of technologies that assist the doctors in their decisions. For this reason, we use FCA as a mining technique that has the potential to generate conceptual structures that can improve the quality of clinical decisions.

We are considering a collection of data from the Otorhinolaryngology department from a teaching hospital in Romania. This department is specialized in the diagnosis and treatment of ear, nose and throat disorders. The data collected for multiple patients contains information about symptoms presented by the patients and the diagnostics given by the doctors following a set of test and investigations. According to the importance of the symptoms and diagnostics, they are each divided into two categories: principal and secondary symptoms, respectively diagnostics.

Our analysis focuses on finding and visualizing patterns among different pairs of these elements, for instance analyzing correlations among principal and secondary symptoms, or among principal symptoms and principal diagnostics.

Using these data, we show how new knowledge about medical investigations can be discovered, by following 3 steps: finding concepts, finding relations between concepts and building knowledge concept lattices. Datasets are interpreted as many-valued contexts. We use `FCA Tools Bundle`[2] system ([9, 10]) to build conceptual scales and to visualize knowledge clusters.

Figure 1 reveals the correlations that exists in the dataset considering the dyadic case of diagnostics and symptoms. Due to the huge amount of data and for the purpose of the article, we have filtered our data by selecting only patients who had `Deviated Septum` and `Chronic Sinusitis` among the secondary diagnostics list. We chose to do that in order to exemplify our theory on a relatively small dataset. Afterward, we have selected diagnostics as objects and symptoms as attributes in order to build the formal context.

Deviated Septum	occurs when the thin wall (nasal septum) between your nasal passages is displaced to one side
Chronic Sinusitis	occurs when the spaces inside your nose and head (sinuses) are swollen and inflamed for three months or longer, despite treatment.
Chronic Otitis Media	describes some long-term problems with the middle ear, such as a hole (perforation) in the eardrum that does not heal or a middle ear infection (otitis media) that doesn't improve or keeps returning.
Otosclerosis	is a condition where one or more foci of irregularly laid spongy bone replace part of normally dense enchondral layer of bony otic capsule in the bony labyrinth.
Chronic Pharyngitis	is the chronic inflammation of the pharynx.

Table 1. Diagnostics from the extent of the highlighted node and their medical definitions

Autophony	the unusually loud hearing of a person's own voice
Ear Fullness	a sensation of pressure within the middle ears. This sensation is similar to the fullness we experience going up and down in airplanes or sensation of being deep under water. This pressure is horribly uncomfortable and severely distracting from everyday responsibilities and enjoyments.
Hearing Loss	deafness, or hard of hearing
Headache	the symptom of pain anywhere in the region of the head or neck

Table 2. Symptoms from the intent of the highlighted node and their medical definitions

On the generated context that can be seen in Figure 1, we have highlighted a concept having the extent {`Chronic Otitis Media, Otosclerosis, Chronic Pharyngitis,`

[2] `https://fca-tools-bundle.com/`

Chronic Sinusitis and Deviated Septum} and the intent {Autophony, Ear Fullness, Hearing Loss, Headache}. When looking at a node, the extent of the corresponding concept contains all the objects from the lattice reachable when going (only) downward. Similarly, the intent of the corresponding concept contains all the attributes from the lattice reachable when going (only) upward. Tables 1 and 2 show a detailed description of the extent and intent of the highlighted formal concept.

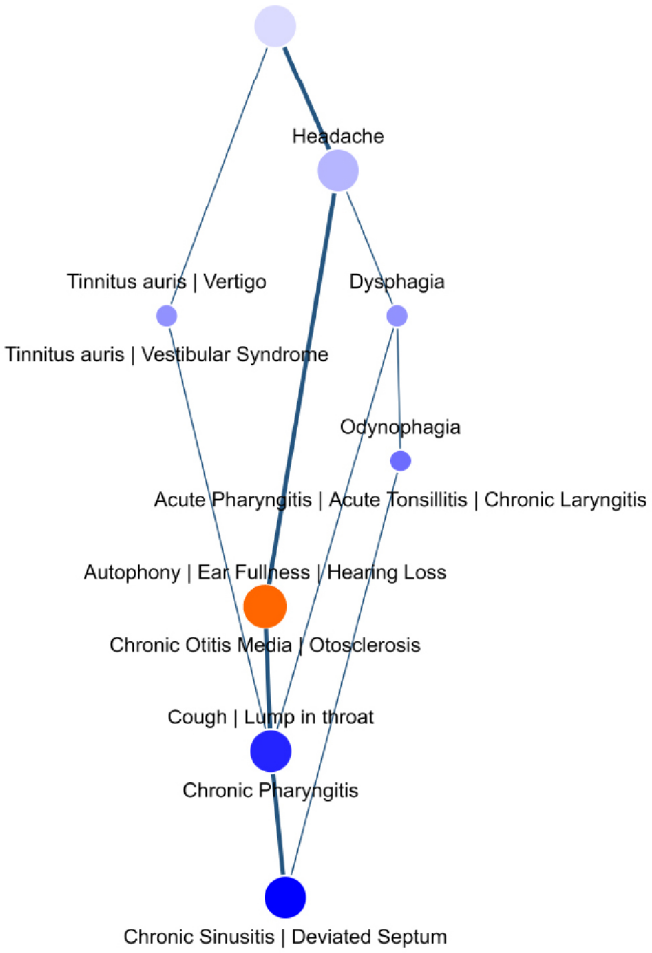


Figure 1. Deviated Septum and Chronic Sinusitis as secondary diagnostics in relation to corresponding symptoms: revealed diagnostics after investigations - LATTICE

The highlighted concept shows that all five diagnostics {Chronic Otitis Media, Otosclerosis, Chronic Pharyngitis, Chronic Sinusitis and Deviated Septum} have a set of common symptoms that need to be taken into consideration: {Autophony, Ear Fullness, Hearing Loss, Headache}. At the same time we can nottice in the lattice that Chronic Otitis Media and Otosclerosis have the same symptoms, which makes the diagnosis difficult. However, three of the diagnostics, namely Chronic Pharyngitis, Chronic Sinusitis and Deviated Septum differentiate themselves by some additional symptoms, such as Cough, Lump in throat, and a few others that can be read from the lattice for each concept.

Therefore, Figure 1 highlights all the diagnostics that a physician should consider when treating a patient, together with a full list of symptoms (either principal or secondary) which may appear.

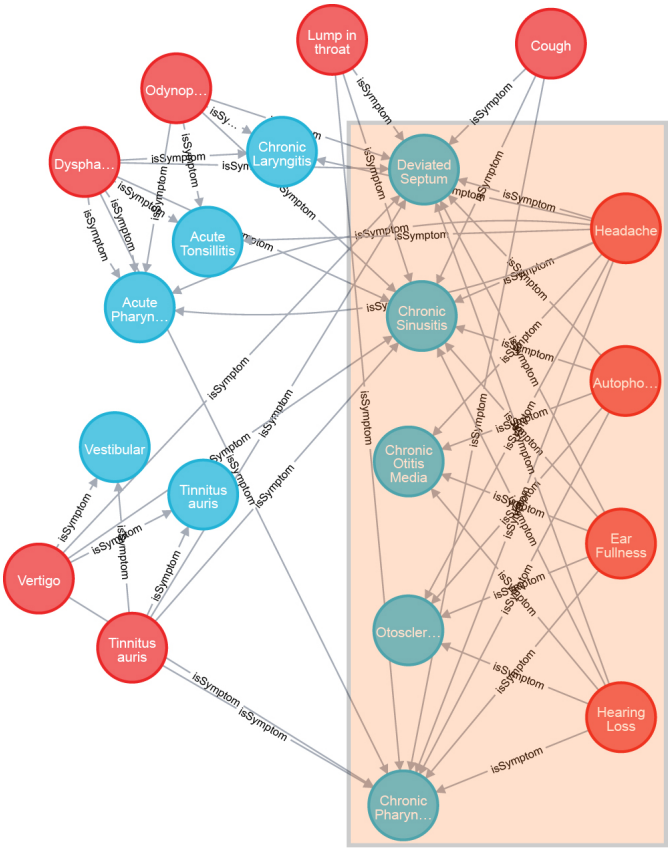


Figure 2. Deviated Septum and Chronic Sinusitis as secondary diagnostics in relation to corresponding symptoms: revealed diagnostics after investigations - NEO4j

In order to gain more information about symptoms and diagnostics, we switch our perspective to a different one, by choosing graph databases. We have processed and stored our medical datasets in the Neo4j graph database. Our purpose was to enrich our knowledge about the medical data, while analyzing different connections between data in the form of correlated nodes. The nodes of the graph correspond to the objects and attributes from the formal concept, while the binary relation is modeled as the directed edges from the graph database. Let us observe that in a formal concept the binary relation is not directed, meaning that saying an object has an attribute or that an attribute belongs to the object is exactly the same thing. However, it does not make any sense to clutter the graph by adding two type of relationships, one from the object to the attribute, and one the other way around. Therefore, we chose to add a single relationship, which in this case can be considered as unordered edges with respect to the graph properties.

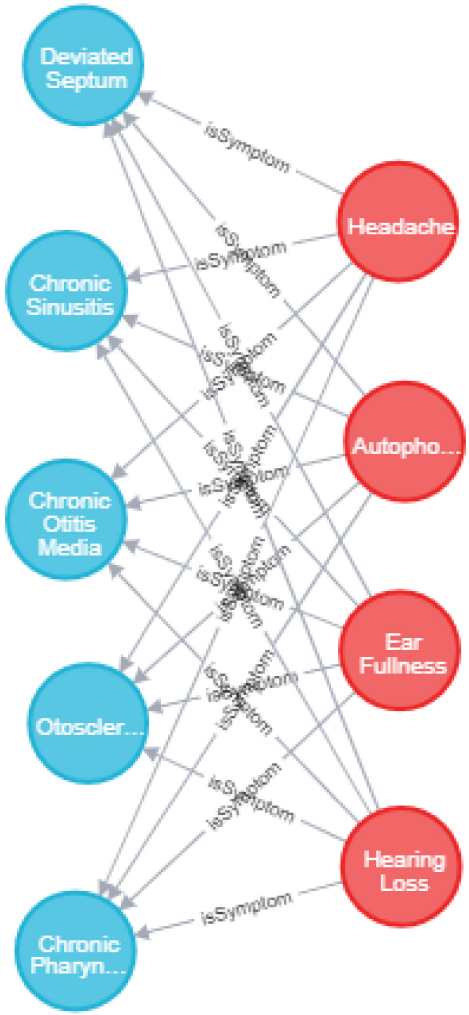


Figure 3. Deviated Septum and Chronic Sinusitis as secondary diagnostics in relation to corresponding symptoms: revealed diagnostics after investigations - NEO4j - zoom only on the concept

In Figure 2 nodes colored in blue are Diagnostics, while nodes colored in red are Symptoms. In this particular case, the relation between Symptoms and Diagnostics is represented with an arrow labeled with `isSymptom`. By following the arrows, i.e. the relationships between different nodes, we can find out different correlations hidden in the medical dataset. Considering the orange highlighted concept presented in Figure 1, we have looked at the graphical representation to identify the same pattern in the generated data graph. Figure 2 presents the same filtered medical dataset where the nodes corresponding to the highlighted concept from Figure 1 are the ones highlighted in the rectangle. This shows how formal contexts and data graphs can be correlated. In this case, we can read the extent and the intent of the corresponding formal concept directly from the graph. We observe that there is no other red node, i.e. symptom, which is in relation to all five diagnostic nodes. Similarly, one can see that no other blue node, i.e. diagnostic, is in relation to all four symptoms identified. Basically we can imagine that a formal concept corresponds to a specific type of strongly connected component in the graph, where there is a relation between all pairs of nodes, with the property that the nodes are of different types, i.e. one is a diagnostic and one is a symptom (obviously it wouldn't make sense to have the relationship "is symptom of" between two diagnostics or between two symptoms).

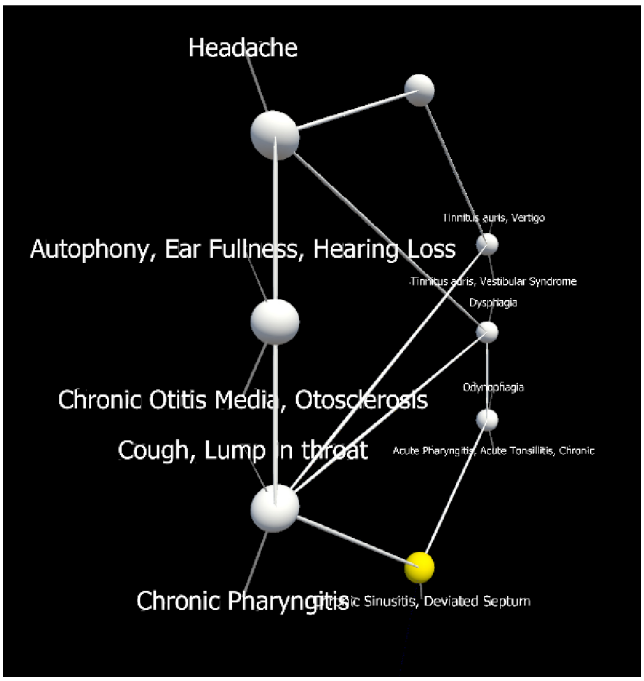


Figure 4. Deviated Septum and Chronic Sinusitis as secondary diagnostics in relation to corresponding symptoms: revealed diagnostics after investigations -3D visualisation of the lattice

Although some data graphs seem a bit hard to read with all the relationships represented, in practice one can choose to exclude or include certain vertices in order to focus on the aspects of interest. Therefore, if we want to analyze the relationship between Symptoms and Diagnostics which are related to `Deviated Septum` and `Chronic Sinusitis` we can choose to exclude nodes which are not connected to all symptoms and diagnostics of interest. In that way, in each of the presented data graphs, we chose to visualize certain relationships between diagnostics and symptoms of the patients. Figure 3 presents the graph containing only the elements corresponding to the formal concept, after choosing to exclude the nodes which are not of interest for this particular example.

By comparing the two representations, the concept lattice obtained with FCA and the data graph obtained with `Neo4j`, we can observe an important advantage of the graph database approach, namely the quantitative information of the clusters which can be easily observed in the data graph, while it is not straightforward in the concept lattice.

When analyzing medical data, interesting facts might stand out, such as rare connections between symptoms and diagnostics. Facts that stand out like this should be analyzed on patients' records over a large period of time and, if they persist, it can lead to the formulation of some hypotheses which can then be researched in more details by medical staff. For instance it would be important to know if there are diagnostics with very similar symptoms, especially if the diagnostics correspond to different medical departments. In that case doctors can be alerted that there is a high chance of a misplaced diagnostic and that, before making the treatment decision, they should consult a doctor with a different specialization in order to exclude diagnostics

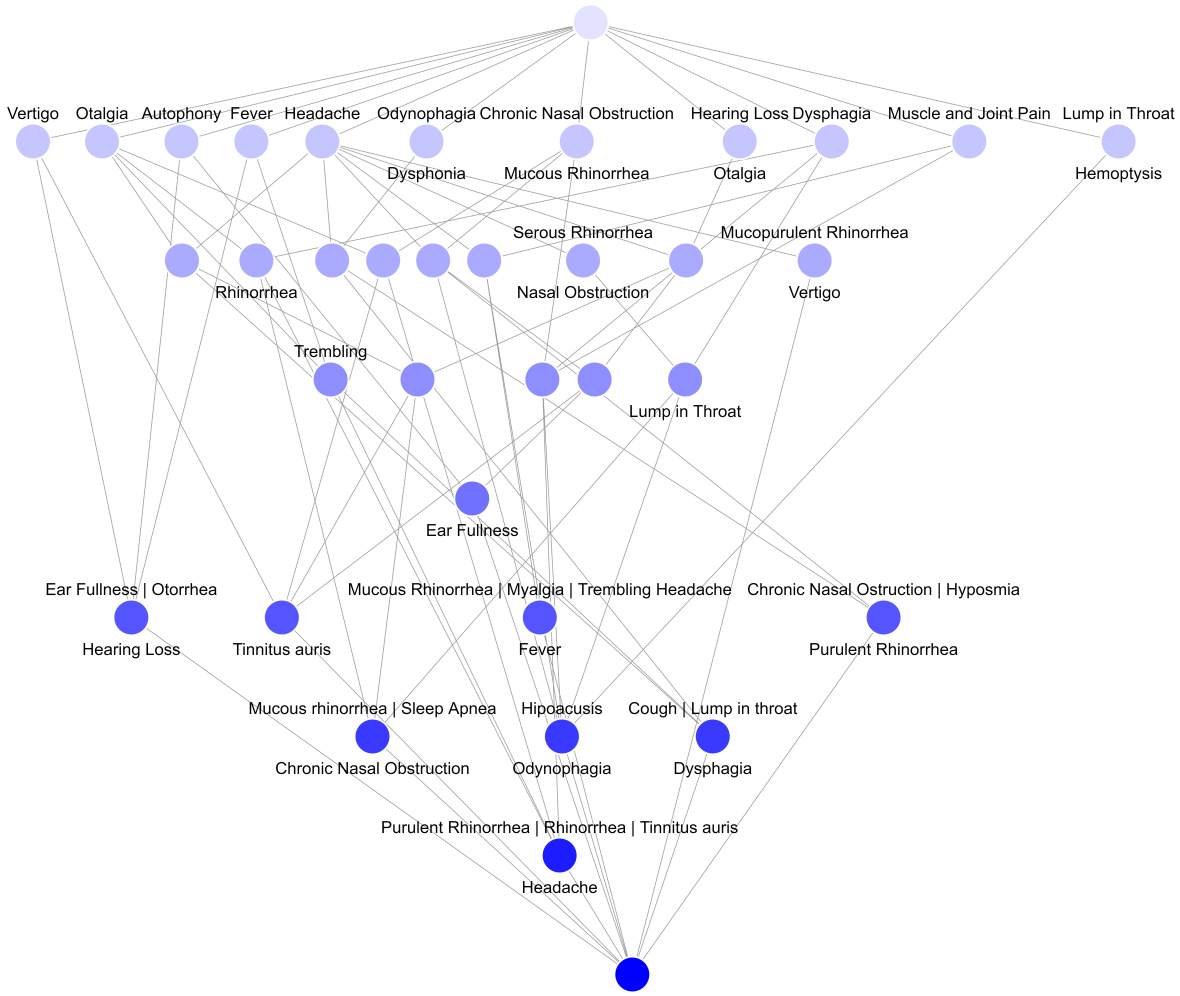


Figure 5. The relation between Principal and Secondary Symptoms - LATTICE

with similar symptoms.

We are especially interested in finding new correlations between symptoms in order to understand which is the cause that led to the diagnostics. For that reason we have represented in Figure 5 the concept lattice, considering principal symptoms as objects and secondary symptoms as attributes. By analyzing the obtained results, doctors can visualize correlations between their patients and coordinate treatment or analyze the differences between them and potential disease progressions.

Due to the fact that usually medical datasets consist of a huge amount of data, visualizing patterns or discovering knowledge is not always an easy task. With the development of new technologies and game engines, the modern graphic capabilities of these technologies increased dramatically. Therefore, we propose a novel approach of navigating through a concept lattice by combining the effectiveness of conceptual scaling with virtual reality. The tool that we have implemented for this purpose is `TOSCANA goes 3D` [20], which allows us to visualize concept lattices by using HTC VIVE HR headsets. As far as we know, this is the first time when knowledge discovery in medical data is enhanced with a virtual reality perspective.

The concept lattices are represented in 3D by using a circular cone like view of the nodes which are at the same depth in the lattice. While exploring the lattice there is the possibility to ”move” around the lattice (teleport or fly) in order to view the lattice from different perspectives or be closer to some nodes (i.e. fly to some node). Moreover, one can choose to rotate the lattice or move the nodes around. This movement options implemented in `TOSCANA goes 3D` offer an important advantage for focusing on a desired concept or analyzing a formal concept through its extent or intent. These might give a valuable perspective over the relations between diagnostics and symptoms and how they are correlated. Moreover, the hidden information extracted in the dyadic case or in the graph based visualization, might be further analyzed in connection with other information found in the analyzed dataset.

Figure 4 shows a printscreen from the 3D visualization of the same concept lattice represented in Figure 1, namely `Deviated`

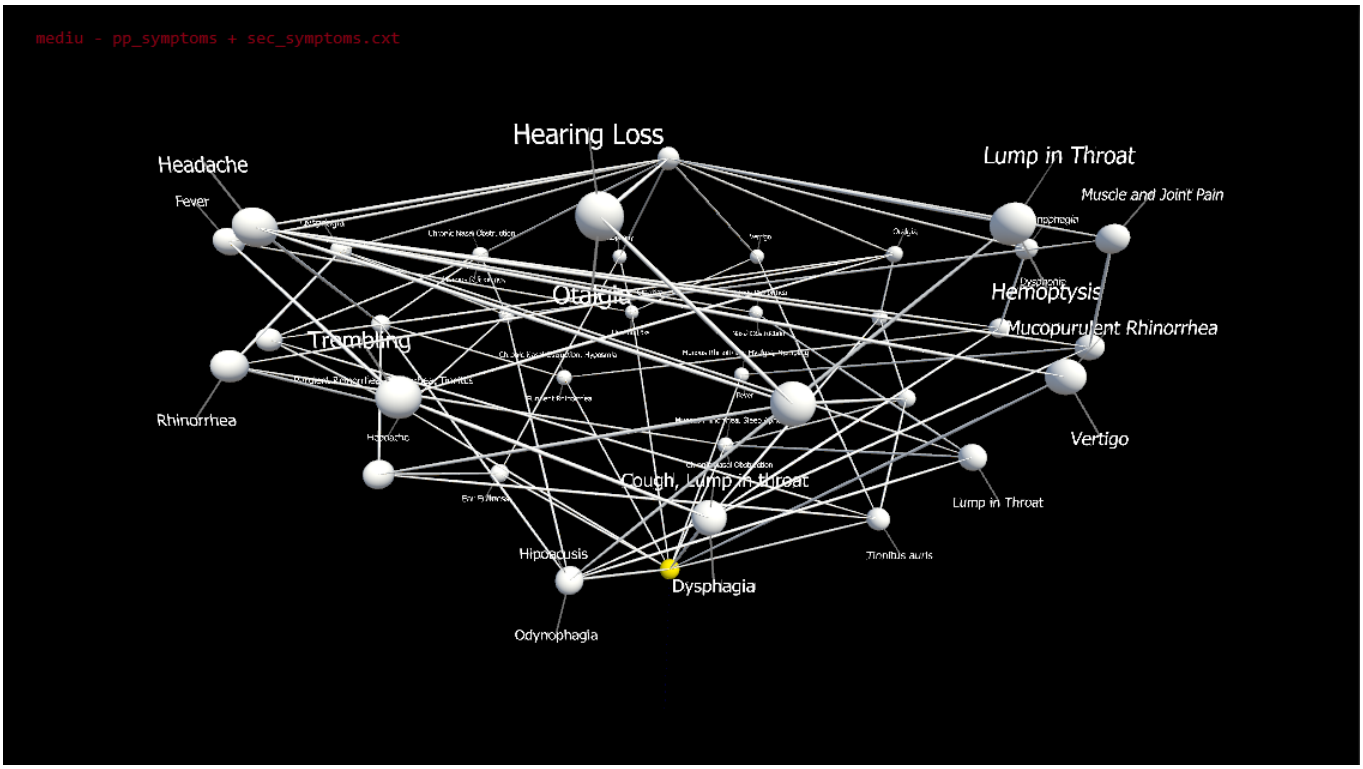


Figure 6. The relation between Principal and Secondary Symptoms -3D visualisation of the lattice

`Septum` and `Chronic Sinusitis` as secondary diagnostics in relation to corresponding symptoms. However, such a flat visualization of the 3D lattice is not conclusive and might seem hard to read, but the whole point of the 3D representation is to be ”inside” the lattice, where you can see all the nodes and navigate among them. The difference between the 3D representation and the 2D representation of a lattice is that in 2D we represent the concepts and their links, so that they do not intersect in the two-dimensional space. In a three-dimensional space the representation looks completely different, since it tries to avoid intersections in the three-dimensional space. Therefore, in 3D it is possible that, looking from one perspective one can see a lot of line intersections in the diagram, while shifting the perspective might give you a clear view of the lattice structure. The corresponding 3D lattice of Figure 5 is represented in Figure 6. We can see that the ”flat” image shown in Figure 6 is not conclusive and seem hardly readable, but, in order to give the reader a feeling of the 3D navigation, a recording can be found following this link[3].

5 Conclusions

Knowledge Discovery based on FCA and graph databases proves to be a valuable pattern extraction and visualization method which can be used in various ways outside of the scientific community. Switching from 2D to 3D, for instance, makes a more detailed navigation through these conceptual structures possible. This might not be very clear while looking at the 2D variant of a 3D structure, but zooming in, flying around, rotating and teleport are impressive new methods to navigate, explore or evaluate knowledge patterns. On the other hand, using Neo4j to explore graph databases proves that graph based knowledge representation can be used as a complementary exploration method.

Further work will focus on further developing of the 3D capabilities of our approach, including also temporal data and analyzing triadic datasets, i.e., datasets comprising objects, which have properties under some certain conditions. Then, the visualization methods developed will be validated and evaluated by experts of the field. Finally, with the help of experts we can compare our methods with other approaches.

[3] `http://www.cs.ubbcluj.ro/~fca/toscana-goes-3d/`

ACKNOWLEDGEMENTS

The present work has received financial support through the project: Entrepreneurship for innovation through doctoral and postdoctoral research, POCU/360/6/13/123886 co-financed by the European Social Fund, through the Operational Program for Human Capital 2014-2020.

REFERENCES

[1] Alexandru Brad, Luciana Neamţu, Silvia Răusanu, and Christian Săcărea, ‘Conceptual knowledge processing grounded logical information system for oncological databases’, in *Proceedings of the International Conference on Knowledge Engineering, Principles and Techniques, KEPT 2011, Cluj-Napoca, Romania, July 4-6, 2011*. Studia Informatica - Issue no. 2 / 2011, (2011).

[2] Dursun Delen, Glenn Walker, and Amit Kadam, 'Predicting breast cancer survivability: a comparison of three data mining methods', *Artif. Intell. Medicine*, **34**(2), 113–127, (2005).

[3] Kenichiro Fujita, Onishi Katsumi, Tadamasa Takemura, and Tomohiro Kuroda, 'The improvement of the electronic health record user experience by screen design principles', *J. Medical Systems*, **44**(1), 21, (2020).

[4] Bernhard Ganter and Rudolf Wille, *Formal Concept Analysis - Mathematical Foundations*, Springer, 1999.

[5] Anamika Gupta, Naveen Kumar, and Vasudha Bhatnagar, 'Analysis of medical data using data mining and formal concept analysis', *International Journal of Medical and Health Sciences*, **1**(11), 591 – 594, (2007).

[6] Diana Halita and Christian Sacarea, 'Is FCA suitable to improve electronic health record systems?', in *24th International Conference on Software, Telecommunications and Computer Networks, SoftCOM 2016, Split, Croatia, September 22-24, 2016*, pp. 1–5. IEEE, (2016).

[7] Nicolas Jay, François Kohler, and Amedeo Napoli, 'Using formal concept analysis for mining and interpreting patient flows within a healthcare network', in *Concept Lattices and Their Applications, Fourth International Conference, CLA 2006, Tunis, Tunisia, October 30 - November 1, 2006, Selected Papers*, eds., Sadok Ben Yahia, Engelbert Mephu Nguifo, and Radim Belohlávek, volume 4923 of *Lecture Notes in Computer Science*, pp. 263–268. Springer, (2006).

[8] Chris Kemper, *Beginning Neo4j*, Apress,, 2015.

[9] Levente Lorand Kis, Christian Sacarea, and Diana-Florina Sotropa, 'Visualizing conceptual structures using FCA tools bundle', in *Graph-Based Representation and Reasoning - 23rd International Conference on Conceptual Structures, ICCS 2018, Edinburgh, UK, June 20-22, 2018, Proceedings*, eds., Peter Chapman, Dominik Endres, and Nathalie Pernelle, volume 10872 of *Lecture Notes in Computer Science*, pp. 193–196. Springer, (2018).

[10] Levente Lorand Kis, Christian Sacarea, and Diana Troanca, 'FCA tools bundle - A tool that enables dyadic and triadic conceptual navigation', in *Proceedings of the 5th International Workshop "What can FCA do for Artificial Intelligence"? co-located with the European Conference on Artificial Intelligence, FCA4AI collocated with ECAI 2016, The Hague, the Netherlands, August 30, 2016*, eds., Sergei O. Kuznetsov, Amedeo Napoli, and Sebastian Rudolph, volume 1703 of *CEUR Workshop Proceedings*, pp. 42–50, (2016).

[11] Mark Needham and Amy Hodler E., *Graph Algorithms: Practical Examples in Apache Spark and Neo4j*, O'Reilly Media, Inc., 2019.

[12] Telung Pan, Yunchun Tsai, and Kowting Fang, 'Using formal concept analysis to design and improve multidisciplinary clinical processes', *WSEAS Transactions on Information Science and Applications*, **5**(6), 880–890, (2008).

[13] Jonas Poelmans, Dmitry I. Ignatov, Sergei O. Kuznetsov, and Guido Dedene, 'Formal concept analysis in knowledge processing: A survey on applications', *Expert Syst. Appl.*, **40**(16), 6538–6560, (2013).

[14] Ian Robinson, Jim Webber, and Emil Eifrem, *Graph Databases: New Opportunities for Connected Data*, O'Reilly Media, Inc., 2nd edn., 2015.

[15] Christian Sacarea, 'Investigating oncological databases using conceptual landscapes', in *Graph-Based Representation and Reasoning - 21st International Conference on Conceptual Structures, ICCS 2014, Iaşi, Romania, July 27-30, 2014, Proceedings*, eds., Nathalie Hernandez, Robert Jäschke, and Madalina Croitoru, volume 8577 of *Lecture Notes in Computer Science*, pp. 299–304. Springer, (2014).

[16] Christian Sacarea, Silviu Boldeanu, Luciana Neamtiu, and Cezara Pastrav, 'Investigating adverse drug reactions using conceptual landscapes', in *IEEE International Conference on Intelligent Computer Communication and Processing, ICCP 2011, Cluj-Napoca, Romania, August 25-27, 2011*, pp. 41–48. IEEE, (2011).

[17] Christian Sacarea, Diana-Florina Sotropa, and Diana Troanca, 'Symptoms investigation by means of formal concept analysis for enhancing medical diagnoses', in *25th International Conference on Software, Telecommunications and Computer Networks, SoftCOM 2017, Split, Croatia, September 21-23, 2017*, eds., Dinko Begusic, Nikola Rozic, Josko Radic, and Matko Saric, pp. 1–5. IEEE, (2017).

[18] Christian Sacarea, Diana-Florina Sotropa, and Diana Troanca, 'Formal concept analysis grounded knowledge discovery in electronic health record systems', in *20th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing, SYNASC 2018, Timisoara, Romania, September 20-23, 2018*, pp. 266–271. IEEE, (2018).

[19] Christian Sacarea, Diana-Florina Sotropa, and Diana Troanca, 'Using analogical complexes to improve human reasoning and decision making in electronic health record systems', in *Graph-Based Representation and Reasoning - 23rd International Conference on Conceptual Structures, ICCS 2018, Edinburgh, UK, June 20-22, 2018, Proceedings*, eds., Peter Chapman, Dominik Endres, and Nathalie Pernelle, volume 10872 of *Lecture Notes in Computer Science*, pp. 9–23. Springer, (2018).

[20] Christian Sacarea, Diana-Florina Sotropa, and Raul-Robert Zavaczki, 'Toscana goes 3d: Using VR to explore life tracks', in *Supplementary Proceedings of ICFCA 2019 Conference and Workshops, Frankfurt, Germany, June 25-28, 2019*, eds., Diana Cristea, Florence Le Ber, Rokia Missaoui, Léonard Kwuida, and Baris Sertkaya, volume 2378 of *CEUR Workshop Proceedings*, pp. 82–87. CEUR-WS.org, (2019).

[21] Douglas West, *Introduction to Graph Theory*, Pearson Education, Inc., 2nd edn., 2001.

[22] Rudolf Wille, *Classification in the Information Age: Proceedings of the 22nd Annual GfKl Conference, Dresden, March 4–6, 1998*, chapter Conceptual Landscapes of Knowledge: A Pragmatic Paradigm for Knowledge Processing, 344–356, Springer Berlin Heidelberg, Berlin, Heidelberg, 1999.

[23] Hamed Majidi Zolbanin, Dursun Delen, and Amir Hassan Zadeh, 'Predicting overall survivability in comorbidity of cancers: A data mining approach', *Decis. Support Syst.*, **74**, 150–161, (2015).

A General Neural Architecture for Carbohydrate and Bolus Recommendations in Type 1 Diabetes Management

Jeremy Beauchamp and **Razvan Bunescu** and **Cindy Marling**[1]

Abstract. People with type 1 diabetes must constantly monitor their blood glucose levels and take actions to keep them from getting either too high or too low. Having a snack will raise blood glucose levels; however, the amount of carbohydrates that should be consumed to reach a target level depends on the recent history of blood glucose levels, meals, boluses, and the basal rate of insulin. Conversely, to lower the blood glucose level, one can administer a bolus of insulin; however, determining the right amount of insulin in the bolus can be cognitively demanding, as it depends on similar contextual factors. In this paper, we show that a generic neural architecture previously used for blood glucose prediction in a *what-if* scenario can be converted to make either carbohydrate or bolus recommendations. Initial experimental evaluations on the task of predicting carbohydrate amounts necessary to reach a target blood glucose level demonstrate the feasibility and potential of this general approach.

1 Introduction and Motivation

Type 1 diabetes is a disease in which the pancreas fails to produce insulin, which is required for blood sugar to be absorbed into cells. Without it, that blood sugar remains in the bloodstream, leading to high blood glucose levels (BGLs). In order to manage type 1 diabetes, insulin must be administered via an external source, such as injections or an insulin pump. People with type 1 diabetes also need to monitor their BGLs closely throughout the day by testing the blood acquired through fingersticks and/or by using a continuous glucose monitoring (CGM) system. If the BGL gets too high (hyperglycemia) or too low (hypoglycemia), the individual responds by eating, taking insulin, or taking some other action to help get their BGL back to within a healthy range. An issue with this, however, is that the person with diabetes must *react* to their BGL, whereas, ideally, they would be able to *proactively* control their BGL. There has been much work in the area of BGL prediction in the past ([1] and [8] for example) with the aim of enabling preemptive actions to manage BGLs before individuals experience the negative symptoms of hypoglycemia or hyperglycemia. However, individuals still need to figure out how much to eat, how much insulin to take, and what other actions they can take to prevent hypoglycemia or hyperglycemia.

The broad goal of the research presented in this paper is to essentially reverse the blood glucose prediction problem, and instead predict how many carbohydrates an individual should eat or how much insulin to administer with a bolus in order to get their BGL to the desired target. We have previously introduced in [6] an LSTM-based neural architecture that was trained such that it could answer *what-if* questions of the type "What will my BGL be in 60 minutes if I eat a snack with 30 carbs 10 minutes from now". We show that by using the BGL target as a feature and the carbohydrates or insulin as labels, a similar architecture can be trained instead to predict the number of carbohydrates that need to be consumed or the amount of insulin that needs to be delivered during the prediction window in order to reach that BGL target.

The work by Mougiakakou and Nikita [7] represents one of the first attempts to use neural networks for recommending insulin regimens and dosages. Bolus calculators were introduced as early as 2003 [11], wherein a standard formula is used to calculate the amount of bolus insulin based on parameters such as carbohydrate intake, carbohydrate-to-insulin ratio, insulin on board, and target BGL. Walsh et al. [10] discuss major sources of errors and potential targets for improvement, such as utilizing the massive quantities of clinical data being collected by bolus advisors. As observed by Cappon et al. in [2], the standard formula approach ignores potentially useful preprandial conditions, such as the glucose rate of change. A feed-forward fully connected neural network was then proposed to exploit CGM information and some easily accessible patient parameters, with experimental evaluations on simulated data showing a small but statistically significant improvement in the blood glucose risk index. Simulated data is also used by Sun et al. in [9], where a basal-bolus advisor is trained using reinforcement learning in order to provide personalized suggestions to people with type 1 diabetes under multiple injections therapy.

The data-driven architecture proposed in this paper is generic in the sense that it can be trained to make recommendations about any variable that can impact BG levels, in particular carbohydrates and insulin. The task of making carbohydrate recommendations is potentially useful in scenarios where patients want to prevent hypoglycemia well in advance, or where a person is interested in achieving a relatively higher target BGL in preparation for an exercise event that is expected to lower it.

As a first step, in this paper we approach the problem of making carbohydrate recommendations. The rest of this paper is organized in the following way: Section 2 provides a more detailed description of the problem. Section 3 describes the model as well as the baselines used to compare against. Section 4 describes the dataset that is used and some of the features of the data. Section 5 discusses some of the training techniques and methods used as well as the results of the experiments that motivated the use of these techniques. Section 6 contains the conclusion and some plans for future work.

2 Three Carbohydrate Recommendation Scenarios

We assume that blood glucose levels are measured at 5 minute intervals through a CGM system. We also assume that discrete deliveries of insulin (boluses) and continuous infusions of insulin (basal rates)

[1] Ohio University, USA, email: {jb199113,bunescu,marling}@ohio.edu

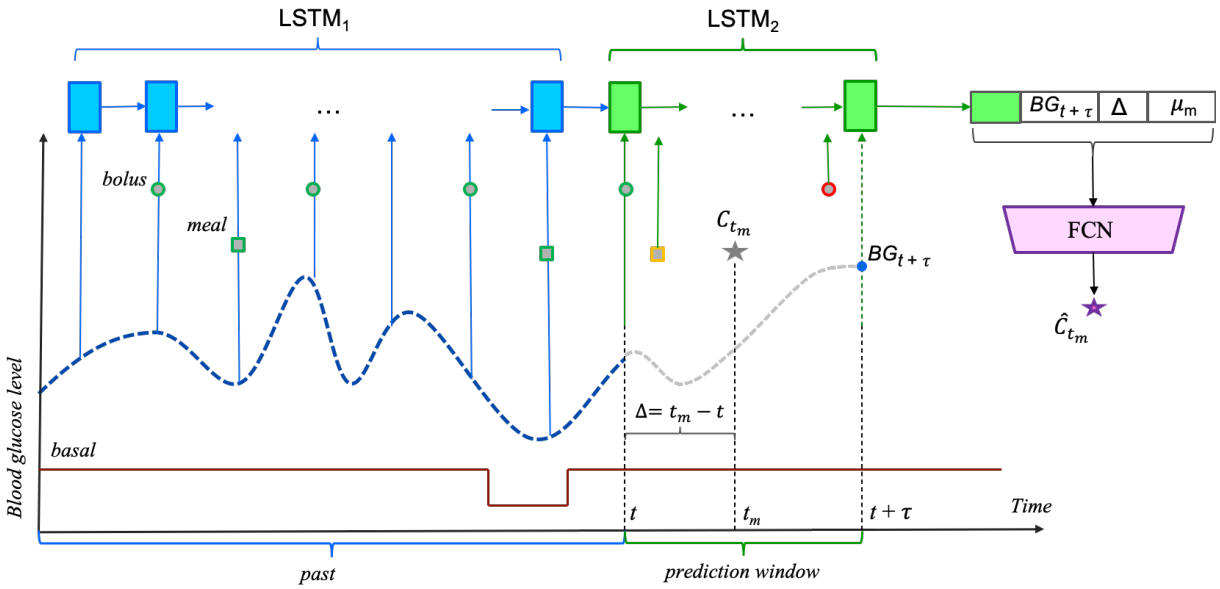


Figure 1. The general neural network architecture for carbohydrate recommendation. The dashed blue line in the graph represents a subject's BGL, while the solid brown line represents the basal rate of insulin. The gray star represents the meal at t_m. The other meals are represented by squares, and boluses are represented by circles. Meals and boluses with a green outline are allowed in all three example scenarios, while those with an orange outline are allowed in scenario S_2 and scenario S_3 examples, and those with a red outline are only allowed in scenario S_3 examples. The blue units in LSTM$_1$ receive input from different time steps in the past. The green units in LSTM$_2$ receive input from the prediction window. The purple trapezoid represents the 5 fully connected layers, whereas the output node at the end computes the carbohydrate prediction.

are recorded. Subjects provide the timing of meals and estimates of the amount of carbohydrates associated with each meal. Given the data available up to the present time t, the problem can formally be defined as predicting the number of grams of carbohydrates (number of carbs) C_{t_m} in a meal that is to be consumed at time $t_m \in [t, t+\tau)$ such that the person's BGL reaches a specified target value $BG_{t+\tau}$ at time $t+\tau$ in the future. Without loss of generality, in this paper we set the *prediction horizon* $\tau = 30$ and 60 minutes. We define three carbohydrate prediction scenarios, depending on whether events such as boluses or other meals happen inside the *prediction window* $[t, t+\tau)$:

1. **Scenario $\mathbf{S_1}$** assumes that there are no events in the prediction window $[t, t+\tau)$. Training a model for this scenario can be difficult due to the scarcity of corresponding training examples, as meals are typically preceded by boluses. The example shown in Figure 1 would be in this scenario if the orange and red outlined meals and boluses were not present.
2. **Scenario $\mathbf{S_2}$** subsumes scenario S_1 by allowing events before the meal, i.e. in the time window $[t, t_m]$. The example that is shown in Figure 1 would be a scenario S_2 example if the bolus outlined in red were not present, and would correspond to answering the following *what-if* question: how many carbs should be consumed at time t_m to achieve the target $BG_{t+\tau}$, if the meal were to be preceded by another meal and a bolus.
3. **Scenario $\mathbf{S_3}$** is the most general and allows events to happen during the entire prediction window $[t, t+\tau)$. The example in Figure 1 is a scenario S_3 example but not a scenario S_1 or scenario S_2 example because of the presence of the orange and red outlined meal and bolus.

We train and evaluate carbohydrate recommendation models for each scenario, using data acquired from 6 subjects with type 1 diabetes [5]. Given the scarcity of training examples for scenario S_1, our starting hypothesis is that models that are trained on examples from scenario S_3 will implicitly learn physiological patterns that will improve performance for the fewer examples in scenario S_1.

3 Baseline Models and Neural Architecture

Given training data containing meals with their corresponding timestamps and carbohydrates, we define the following baselines:

1. **Global average**: The average number of carbs over all of the meals in the subject's training data, μ, are computed and used as the estimate for all future meals, irrespective of context. This is a fairly simple baseline, as it predicts the same value for every example.
2. **ToD average**: In this Time-of-Day (ToD) dependent baseline, an average number of carbs is computed for each of the following five time windows during a day:
   - 12am-6am: μ_1 = early breakfast/late snacks.
   - 6am-10am: μ_2 = breakfast.
   - 10am-2pm: μ_3 = lunch.
   - 2pm-6pm: μ_4 = dinner.
   - 6pm-12am: μ_5 = late dinner/post-dinner snacks.

   The average for each ToD interval is calculated over all of the meals appearing in the corresponding time frame in the subject's training data. At test time, to predict the number of carbs for a meal to be consumed at time t_m, we first determine the ToD interval that contains t_m and output the corresponding ToD average.

Given sufficient historical data, the ToD baseline is expected to perform well for individuals who tend to eat very consistently and have

regular diets. However, it is expected to perform poorly on individuals who have a lot of variation in their diets.

While simple to compute and use at test time, the two baselines are likely to give suboptimal performance, as their predictions ignore the history of BG values, insulin (boluses and basal rates), and meals, all of which could significantly modulate the effect a future meal might have on the BGL. To exploit this information, we propose the general neural network architecture shown in Figure 1. The first component in the architecture is a recurrent neural network (RNN) instantiated using Long Short-Term Memory (LSTM) cells [3], which is run over the previous 6 hours of data, up to the present time t. At each time step (5 minutes), this LSTM network takes as input the BGL, the carbohydrates, and the insulin dosages recorded at that time step. While sufficient for processing data corresponding to scenario S_1, this LSTM cannot be used to process events in the prediction window $[t, t+\tau)$ that may appear in scenarios S_2 and S_3, for which BGL values are not available. Therefore, in these scenarios, the final state computed by the first LSTM model (LSTM_1) at time t is projected and used as the initial state for a second LSTM model (LSTM_2) that is run over the time steps between $(t, t+\tau)$. The final state computed either by LSTM_1 (for scenario S_1) or LSTM_2 (for scenarios S_2 and S_3) is then used as input to a fully connected network (FCN) whose output node computes $\hat{C}_{t_m}$, an estimate of the carbohydrates at time t_m. Besides the LSTM final state, the input to the FCN contains the following additional features:

1. The target BGL at τ minutes into the future, i.e. $BG_{t+\tau}$.
2. The time interval $\Delta = t_m - t$ between the intended meal time and the present.
3. The ToD average computed for Baseline 2 corresponding to the time the meal was eaten.

The entire architecture is trained to minimize the mean squared error between the actual carbohydrates C_{t_m} recorded in the training data and the estimated value $\hat{C}_{t_m}$ computed by the output node of the FCN module. Each LSTM uses vectors of size 100 for the states and gates, whereas the FCN is built with 5 hidden layers, each consisting of 200 ReLU neurons, and one linear output node.

4 Dataset

The data used for the model was collected from 6 subjects with type 1 diabetes [5]. Information including the basal rate of insulin, boluses, meals, and BGL readings was collected over roughly 50 days, although the exact amount of time varies from subject to subject. This time series data is split into three sets, as follows: the last 10 days of data for each subject are used as testing, the previous 10 days are used as validation, and the remainder of the data is used for training.

4.1 From Meal Events to Examples

Since the total number of available examples is directly related to the number of meals, it is useful to know how many meals each subject had. This is shown in Table 1, together with the average number of carbs per meal (Avg), and the corresponding standard deviation (StdDev). Most subjects have a similar average number of carbohydrates in their meals, with the exception of 570 who has a significantly larger number of carbs per meal on average, and more importantly, a much higher standard deviation than the other subjects.

A meal event occurring at time t_m may give rise to multiple examples, depending on the position of t_m in the interval $[t, t+\tau)$. When $\tau = 30$ minutes, an example is created for every possible position of t_m within $[t, t+\tau)$. However, when $\tau = 60$ minutes, an example is created for every position of t_m within $[t, t+30]$, to ensure that there are at least 30 minutes between the meal and the prediction horizon. Table 2 below shows the resulting number of examples for $\tau = 30$ and 60 minutes, in each of the three scenarios. Note that there are fewer examples in scenarios S_1 and S_2 when $\tau = 60$ vs. 30 minutes, despite there being more scenario S_3 examples. This can be explained by the scenarios S_1 and S_2 criteria being even more difficult to meet when $\tau = 60$ minutes, i.e. there cannot be any event within $[t, t+60)$ for S_1, or any event within $[t_m, t+60)$ for S_2.

Table 1. Meal statistics, per subject and total.

		Carbs Per Meal	
Subject	Meals	Avg	StdDev
559	179	36.0	16.0
563	153	29.9	16.3
570	169	105.3	42.0
575	284	40.6	22.9
588	257	30.8	16.6
591	249	31.6	14.2
Total	1291	43.5	33.1

Table 2. Example counts by scenario, for 30 and 60 minutes.

	Scenario S_1		Scenario S_2		Scenario S_3	
Dataset	30	60	30	60	30	60
Training	2396	1923	3889	3491	5096	5931
Validation	629	510	1061	981	1388	1626
Testing	469	339	950	851	1236	1435
Total	3494	2772	5900	5323	7720	8992

5 Experimental Evaluation

The Adam [4] variant of gradient descent is used for training, with the learning rate and mini-batch size being tuned on the validation data. In an effort to avoid overfitting, early stopping with a patience of 5 epochs and dropout with a rate of 10% are used for both models. Interestingly, dropout was found to help the model if it was only applied to the LSTM networks of the model at each time step and not the fully connected network.

Since the overall number of examples available in the dataset is low, the performance was improved by first pretraining a generic model on the combined data from all 6 subjects. Then, for each subject, a new model is initialized with the weights of the generic model, and then fine-tuned on the subject's training data. For each subject, five models were trained with different seedings of the random number generators. We also experimented with fine-tuning models on the union of the training and validation data instead of just the training data. When this combined data is used, the average carb values used in the baselines are recalculated over the union of the training and validation data for each subject.

5.1 Results

The metrics used to evaluate the performance of the models are the root mean squared error (RMSE) and the mean absolute error (MAE), which is less sensitive to large errors. At the end of the training process, there are five fine-tuned models for each subject. The *average* RMSE and MAE of the five models are reported, as well as

the RMSE and MAE of the *best* model. The model that is considered the "best" is the one that had the lowest MAE on the validation data. The results of the five models for each subject are also averaged across all subjects to obtain one overall RMSE and one overall MAE value for the *average model* and the *best model* scores. The baselines are treated much the same, as their RMSE and MAE values are averaged across all subjects to give an RMSE and an MAE score for each baseline.

Table 3 compares the validation results achieved in scenario S_3 by models with and without pretraining for $\tau = 30$ minutes. This experiment clearly shows the benefit of pretraining the models: both the RMSE and MAE are noticeably lower for the pretrained models. As a result, pretraining is always used as part of the training process for both values of τ.

Table 3. Results with and without pretraining, $\tau = 30$.

Setting	RMSE	MAE
Without Pretraining	22.2	15.5
With Pretraining	**20.7**	**14.5**

Table 4 compares models that were fine-tuned on training and validation data with models fine-tuned solely on the training data, in scenario S_3. The results show that the extra examples provided by the validation data proved helpful in improving performance. It is interesting to note that using the combined training-validation data only slightly helped the baselines, but helped the LSTM-based models by a noticeable margin.

Table 4. Fine-tuning on Training vs. Training $\cup$ Validation, $\tau = 30$.

Fine-tuning	Baselines & Models	RMSE	MAE
Training	Global Average	23.3	19.2
	ToD Average	22.5	17.8
	Average Model	21.3	16.0
	Best Model	**20.7**	**15.3**
Training $\cup$ Validation	Global Average	23.1	19.0
	ToD Average	22.2	17.7
	Average Model	20.1	15.0
	Best Model	**19.2**	**14.2**

Table 5 compares the Baselines (Global and ToD averages) with the trained Models (Best and Average) in terms of their RMSE and MAE in the three scenarios.

Table 5. Results for scenarios S_1, S_2, and S_3, for $\tau = 30$ and 60 minutes.

		RMSE		MAE	
	Baselines & Models	30	60	30	60
S_1	Global Average	19.7	18.4	15.7	15.0
	ToD Average	**18.9**	**17.6**	14.8	14.4
	Average Model	19.3	19.5	14.1	**13.9**
	Best Model	19.0	19.8	**13.9**	**13.9**
S_2	Global Average	18.4	17.1	14.5	13.8
	ToD Average	17.4	15.9	13.1	12.2
	Average Model	16.2	**15.3**	11.9	11.4
	Best Model	**15.8**	15.4	**11.6**	**10.9**
S_3	Global Average	18.5	18.6	14.6	14.7
	ToD Average	17.5	17.6	13.2	13.3
	Average Model	15.7	15.6	11.5	11.3
	Best Model	**15.6**	**14.8**	**11.4**	**10.6**

Overall, the LSTM-based models (Average or Best) had the best RMSE and MAE performance across all three scenarios, with the exception of the RMSE scores for scenario S_1. Compared to the other two scenarios, the LSTM models and the baselines have a lower performance in S_1. The decline in performance is even more apparent for the LSTM models, which cannot beat the time-dependent baseline in terms of RMSE for both the 30 minute and 60 minute prediction horizons. This can be explained by the limited number of examples for scenario S_1: since there are so few testing examples in this scenario per subject, one bad prediction can hurt the results significantly, more so for the RMSE than the MAE. Furthermore, the trained models tend to make very similar predictions for all examples stemming from a specific meal, meaning that if the model made a bad prediction for one test example, it likely made a series of similarly bad predictions.

To alleviate the scarcity of training examples in scenario S_1, models trained on S_3 examples, which are the most plentiful and subsume S_1, were evaluated separately on test examples from S_1. This gives an indication on whether any transfer learning is taking place. Table 6 shows the results of this transfer learning experiment, indicating that training on the additional examples from scenario S_3 helps improve performance on scenario S_1 to the level that now the LSTM-based models outperform both baselines.

Table 6. Comparative performance on scenario S_1 test examples: Baselines vs. LSTM-based models trained on S_1 and S_3 examples.

		RMSE on S_1		MAE on S_1	
	Baselines & Models	30	60	30	60
	Global Average	19.7	18.4	15.7	15.0
	ToD Average	18.9	17.6	14.8	14.4
Training on S_1	Average Model	19.3	19.5	14.1	13.9
	Best Model	19.0	19.8	13.9	13.9
Training on S_3	Average Model	**18.2**	17.6	**13.6**	13.3
	Best Model	18.3	**16.7**	13.8	**13.0**

6 Conclusion and Future Work

We introduced a generic neural architecture, composed of two chained LSTMs and a fully connected network, with the purpose of training data-driven models for making recommendations with respect to any type of quantitative events that may impact BG levels, in particular carbohydrate amounts and bolus insulin dosages. Experimental evaluations on the task of carbohydrate recommendations within a 30 or 60 minute prediction window demonstrate the feasibility and potential of the proposed architecture, as well as its ability to benefit from pre-training and transfer learning. Future plans include evaluating carbohydrate recommendations within larger prediction windows, as well as training the architecture for bolus recommendations.

ACKNOWLEDGEMENTS

This work was supported by grant 1R21EB022356 from the National Institutes of Health (NIH). Conversations with Josep Vehi helped shape the research directions presented herein. The contributions of physician collaborators Frank Schwartz, MD, and Amber Healy, DO, are gratefully acknowledged. We would also like to thank the anonymous people with type 1 diabetes who provided their blood glucose, insulin, and meal data.

REFERENCES

[1] R. Bunescu, N. Struble, C. Marling, J. Shubrook, and F. Schwartz, 'Blood glucose level prediction using physiological models and support vector regression', in *Proceedings of the Twelfth International Conference on Machine Learning and Applications (ICMLA)*, pp. 135–140. IEEE Press, (2013).

[2] G. Cappon, M. Vettoretti, F. Marturano, A. Facchinetti, and G. Sparacino, 'A neural-network-based approach to personalize insulin bolus calculation using continuous glucose monitoring', *Journal of Diabetes Science and Technology*, **12**(2), 265–272, (2018).

[3] Sepp Hochreiter and Jürgen Schmidhuber, 'Long short-term memory', *Neural Computation*, **9**, 1735–1780, (12 1997).

[4] D. P. Kingma and J. L. Ba, 'Adam: A method for stochastic optimization', in *Third International Conference for Learning Representations (ICLR)*, San Diego, California, (2015).

[5] C. Marling and R. Bunescu, 'The OhioT1DM dataset for blood glucose level prediction', in *The 3rd International Workshop on Knowledge Discovery in Healthcare Data*, Stockholm, Sweden, (2018). Available at http://ceur-ws.org/Vol-2148/paper09.pdf.

[6] S. Mirshekarian, H. Shen, R. Bunescu, and C. Marling, 'LSTMs and neural attention models for blood glucose prediction: Comparative experiments on real and synthetic data', in *Proceedings of the 41st International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC 2019)*, Berlin, Germany, (2019).

[7] S. G. Mougiakakou and K. S. Nikita, 'A neural network approach for insulin regime and dose adjustment in type 1 diabetes', *Diabetes Technology & Therapeutics*, **2**(3), 381–389, (2000).

[8] K. Plis, R. Bunescu, C. Marling, J. Shubrook, and F. Schwartz, 'A machine learning approach to predicting blood glucose levels for diabetes management', in *Modern Artificial Intelligence for Health Analytics: Papers Presented at the Twenty-Eighth AAAI Conference on Artificial Intelligence*, pp. 35–39. AAAI Press, (2014).

[9] Q. Sun, M. V. Jankovic, J. Budzinski, B. Moore, P. Diem, C. Stettler, and S. G. Mougiakakou, 'A dual mode adaptive basal-bolus advisor based on reinforcement learning', *IEEE Journal of Biomedical and Health Informatics*, **23**(6), 2633–2641, (2019).

[10] J. Walsh, R. Roberts, T. S. Bailey, and L. Heinemann, 'Bolus advisors: Sources of error, targets for improvement', *Journal of Diabetes Science and Technology*, **12**(1), 190–198, (2018).

[11] H. C. Zisser, L. T. Robinson, W. Bevier, E. Dassau, C. L. Ellingsen, F. J. Doyle, and L. Jovanovic, 'Bolus calculator: A review of four "smart" insulin pumps.', *Diabetes Technology & Therapeutics*, **10**(6), 441–444, (2008).

Region Proposal Network for Lung Nodule Detection and Segmentation

Mohammad Hesam Hesamian[1], Wenjing Jia, Xiangjian He, Paul Kennedy

Abstract. Lung nodule detection and segmentation play a critical role in detecting and determining the stage of lung cancer. This paper proposes a two-stage segmentation method which is capable of improving the accuracy of detecting and segmentation of lung nodules from 2D CT images. The first stage of our approach proposes multiple regions, potentially containing the tumour, and the second stage performs the pixel-level segmentation from the resultant regions. Moreover, we propose an adaptive weighting loss to effectively address the issue of class imbalance in lung CT image segmentation. We evaluate our proposed solution on a widely adopted benchmark dataset of LIDC. We have achieved a promising result of 92.78% for average DCS that puts our method among the top lung nodule segmentation methods.

key words: Nodule segmentation, Deep learning, Region proposal network

1 Introduction

Lung cancer, as one of the deadliest cancers, is responsible for major cancer deaths worldwide [16]. Early detection and accurate classification of the lung nodules plays a significant role in increasing the survival rate of the patients. Manual process of detection and segmentation of lung nodule is a challenging task which requires lots of time, proficiency and yet various types of errors may occur. With the increasing growth in the availability of medical images such as computed tomography (CT) scans, automatic nodule detection and segmentation has become a reliable tool to help the radiologist in their tough task of lung image analysis.

Recently, convolutional neural networks (CNNs) have shown the capability to effectively extract image features for successful pattern detection and segmentation across a variety of situations from scene to medical images [18, 3, 4]. Similarly, deep learning approaches have been used for various tasks of medical image analysis, including organ detection, lesion classification and tumour segmentation [13, 6, 7]. Among all those applications, lung nodule segmentation is known to be a challenging task due to the heterogeneous appearance of the lung tumour and also the great similarity between tumour and non-tumour substances in the lung area.

Another severe challenge in medical image segmentation is to deal with the class imbalance [20]. In a fully annotated CT image of the lung, the area occupied by tumour is much smaller than the rest of the lung. This is due to the sparse distribution of pulmonary nodules in the lung. This issue gets more severe when we are performing a semantic segmentation task in which each pixel is considered as one sample. In such a case, the number of samples (pixels) corresponding to the tumour is significantly lower than the rest of the lung area. The class imbalance issue affects the fully convolutional networks [23] more than others. Moreover, the ratio of tumour pixels to background pixels significantly varies from sample to sample. To address this problem, we propose an adaptive weighted loss to alleviate the class imbalance issue at the sample level.

The main contributions of this paper can be summarized in three folds. First, we propose a two-stage network to reduce the dependency on dense sliding window searching for accurate segmentation of the lung nodules. Second, we propose an adaptive weighted loss function to address the class imbalance issue to improve segmentation accuracy. Third, we perform a far distant transfer learning strategy in which the weights are transferred from a general object detection model.

2 Related works

Two major categories of studies have explored lung image segmentation. The first one is the whole lung segmentation, and the second is the lung tumour segmentation. Whole lung segmentation aims to distinguish the border of the entire lung from the rest of the elements appearing in a cardiac CT image [2, 5]. It helps to determine the size and shape of the lung. It is also often used as the first stage for the lung tumour segmentation with the purpose of reducing the false positive cases caused by non-lung areas of CT image.

Recently, many CAD systems based on deep learning are proposed for automatic lung cancer detection. For example, ZNET [24] employed U-Net fully convolutional network architecture for candidate selection on axial slices. For the subsequent false positive reduction, three orthogonal slices of each candidate were fed to the same wide residual network. Wang et al. in [22] proposed a model that can capture a set of nodule-sensitive features from CT images. The 2D branch of the model learns multi-scale 2D features from 2D patches. In the 3D branch, a novel central pooling layer helped the model to select the features around the target voxels effectively. In another study, a region CNN (R-CNN) is proposed for lung nodule segmentation from 3D lung patches [23]. In this model, Deep Active Self-paced Learning (DASL) was introduced to reduce the dependency of the network to fully annotated data. It utilized unannotated samples by taking to account both the knowledge know before training and the knowledge made during the training. Jiang et al. [12] proposed a residually connected multiple resolution network, which was able to combine the features in various resolution inputs simultaneously. Images with different resolution were passed through two separate residual networks, and the extracted features were refined and concatenated. This technique helped them to improve the localization

[1] School of Electrical and Data Engineering, University of Technology Sydney. Email: mh.hesamian@gmail.com

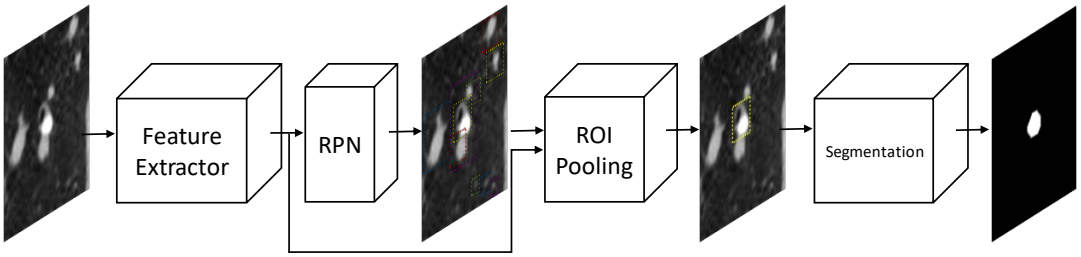


Figure 1. The proposed network model

and negative effect of multiple pooling operation.

Generally, 3D models have a huge demand for memory and high processing cost. Due to these limitations, the algorithms that can be implemented on 3D image analysis are restricted. Moreover, CT scans usually have different slice thicknesses which are not recommended to be treated uniformly in a 3D model [24]. Moreover, 2D models require fewer resources for training and also not affected by the slice thickness.

In two-stage detections methods, region proposal networks (RPN) attracted lots of attention. RPN was first introduced to general object detection tasks by [19]. In the proposed structure, there is a classifier which determines the probability of having a target object at the anchor. Then the regression regresses the coordinates of the proposal. The candidature multiple anchors and sharing the features make the RPN efficient in time and detection. Employing the bounding box regression enables the RPN to produce more precise proposals. RPN has many successful variation and application[17, 15] yet it has not been explored adequately in lung nodule segmentation task.

According to all shortcomings of the current methods mentioned above, and the enormous potential of the RPN networks, we were motivated to design and develop a model to combine the benefits of two-stage detection and segmentation models with higher segmentation accuracy.

3 The Proposed Method

There are several image modalities, which can be used for lung abnormality detection such as PET, SPECT, X-Ray, MRI and CT. PET and SPECT are mainly utilized for metabolism characterization. Therefore, they can unveil the functional abnormalities in an organ. CT, X-Ray and MRI are structural image modalities which are able to hand out anatomical information about the organ. The simplicity and lower price of CT scan have changed it to primary image modality for cancer detection and screening [25]. For this study, we used the benchmark dataset of LIDC-IDRI [1], consisting of 1024 chest CT scans. Each case is associated with an XML file, denoting the boundary pixels of the tumour, marked by four radiologists. For this study, the nodules with size $>$ 3mm are selected. These scans are broken down to slices and converted to JPG format in the original size.

3.1 Network Structure

There are some two-stage nodule detection systems, where in the first stage the nodule candidates are detected, and in the second stage, the false positive is reduced [24]. But in our approach, inspired by the development of general object detection [8], we propose a two-stage method capable of nodule detection and segmentation. The model segments the elements of CT scan into two classes of tumour and background. The general building block of the network is presented in Fig. 1. The first section of the network is the feature extractor in which general feature maps are created. The ResNet101 [9] is used as the backbone of the system. The residual connections of the ResNet allow using deeper structures without facing the gradient vanishing problem.

The extracted feature maps are then passed to the RPN. By this method, the convolution layers are reuses. Thus, it saves lots of computations. This module produces several candidate regions, containing the potential malignant tumours. Since our model has only two classes of tumour and background, we limit the number of proposed regions to 300. Having a limited number of regions will help to reduce the possibility of false positive as well as speed up the training process. Each of the candidate bounding boxes is then given to the ROI pooling. At this stage, the network evaluates the proposed regions according to intersection over union (IoU) value. The IoU is set to 0.5 to reduce the false positive. It means a fixed number of the ROIs (50) with the IoU more than 0.5 will be passed to the segmentation module for semantic segmentation task. In case of no ROI satisfying the condition, the input sample is considered as a negative sample. The last block of the network will produce the segmentation mask for the proposed ROI.

3.2 Adaptive Loss Coefficients

One of the main reasons of accuracy drop in nodule segmentation is the class imbalance of the training samples, due to which, during the training model will not learn equally from all classes. Therefore, one of the main challenges in medical image analysis is how to effectively modify the model to overcome the class imbalance issue and maximize the learning capability of the model.

In the segmentation section of the proposed model, samples will be segmented in the two classes of background and tumour. Thus, we applied a binary cross-entropy function as the segmentation loss. This is shown in Eq. 1.

$$L(S) = -\frac{1}{N}\sum_{n=1}^{N}\left[y_n \log \hat{y_n} + (1-y_n)\log(1-\hat{y_n})\right] \quad (1)$$

In this equation, y and $\hat{y}$ represent the ground truth and the predicted value of each pixel, and N is the total number of the pixels in the given sample of S.

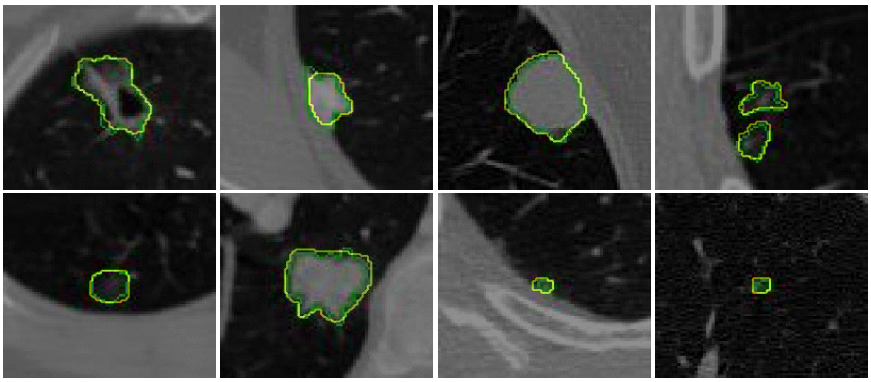


Figure 2. Visualization of the network output. The green color is the ground truth marked by the radiologists. The yellow is the prediction, generated by the proposed model. Best viewed in color.

Then the dynamic loss coefficients of γ_1 and γ_2 are calculated for each of the given ROIs representing the proportion of tumour and background pixels. These coefficients compensate the imbalance between the numbers of pixels in the two classes.

Since in our model, each pixel is considered as one sample, the loss will be calculated at the pixel level. Therefore, the final loss function of Eq. 2 is formed by applying the dynamic loss coefficient to the Eq. 1. The loss is calculated for each pixel of p_i, where p_i denotes the i^{th} pixel of a given flattened ROI.

$$Loss(p_i) = \begin{cases} \gamma_1 \times \Big[- y(p_i) \log \hat{y}(p_i) \Big], & \text{if } p_i \in tumour \\ \\ \gamma_2 \times \Big[- y(p_i) \log(1 - \hat{y}(p_i)) \Big], & \text{if } p_i \in bg \end{cases} \tag{2}$$

The proposed weighted loss updates its coefficients for each ROI proposed by ROI selection module. This process helps to address the class imbalance more accurately because the proportion of tumour to background varies significantly from sample to sample. The common weighting loss such as what proposed in [11] applies a fixed set of the coefficient for all the samples. These coefficients are calculated by counting and averaging the tumour pixel and background pixels over the entire dataset. Application of a fixed set of weight to all the samples does not seem to be an optimal solution while the adaptive weighted loss allows the network to address the class imbalance more effectively within individual samples.

4 Experiments and Discussion

In this section, we experimentally evaluated our solution on the widely used benchmark dataset LIDC [1]. Our model is implemented with the Keras library, and all the experiments are conducted on Linux (REH7.0) with an Nvidia Quadro P5000 GPU of 16 GB memory.

4.1 Data preparation

We evaluated our solution on the publicly available dataset of LIDC [1], which contains 1024 lung scans each annotated by at least four radiologists. All data are selected and used for training and testing of the network with the 5-fold cross validation approach.

Similar to most of the other medical datasets, our data was also imbalanced toward the tumour class. Training a model on such data will lead to learning more features from one class and ignoring the others. As another technique to combat this issue, we have selected a patch-wise training strategy. For this purpose, we have extracted the patches around the tumour and use them as training samples. According to nodule size distribution statistics, 85% of nodules can be covered by a patch of 30×30 (voxels), and 99% of nodules can be covered by a 40×40 (voxels) patch [24]. Hence, we have extracted the patches of 76×76 pixel to ensure almost all of the tumours are covered. At the next stage, we heavily augment our training data by using filipping, rotating, zooming and shrinking the input samples. Data augmentation is reported that data augmentation can also help to improve the performance and robustness of CNNs [10]. This data augmentation serves two purposes of avoiding the overfitting and extending the training dataset size.

4.2 Training

We have employed a two-step full network adaptation strategy to transfer the weights from a far distant source. In this process, the weights are initialized from a pre-trained model trained on COCO [14] dataset for the general object detection task. Transferring the weights from such model and using them for a different task of medical object segmentation, is called 'far transfer learning'. Generally, transfer learning is proven to alleviate the overfitting issue on the small training dataset and improve the convergence speed of the training. Theoretically, transfer learning has better performance when the task of source and target models are more similar [10]. Thus, some believe that far transfer learning may not produce good results [21] but, our achieved results are demonstrating that far transfer learning combined with a careful fine-tuning strategy can deliver competitive results.

As a part of our weight transferring, the weights for the feature extractor layers are initialized from the pre-trained model, and the last layers are initialized randomly. During the first stage of training, all the network weights except for the feature extractor weights are fixed. At this stage, only the feature extractor is trained on the input data for a couple of epochs. The rest of the network weights are injected at the second stage of the training, and all the network layers are trained together afterwards.

Table 1. The detection rate of nodules of various sizes, obtained with our proposed approach.

Size	Measurements (%)			
	IoU=0.3	IoU=0.4	IoU=0.5	IoU= 0.6
Tumour size <10mm	97.54 ± 0.56	95.81 ± 0.67	92.32 ± 1.07	84.76 ± 1.55
10 < Tumour size < 30 mm	98.13 ± 0.72	97.56 ± 0.64	95.57 ± 1.06	92.04 ± 0.95
Tumour size > 30mm	95.91 ± 4.56	94.25 ± 4.21	93.26 ± 4.46	90.16 ± 7.95
Average	97.21 ± 1.95	95.87 ± 1.84	93.73 ± 2.19	88.99 ± 3.78

5-fold cross-validation is employed to validate the results of the testing. The data set is divided into five equal subsets, and each time four of them are used for training, and the one remaining is used for testing. The LIDC dataset has not a separate set of train and test set. Hence, we divided the data into two main sets of train and test for the purpose of 5-fold cross-validation. Moreover, a portion of training data (10%) has already been used for validation at the end of each epoch. By this method, we will ensure that all the samples are used at least once for training and testing.

4.3 Experiment results

Fig. 2 shows the qualitative results of tumours detected and segmented for various tumour types and sizes.

To highlight the performance of our proposed model, we quantitatively evaluate its performance for two tasks, *i.e.,* detection and segmentation, respectively.

4.3.1 Results of tumour detection

Our proposed method performs the detection task through a segmentation approach. This detection method differentiate our model from the pure detection models. For the detection part, we measure the detection accuracy of the tumours under various IoU values of 0.3, 0.4, 0.5 and 0.6. Table 1 shows the results of tumour detection for three tumour size categories. The results are presented with various IoU values to analyse the detection performance clearly. For instance, in the case of $IoU = 0.5$, if the IoU of prediction mask with the ground truth is more than 0.5, the tumour is considered as correctly detected. As expected, the detection accuracy is dramatically increased when the tumour size increased.

The results in Table 1 highlights the significant detection accuracy of the proposed model in detecting the small size tumours.

4.3.2 Results of tumour segmentation

To evaluate the segmentation performance, we measure the Dice score of segmentation (DCS), sensitivity and positive predictive value (PPV) as defined as in Eqs. 3, 4 and 5, respectively, as:

$$DSC = \frac{2TP}{2TP + FP + FN} \tag{3}$$

$$Recall = \frac{TP}{TP + FN} \tag{4}$$

and

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

TP and FP refer to true positive and false positive while TN and FN denote the true negative and false negative.

The results presented in the Table 2 show that the proposed method is able to perform the challenging task of lung nodule segmentation with higher accuracy than the state-of-the-art methods. The dice score of our segmentation model is almost 10% higher than the listed methods. Similarly, the precision of our method is much higher than all the listed methods. In case of the recall, we got slightly (less than 1 percent) lower than CF-CNN, but CF-CNN produced wider standard deviation. It shows that our model was more stable throughout all the input cases.

4.3.3 Discussion

The main reason for this improvement is the structure of the network. In the proposed structure, at the first stage, some ROIs are extracted as the potential tumour. Then, the most tumour-like ROIs are selected through the ROI pooling module. Then the second stage performs the segmentation on the selected ROIs. This two-step strategy eliminates the necessity to scan the entire input image via a sliding window that leads to creating the prediction at every position. By performing these two steps, many of the nodule-like patterns which may confuse the model are eliminated. Therefore, the accuracy of segmentation is improved by only focusing on the more relevant tumour features. Moreover, the training and testing speed will increase as there would be no full input search via the sliding window.

The second reason for the improved accuracy is the application of novel adaptive loss in training of the segmentation module. This adaptive loss helps to address the class imbalance issue of the medical images within the individual sample. By application of a dynamic pair of class weight coefficients, the model can derive more balanced features from each sample.

The transfer learning strategy used for training of the model helped the model to be more stable and prevent it from overfitting. Moreover, one of the reasons for achieving a smaller standard deviation value compared with other studies could be the application of transfer learning.

5 Conclusion

We have proposed a two-stage framework for accurate segmentation of the lung nodules. The first stage of the network provides some potential nodule areas. The second stage accurately segments the selected ROIs. To effectively address the class imbalance issue of the small organ segmentation, we proposed an adaptive weighted loss function, where the weight coefficients are calculated per sample. This approach leads to extract more accurate features and therefore, more precise segmentation of the input. The model was tested on the

Table 2. Quantitative evaluation of the achieved results and comparison. Models marked with * use 3D processing.

Methods	Measurements		
	Dice (%)	Recall (%)	Precision (%)
Nodule R-CNN* [23]	64 ± 0.44	-	-
Hesamian et al. [11]	81.24 ± 1.24	-	79.75 ± 4.08
CF-CNN* [22]	80.47 ± 10.76	92.75 ± 12.83	75.84 ± 13.14
Jiang et al. [12]	68 ± 0.23	85 ± 0.13	67 ± 0.22
Proposed method	92.78 ± 0.1	92.31 ± 0.27	93.17 ± 0.18

publicly available dataset of LIDC and was able to deliver an average detection accuracy of 93.73% (IoU = 0.5) and average dice score of 92.78% for the segmentation part. At last, the achieved results demonstrate that a far distant transfer learning with careful weight initialization can perform competitive results.

REFERENCES

[1] Samuel G Armato III, Geoffrey McLennan, Luc Bidaut, Michael F McNitt-Gray, Charles R Meyer, Anthony P Reeves, Binsheng Zhao, Denise R Aberle, Claudia I Henschke, Eric A Hoffman, et al., 'The lung image database consortium LIDC and image database resource initiative (idri): a completed reference database of lung nodules on ct scans', *Medical physics*, **38**(2), 915–931, (2011).

[2] Geng Chen, Dehui Xiang, Bin Zhang, Haihong Tian, Xiaoling Yang, Fei Shi, Weifang Zhu, Bei Tian, and Xinjian Chen, 'Automatic pathological lung segmentation in low-dose ct image using eigenspace sparse shape composition', *IEEE transactions on medical imaging*, (2019).

[3] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille, 'Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs', *IEEE transactions on pattern analysis and machine intelligence*, **40**(4), 834–848, (2018).

[4] Noel CF Codella, David Gutman, M Emre Celebi, Brian Helba, Michael A Marchetti, Stephen W Dusza, Aadi Kalloo, Konstantinos Liopyris, Nabin Mishra, Harald Kittler, et al., 'Skin lesion analysis toward melanoma detection: A challenge at the 2017 international symposium on biomedical imaging ISBI, hosted by the international skin imaging collaboration ISIC', in *Biomedical Imaging (ISBI 2018), 2018 IEEE 15th International Symposium on*, pp. 168–172. IEEE, (2018).

[5] Yu Gordienko, Peng Gang, Jiang Hui, Wei Zeng, Yu Kochura, O Alienin, O Rokovyi, and S Stirenko, 'Deep learning with lung segmentation and bone shadow exclusion techniques for chest X-Ray analysis of lung cancer', in *International Conference on Theory and Applications of Fuzzy Systems and Soft Computing*, pp. 638–647. Springer, (2018).

[6] Nazia Hameed, Antesar M Shabut, Miltu K Ghosh, and M Alamgir Hossain, 'Multi-class multi-level classification algorithm for skin lesions classification using machine learning techniques', *Expert Systems with Applications*, **141**, 112961, (2020).

[7] Sardar Hamidian, Berkman Sahiner, Nicholas Petrick, and Aria Pezeshk, '3D convolutional neural network for automatic detection of lung nodules in chest CT', in *Medical Imaging 2017: Computer-Aided Diagnosis*, volume 10134, p. 1013409. International Society for Optics and Photonics, (2017).

[8] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick, 'Mask r-cnn', in *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969, (2017).

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, 'Deep residual learning for image recognition', in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, (2016).

[10] Mohammad Hesam Hesamian, Wenjing Jia, Xiangjian He, and Paul Kennedy, 'Deep learning techniques for medical image segmentation: Achievements and challenges', *Journal of digital imaging*, 1–15, (2019).

[11] Mohammad Hesam Hesamian, Wenjing Jia, Xiangjian He, and Paul J Kennedy, 'Atrous convolution for binary semantic segmentation of lung nodule', in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1015–1019. IEEE, (2019).

[12] Jue Jiang, Yu-Chi Hu, Chia-Ju Liu, Darragh Halpenny, Matthew D Hellmann, Joseph O Deasy, Gig Mageras, and Harini Veeraraghavan, 'Multiple resolution residually connected feature streams for automatic lung tumor segmentation from ct images', *IEEE transactions on medical imaging*, **38**(1), 134–144, (2018).

[13] Yang Lei, Tonghe Wang, Sibo Tian, Xue Dong, Ashesh B Jani, David Schuster, Walter J Curran, Pretesh Patel, Tian Liu, and Xiaofeng Yang, 'Male pelvic multi-organ segmentation aided by cbct-based synthetic mri', *Physics in Medicine & Biology*, **65**(3), 035013, (2020).

[14] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick, 'Microsoft coco: Common objects in context', in *European conference on computer vision*, pp. 740–755. Springer, (2014).

[15] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg, 'Ssd: Single shot multibox detector', in *European conference on computer vision*, pp. 21–37. Springer, (2016).

[16] Kimberly D Miller, Ann Goding Sauer, Ana P Ortiz, Stacey A Fedewa, Paulo S Pinheiro, Guillermo Tortolero-Luna, Dinorah Martinez-Tyson, Ahmedin Jemal, and Rebecca L Siegel, 'Cancer statistics for hispanics/latinos, 2018', *CA: a cancer journal for clinicians*, **68**(6), 425–445, (2018).

[17] Joseph Redmon and Ali Farhadi, 'Yolo9000: better, faster, stronger', in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7263–7271, (2017).

[18] Joseph Redmon and Ali Farhadi, 'Yolov3: An incremental improvement', *arXiv preprint arXiv:1804.02767*, (2018).

[19] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun, 'Faster r-cnn: Towards real-time object detection with region proposal networks', in *Advances in neural information processing systems*, pp. 91–99, (2015).

[20] Skylar Stolte and Ruogu Fang, 'A survey on medical image analysis in diabetic retinopathy', *Medical Image Analysis*, 101742, (2020).

[21] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri, 'Deep end2end voxel2voxel prediction', in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 17–24, (2016).

[22] Shuo Wang, Mu Zhou, Zaiyi Liu, Zhenyu Liu, Dongsheng Gu, Yali Zang, Di Dong, Olivier Gevaert, and Jie Tian, 'Central focused convolutional neural networks: Developing a data-driven model for lung nodule segmentation', *Medical image analysis*, **40**, 172–183, (2017).

[23] Wenzhe Wang, Yifei Lu, Bian Wu, Tingting Chen, Danny Z Chen, and Jian Wu, 'Deep active self-paced learning for accurate pulmonary nodule segmentation', in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 723–731. Springer, (2018).

[24] Hongtao Xie, Dongbao Yang, Nannan Sun, Zhineng Chen, and Yongdong Zhang, 'Automated pulmonary nodule detection in ct images using deep convolutional neural networks', *Pattern Recognition*, **85**, 109–119, (2019).

[25] Junjie Zhang, Yong Xia, Hengfei Cui, and Yanning Zhang, 'Pulmonary nodule detection in medical images: a survey', *Biomedical Signal Processing and Control*, **43**, 138–147, (2018).

In Silico Comparison of Continuous Glucose Monitor Failure Mode Strategies for an Artificial Pancreas

Yunjie (Lisa) Lu [1 2] and **Abigail Koay**[1] and **Michael Mayo**[1]

Abstract. An artificial pancreas is a medical Internet of Things-based system consisting of a continuous glucose monitor, an insulin pump, and a micro-controller. The use of artificial pancreas systems is becoming increasingly popular amongst patients with type 1 diabetes due to its effective ability to allow the patient better control of his/her own blood glucose levels compared to other more standard treatments. In this paper, the problem of missing sensor readings in the glucose monitor data is considered. How should the micro-controller (which adjusts the insulin pump based on monitor readings) react when the glucose monitor stops transmitting for an unpredictable period of time? A strategy that answers this question is called a failure mode strategy. In this paper, several potential failure mode strategies are explored in the context of simulation experiments. Results are presented showing that at least one effective and simple failure mode strategy (0.5μ & LR$<$72) exists.

1 INTRODUCTION

An artificial pancreas (AP) [10] is a real-time, closed-loop insulin delivery system for patients with type 1 diabetes (T1D). It consists of three components: a continuous glucose monitor (CGM) [3], a controller, and an insulin pump. The CGM estimates the wearer's blood glucose level (BGL) by sampling the interstitial fluid in the subcutaneous tissue beneath the skin; a small needle-like sensor that is applied usually to the abdomen or the upper arm is used for this purpose. Readings from the sensor are taken at set intervals, typically every five minutes, and are sent wirelessly to the controller. The controller in turn adjusts the rate of delivery of insulin (or insulin dose size) during the next interval on the basis of CGM readings [8].

In this work, the problem of missing data in the CGM trace is considered [2]. Missing data is a potentially significant problem for an AP because the controller may not have been designed to deal with this situation, and the default failure mode strategy may be to simply turn the pump off or revert back to a preset basal rate [1] [8]. This can lead to risks if the user is unaware of the problem. For example, a reduced insulin dosage at meal times may lead to periods of severe hyperglycemia, increasing the risk of long-term complications, for example, heart attack and kidney damage. On the other hand, if the pump keeps running while the sensor is off, it could potentially overdose the patient on insulin leading severe hypoglycemia such as coma or seizure, which can be more life-threatening than hyperglycemia. Additionally, non-severe hypoglycemia can lead to discomfort, such as anxiety or blurred vision.

Gaps in real CGM traces occur for a variety of reasons. In most cases, they are benign (e.g. the sensor is no longer accurate and needs to be replaced) but there is also the possibility of deliberate malicious interference (for example, [9] illustrate how the wireless connection between CGM and controller can be jammed).

As mentioned, a simple default strategy if connectivity to the sensor is lost is to simply turn the insulin pump off or switch to a preset basal rate. However, this is unsatisfactory for the reasons mentioned above. An alternative idea is to have the controller replace the missing CGM readings with estimates, and then use these for the insulin dose calculations so that the decisions on insulin delivery rate can still be made. Unfortunately, there is currently no effective method for dealing with missing CGM readings in real-time [12], as most effective time series forecasting (or replacement) methods require readings from the future.

Therefore in this paper approaches based on changing the behaviour of the insulin pump when the sensor is down are considered. It is shown via a set of experiments involving virtual patients and a state-of-the-art AP controller that viable alternative failure mode strategies that can replace the simplistic "pump off" strategy exist. Note that imputation of missing glucose values is not a focus here; instead we are concerning with how the pump should behave when the sensor is unavailable.

2 BACKGROUND

To perform the comparison of failure mode strategies, a virtual patient simulator along with a modern AP controller based on fuzzy logic was used.

2.1 Virtual Patient Simulator

The simulator utilised in this work is an open-source T1D patient simulator Simglucose [11]. The simulator is a re-implementation of the 2008 version of UVA/Padova T1D simulator described by Dalla Man et al. [5] which has been approved by the Food and Drug Administration (FDA) for pre-clinical trials of specific insulin treatments such as control algorithms for an AP. In brief, the simulator models all components of a patient and AP system. There are several choices for CGM type, each of which come with appropriate sensor error models and reading limits. Additionally, several different types of simulated insulin pump are available.

The virtual patient model used by the simulator describes the glucose/insulin subsystem and is modelled using compartments. Important processes in the virtual patient simulator include a model of the gastrointestinal tract, from which the rate of appearance of glucose in the glucose subsystem is determined during simulated meals; the insulin subsystem, which models both the rate of appearance of insulin, its rate of degradation, and its interactions with the liver and

[1] Department of Computer Science, University of Waikato, New Zealand
[2] contact author email: yl606@students.waikato.ac.nz

body tissues; and the production and storage of glucose in the liver and its utilisation in muscle and adipose tissue. Each virtual patient in the simulator is generated randomly from a joint probability distribution over tens of physiological parameters, and ten different adult patients are currently available in the simulator.

The simulator models meals as well, and the timing and size of meals in terms of carbohydrate (CHO) count are also randomly generated from an appropriate probability distribution.

Figure 1 shows a trace of the simulator's output for one virtual patient. The figure shows CGM readings, actual BGL (not always identical due to sensor lag and error), and the insulin pump rate. The normoglycemia range (see Table 1) is also shown.

2.2 Fuzzy Logic Controller

The controller component of an AP serves the function of monitoring (and aggregating) incoming CGM sensor trace data, and then using this information to make making insulin pump rate adjustment decisions in real time (for example, to reduce the risk of hypoglycemia induced by a rapid decline in BGL).

Many algorithms have been proposed for solving this control problem, and they can be broadly grouped into control theory-based approaches such as proportional-integral-derivative (PID) controllers; logic-based approaches; adaptive statistical based-approaches (e.g. based on moving averages); and machine learning-based approaches [10]. A drawback of several of these approaches is the need to either set parameters which are patient-specific (e.g. a PID controller has at least three important constants to set) or to run the controller for a period of time in order to train a predictive model.

In order to circumvent these issues (mostly), we implemented the fuzzy logic controller (FLC) proposed by Mauseth et al. [6] and recently updated [7]. The FLC encapsulates expert knowledge about insulin dosing in the form of fuzzy logic rules. The idea is that the fuzzy rules comprise knowledge about what an expert diabetes clinician would do in a given situation where an insulin dose needs to be set. Since it is a knowledge-based approach, the controller is not overly dependent on setting constants or training data. Furthermore, the inputs to the controller are straightforward: the current BGL is the first input; the BGL velocity (change since the last reading) is the second input; and the BGL acceleration (how the velocity is changing) is the third input.

These three inputs are then "fuzzified" (mapped to fuzzy sets), and following that a fuzzy lookup table containing the expert knowledge is consulted for optimal dose calculation. In general, the table is designed such that higher accelerations and velocities lead to higher doses, while low absolute BGL values and strongly negative velocities switch the insulin pump off.

The FLC does have a single patient-specific parameter, the "personalisation factor" (PF), which can range from zero to ten. The PF dictates the aggressiveness of the insulin treatment, with a value of ten corresponding to a very weak treatment (all doses being multiplied by a factor of 0.002), while a PF value of zero is the strongest treatment (all doses scaled by 1.74). Most patients should be expected to have a PF of around five, indicating a scaling factor of 0.92 of the dose computed by the FLC.

3 METHOD

In this section, the experimental setup, in particular the methods by which the PF values are set on a patient-wise basis, and how artificial missing data gaps were generated in the simulated CGM trace are described.

3.1 Optimised PF value

Each individual has significantly different insulin sensitivity. Some are more sensitive to insulin, so their blood glucose levels drop rapidly in response to the same insulin dose than those who are less sensitive. Thus, the FLC's PF value is necessarily unique for each individual.

To determine the optimal PF value for each adult virtual patient in the T1D simulator, the following procedure was followed: for each possible PF value, the simulator was run for ten virtual days at a sampling interval of five minutes. The amount of time spent in normoglycemia for each patient and each PF value respectively was recorded. The PF value for each adult virtual patient which maximised the time spent in normoglycemia was then selected.

3.2 Missing Data Generation

One guideline concerning proper usage of CGM [4] state that about 70-80% coverage of sensor reading coverage is required over two weeks for calculating metrics. Assuming this is the usual case, then 336 hours of sensor use over two weeks can be expected, leaving 68 hours of total CGM sensor gap over two weeks, or 4.8 hours per day on average. Additionally, gaps are usually not scattered random individual sensor readings, but are likely to occur contiguously in blocks since events such as faults tend to last for significant periods of time. On the basis of this assumption, we implemented a method to generate missing data gaps in the CGM trace. Essentially, the program generates a random start time (between midnight to 1900 hours, including mealtime) for each gap as well as a random length for the gap (between 0.5 hours and five hours). See Figure 1 for an example of the trace with gaps.

3.3 Failure Mode Strategies

In this section, five different failure model strategies are described. As a baseline, results for an "ideal" set of simulations in which there are no sensor trace gaps are also included. The failure mode strategies are as follows:

1. Pump off
   This strategy simply switches the insulin pump off when sensor gaps are detected.
2. μ
   μ is defined as the average value of the total insulin dose from the previous one hour. Since there are twelve readings per hour in an AP with a five minute reading interval, the average dose is defined as
   $$\mu = \frac{1}{12} \sum_{n=1}^{12} d_i \tag{1}$$
   where d_i is the insulin dose i adjustments previously. The μ strategy operates the pump at a constant rate of μ.
3. 0.5μ
   On the basis that μ may produce doses that are sometimes too high, the 0.5μ strategy is calculated in the same way, except that the constant dose size is halved.

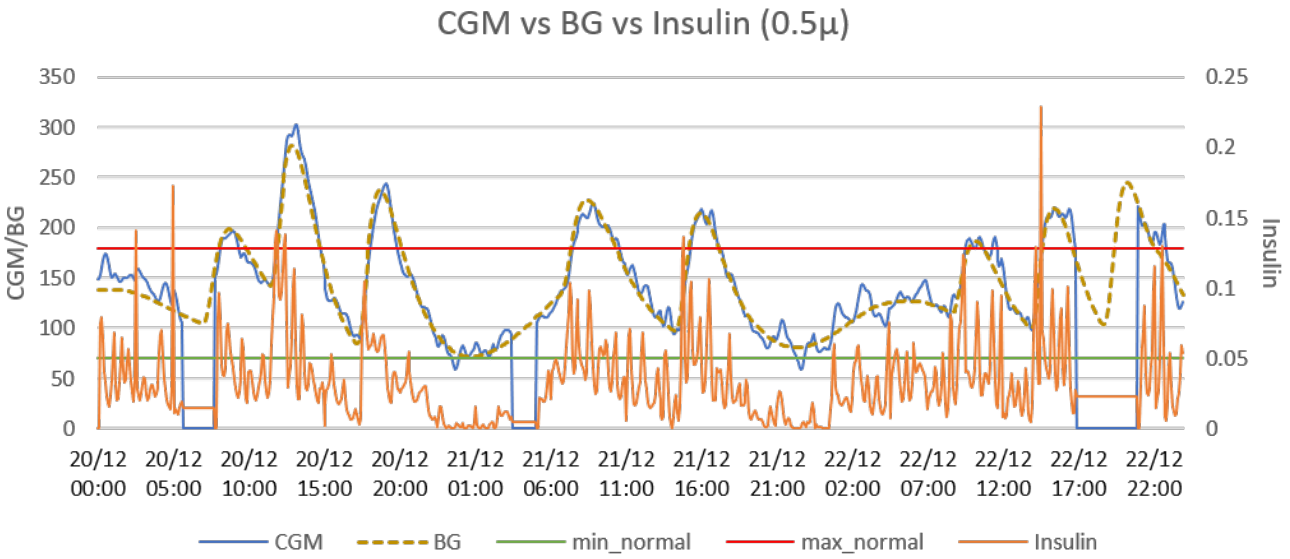


Figure 1: Example of data generated by running Simglucose for one virtual patient for 72 hours by using 0.5μ method. Gaps are shown as periods of zeros in the CGM trace. Not shown here is the CHO intake that primarily drives the fluctuations in BGL.

4. Random choice (Rand_choice)
   Random choice constructs a method that randomly chooses a value from the last one hour of insulin dosages, on the basis that varying the dose size might be beneficial compared to the μ strategies which computes constant doses. The expected value of the random choice strategy equivalent to the dose as calculated by μ.
5. Hybrid method (0.5μ & LR<72)
   Following initial results showing that the 0.5μ strategy was effective, a new strategy was defined that extended 0.5μ with a constraint: if the last reading (LR) before the sensor is down is smaller than a preset value (in this case, 72.0 mg/dL), then the pump will be turned off; otherwise, the 0.5μ strategy will be applied. 72 mg/dL was chosen as a threshold because it is just above the L1 hypoglycemia threshold defined in Table 1, and it is desirable that this threshold is not crossed. The equation for this strategy is:

$$\text{dose} = \begin{cases} 0 & \text{if LR} < 72.0 \\ 0.5\mu & \text{otherwise} \end{cases} \quad (2)$$

4 EVALUATION

In each experiments, 100 simulated days per adult virtual patient per failure mode strategy was run. This gave a total of 10×6 runs per patient. Each run took approximately ten minutes on a laptop with a 2.6GHz Intel Core i7-9750 CPU processor. For all experiments, the simulated GuardianRT CGM was used in conjunction with the Insulet insulin pump.

Table 2 shows for each patient with their personalised PF value according to the method of selecting the optimal PF value described in the previous section.

The effectiveness of each failure mode strategy was then determined by calculating the time spent in standard glycemic ranges as defined in Table 1. As mentioned, hypoglycemia can be considerably more dangerous than hyperglycemia, so therefore time spent in the L2 Hypoglycemia range is the most significant statistic in the results.

Figure 2 shows that these five strategies provide similar performance in the L1 Hyperglycemia range. All the median values are quite close to each other (i.e. between 18.21 and 19.44). μ and Rand_choice perform quite well most of the time. The lowest percentage obtained due to Rand_choice is below 5% for one patient. However, 0.5μ and hybrid strategy have a more stable performance

Range	Definition
L2 Hypoglycemia	<54 mg/dL (<3.0 mmol/L)
L1 Hypoglycemia	54 to <70 mg/dL (3.0 to <3.9 mmol/L)
Normoglycemia	70 to 180 mg/dL (3.9 to 10.0 mmol/L)
L1 Hyperglycemia	>180 to 250 mg/dL (>10.0 to 13.9 mmol/L)
L2 Hyperglycemia	>250 mg/dL (>13.9 mmol/L)

Table 1: Definitions of glycemic ranges used by [4].

Adult#	1	2	3	4	5	6	7	8	9	10
PF	6	4	8	6	5	6	7	6	7	6

Table 2: The optimised PF value for each adult virtual patient.

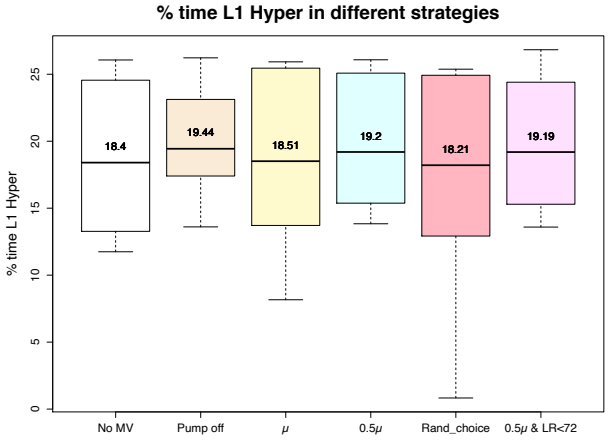


Figure 2: Percentage of time spent in L1 hyperglycemia by failure mode strategy across ten adult virtual patients.

with no significant outliers.

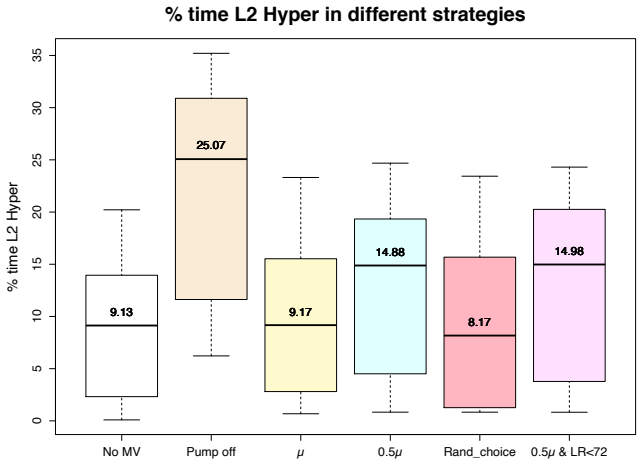


Figure 3: Percentage of time spent in L2 hyperglycemia by failure mode strategy across ten adult virtual patients.

Figure 3 depicts time spend in L2 hyperglycemia by the virtual patients. It can be noticed that because of the high chance of being severe hyperglycemia, the pump off strategy is probably the least ideal failure mode strategy. Two strategies that produce the lowest chance of being L2 Hyperglycemic are μ and Rand_choice.

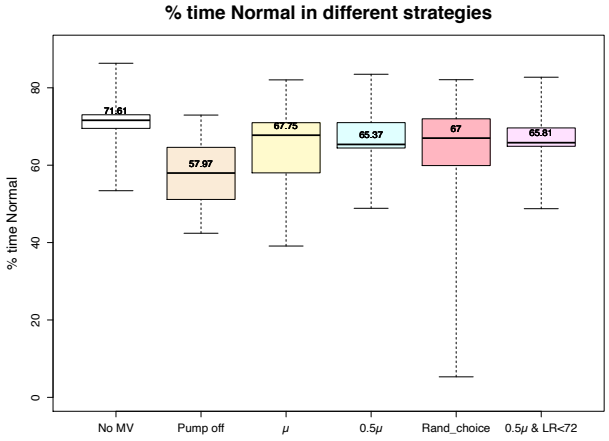


Figure 4: Percentage of time spent in normoglycemia by failure mode strategy across ten adult virtual patients.

Figure 4 depicts time spent in normoglycemia. Unlike the other figures where lower is better, in this case higher is better. The figure indicates that the last four strategies output similar median values which are all quite close to the baseline no missing value strategy. The best two strategies that give us the highest median time are again μ and Rand_choice. However, it is quite noticeable that these two methods, especially Rand_choice, as not stable, as observed before. The percentage time spent in normoglycemia can drop to under 10% for one of these strategies. The 0.5μ & LR$<$72 strategy gives the most stable performance across virtual patients.

Figure 5 and 6 show time spent in the hypoglycemic ranges. From these box plots, it can be observed that both μ and Rand_choice significantly underperform compared to the other strategies. In terms of the more dangerous L2 hypoglycemia range (Figure 6), the pump off and hybrid strategies work best.

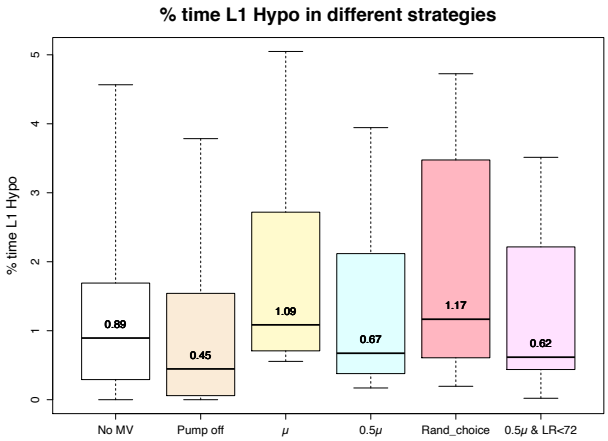


Figure 5: Percentage of time spent in L1 hypoglycemia by failure mode strategy across ten adult virtual patients.

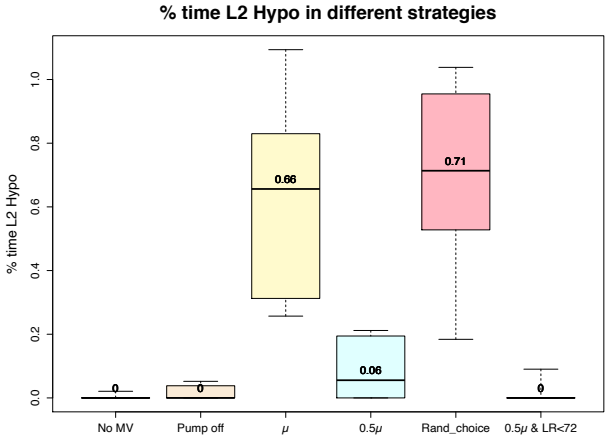


Figure 6: Percentage of time spent in L2 hypoglycemia by failure mode strategy across ten adult virtual patients.

5 CONCLUSION & FUTURE WORK

In this study, it was found that the hybrid method works well and is furthermore simple to implement. In most cases, it was close to the no missing values baseline strategy, especially in terms of time spent in the critical L2 hypoglycemia range.

The hybrid method could therefore be used as a baseline for the development of more sophisticated failure mode strategies (such as those based on online CGM imputation or machine learning methods) in the future. It is also important to test this failure mode strategy with other controllers and on other twenty patients (i.e., ten children and ten adolescent patients) in addition to the FLC and ten adult patients analysed in this paper.

REFERENCES

[1] B. Wayne Bequette, 'Fault detection and safety in closed-loop artificial pancreas systems', *Journal of Diabetes Science and Technology*, **8**(6), 1204–1214, (2014). PMID: 25049365.

[2] Helga Blauw, Patrick Keith-Hynes, Robin Koops, and J. Hans DeVires, 'A review of safety and design requirements of the artificial pancreas', *Annals of biomedical engineering*, **44**(11), 3158–3172, (2016).

[3] Giacomo Cappon, Giada Acciaroli, Martina Vettoretti, Andrea Facchinetti, and Giovanni Sparacino, 'Wearable continuous glucose monitoring sensors: A revolution in diabetes treatment', *Electronics*, **6**(3), 65, (2017).

[4] Thomas Danne, Revital Nimri, Tadej Battelino, Richard M. Bergenstal, Kelly L. Close, J. Hans DeVries, Satish Garg, Lutz Heinemann, Irl Hirsch, Stephanie A. Amiel, Roy Beck, Emanuele Bosi, Bruce Buckingham, Claudio Cobelli, Eyal Dassau, Francis J. Doyle, Simon Heller, Roman Hovorka, Weiping Jia, Tim Jones, Olga Kordonouri, Boris Kovatchev, Aaron Kowalski, Lori Laffel, David Maahs, Helen R. Murphy, Kirsten Nørgaard, Christopher G. Parkin, Eric Renard, Banshi Saboo, Mauro Scharf, William V. Tamborlane, Stuart A. Weinzimer, and Moshe Phillip, 'International consensus on use of continuous glucose monitoring', *Diabetes Care*, **40**(12), 1631–1640, (2017).

[5] Chiara Dalla Man, Francesco Micheletto, Dayu Lv, Marc Breton, Boris Kovatchev, and Claudio Cobelli, 'The UVA/Padova type 1 diabetes simulator: New features', *Journal of Diabetes Science and Technology*, **8**(1), 26–34, (2014). PMID: 24876534.

[6] Richard Mauseth, Irl B. Hirsch, Jennifer Bollyky, Robert Kircher, Don Matheson, Srinath Sanda, and Carla Greenbaum, 'Use of a "fuzzy logic" controller in a closed-loop artificial pancreas', *Diabetes Technology & Therapeutics*, **15**(8), 628–633, (2013). PMID: 23829285.

[7] Richard Mauseth, Sandra M. Lord, Irl B. Hirsch, Robert C. Kircher, Don P. Matheson, and Carla J. Greenbaum, 'Stress testing of an artificial pancreas system with pizza and exercise leads to improvements in the system's fuzzy logic controller', *Journal of Diabetes Science and Technology*, **9**(6), 1253–1259, (2015). PMID: 26370244.

[8] Charrise M. Ramkissoon, B. Wayne Bequette Brian Aufderheide, and Josep Vehi, 'A review of safety and hazards associated with the artificial pancreas', *IEEE reviews in biomedical engineering*, **10**, 44–62, (2017).

[9] Luca Reverberi and David Oswald, 'Breaking (and fixing) a widely used continuous glucose monitoring system', in *11th USENIX Workshop on Offensive Technologies (WOOT 17)*, Vancouver, BC, (August 2017). USENIX Association.

[10] Dawei Shi, Sunil Deshpande, Eyal Dassau, and Francis J. Doyle, 'Chapter 1 - Feedback control algorithms for automated glucose management in t1dm: the state of the art', in *The Artificial Pancreas*, eds., Ricardo S. Sánchez-Peña and Daniel R. Cherñavvsky, 1 – 27, Academic Press, (2019).

[11] Jinyu Xie. Simglucose v0.2.1, https://github.com/jxx123/simglucose, last accessed 28 Feb 2020, 2018.

[12] Sara Zulj, Paulo Carvalho, Rogerio Ribeiro, and Ratko Magjarevic, 'Handling missing data in CGM records', in *Future Trends in Biomedical and Health Informatics and Cybersecurity in Medical Devices*, eds., Kang-Ping Lin, Ratko Magjarevic, and Paulo de Carvalho, pp. 420–427, Cham, (2020). Springer International Publishing.

Towards Causal Knowledge Graphs - Position Paper

Eva Blomqvist[1] and **Marjan Alirezaie**[2] and **Marina Santini**[3]

Abstract. In this position paper, we highlight that being able to analyse the cause-effect relationships for determining the causal status among a set of events is an essential requirement in many contexts and argue that cannot be overlooked when building systems targeting real-world use cases. This is especially true for medical contexts where the understanding of the cause(s) of a symptom, or observation, is of vital importance. However, most approaches purely based on Machine Learning (ML) do not explicitly represent and reason with causal relations, and may therefore mistake correlation for causation. In the paper, we therefore argue for an approach to extract causal relations from text, and represent them in the form of Knowledge Graphs (KG), to empower downstream ML applications, or AI systems in general, with the ability to distinguish correlation from causation and reason with causality in an explicit manner. So far, the bottlenecks in KG creation have been scalability and accuracy of automated methods, hence, we argue that two novel features are required from methods for addressing these challenges, i.e. (i) the use of Knowledge Patterns to guide the KG generation process towards a certain resulting knowledge structure, and (ii) the use of a semantic referee to automatically curate the extracted knowledge. We claim that this will be an important step forward for supporting interpretable AI systems, and integrating ML and knowledge representation approaches, such as KGs, which should also generalise well to other types of relations, apart from causality.

1 Introduction

Knowledge Graphs (KGs) have emerged in the past decade as a prominent form of knowledge representation, frequently used by large enterprises such as Google, Facebook, Amazon, Siemens, and many more [16]. A KG is simply a graph representing some set of data, usually coupled with a way to explicitly represent the meaning of the data, e.g. an ontology. This can be seen as a revival of graph-based knowledge representation, with roots in the early 1970's (for instance, the term knowledge graph was used as early as 1972 by [28]), but with recent advances mainly related to the Semantic Web, such as Linked Data on the Web, and Semantic Web ontologies. This renewed popularity has been accelerated by two main realisations regarding Machine Learning (ML), including Deep Learning (DL) models: Although outperforming humans on many specific tasks, ML/DL methods (i) are often unable to determine the semantics of the correlations found in the data, and (ii) lack the ability to transparently explain a prediction. A particularly challenging example is the case of causal relations. As pointed out by [17] the future development of AI depends on building systems that incorporate the notion of causality, e.g. to allow the system to reason about situations that have not been previously encountered, based on general principles. There is an active field of research developing specific ML/DL algorithms targeting causal learning and reasoning. However, only targeting ML/DL-based causal reasoning does not necessarily improve interpretability, hence there is a need to also develop methods for producing and utilising interpretable causal models, as we shall discuss further in Section 3.

KGs, being symbolic models, allow to define the semantics of relations in data, at the level of formalisation necessary for an intended task, e.g., through ontologies if needed, and by integration with ML/DL methods this supports interpretability of predictions. Hence, KGs can be used to address both the main shortcomings of ML/DL mentioned earlier, but the construction of KGs is a major bottleneck in their adoption, just as was the case with knowledge representation in general, in early AI systems. Outside large companies, such as Google, and huge crowdsourcing initiatives, such as Wikidata [32], it is usually infeasible to construct large scale KGs "manually". Rather, they have to be bootstrapped from existing sources, such as semi-structured data or text. Current KG generation algorithms, however, either do not take into account the desired formalisation of the KG at all, or they hard-code it into the extraction algorithm. An example of the latter is DBPedia [20], which is specific to a Wiki source and results are expressed using a fixed ontology, which means the method does not generalise to new settings or other input structures. Additionally, the quality of the generated KGs is usually poor [11], requiring manual curation, and further, no automated approach so far targets complex relations, e.g. causality. Therefore, it is our goal to specifically target new methods and algorithms for KG generation from text, which (a) explicitly take KG requirements into account, e.g. allowing to flexibly specify the required schema of the output graph, and (b) automate the curation process, to radically improve the quality of resulting KGs. In order to fulfil a specific set of KG requirements, as well as to achieve a sufficient level of accuracy, we propose to use the notion of Knowledge Patterns (KPs) [**?**] as formalisations of KG requirements. A KP represents both a linguistic frame that can be detected in text [2], but also the representation of that frame in the desired KG output formalism, i.e. similar to the notion of Ontology Design Patterns (ODP) [5, 4]. In order to tackle a particularly important obstacle to the future development of the AI field, i.e., considering the importance of causal models and reasoning, we intend to specifically target KPs and KGs targeting complex causal relations.

2 ML - Causality and Interpretability

While ML methods perform very well in learning complex connections between large amounts of input and output data, there is no guarantee that they capture causation (cause and effect relations). This shortcoming stems in part from the ignorance of data-driven

[1] Linköping University, Sweden, email: eva.blomqvist@liu.se
[2] Örebro University, Sweden, email: marjan.alirezaie@oru.se
[3] RISE, Research Institutes of Sweden, email: marina.santini@ri.se

methods with respect to reasoning techniques, which are effortlessly applied by humans. Consider the two imaginary groups of people: Group A: *100 asthmatic people with a death rate of 40%*, and Group B: *100 asthmatic people who also suffer from pneumonia, with a death rate of 35%*. A ML method solely fed with the data can only learn a nonsense result saying: *asthmatics with pneumonia have more chances to live!* [8]. The learning method has perhaps learned the associations (or correlations) among the variables in the data correctly. However, due to the absence of context and common sense knowledge, and also the lack of reasoning abilities, the method has not been able to explicitly and correctly capture the cause-effect relations. That is why the outcome of the example above is not only counterintuitive, but also misleading. By context, we refer to any information that may not be represented in the observed data directly, and may include the actual causes behind the observations, e.g., some set of background information about the setting. In the given example, people in Group B are more high risk patients than those in Group A. The lower death rate of people in Group B can have different reasons, for instance, due to their high risk status they may more likely be taken to the intensive care unit (ICU) or they may be taking more effective medicines, which are all factors (or features) not considered by the learning model [29]. Additionally, some common sense knowledge, such that additional diseases generally increase mortality rather than decreasing it, could have also supported a system in avoiding the erroneous conclusion, e.g., through using knowledge representations as a referee for the learned model [1], as we will discuss furhter later on in this paper.

To provide sufficient support for a reliable and precise prediction or diagnosis process, every prediction made by a system needs to be perfectly transparent and interpretable by the user. This is necessary for any autonomous system to act as the support for humans in making decisions, and even legally and ethically required in many domains, including the medical domain. Although ML should definitely be a part of the solution, what is predicted needs to be interpretable, so that any conclusion based on that knowledge can be explained in detail, most often including some notion of reliability or confidence. A solution to this shortcoming of ML methods is to integrate them with explicitly represented knowledge, such as in the case of causality, a formal causal model that reflects all the possible and existing relations, including cause-effect ones, among the concepts of a given domain.

3 Causal Models

By causal model, we refer to a parametric model that represents a set of probability densities over variables including concepts defined in a system (e.g., diseases and symptoms in the context of medicine), together with the plausible causal relationships between them [31]. Once available, integration of causal information (inferred from a causal model) with the training (observational) data, can enable a ML/DL method to also learn the causes behind its mistakes (i.e., misclassification) [1], and consequently improve its performance. In this paper we specifically target causal *relations*, i.e. the focus is not on determining the probability distributions but rather on the underlying knowledge representation.

Although recent research reflects the considerable impact of causal inference in different domains, such as public health [15] or earth science and climate change [27], it is still also challenging to involve causal models within a learning process. One of the hindering factors is, in fact, the lack of available domain-related causal models compatible with the data used for learning [22], which leads to the need of manually creating such models for each use case. However, for many domains nowadays, such as e-health and patient monitoring through smart homes, both the set of potential outcomes and the set of variables are extremely large. Therefore, manually constructing and maintaining causal models requires a huge effort, and cannot be easily adapted to a new domain. Even further, manual construction of models representing all the environmental features and relations may not even be practically feasible, due to the changing nature of the environment. This has already changed the focus of research to automatically generating causal models [22], which is a line of research we are also contributing to.

Furthermore, causal relations are usually not as simple as one explicit link between two well-defined (cause and effect) concepts. Depending on the context and the conditions, we may, for instance, end up with a set of causations with different certainty values. The appropriate modelling of the causal relations also heavily depend on the use cases of the resulting model, e.g., the kind of reasoning and prediction tasks that it should support. For instance, reasoning on potential guideline and treatment interactions in an individual patient context, e.g., the target use case of [7], requires a highly complex causal model, while in other cases a more simple one might suffice. In Fig. 1, we illustrate this through two examples. At the right (b) is a highly complex conceptual model (inspired by the model in [7]) representing the belief that a causal relation exists, with some frequency and strength. At the left (a) is a also a causal relation, but represented as a much more simple conceptual model.

Our proposed method intends to address the lack of causal models, by automating the generation of highly accuracte causal KGs from text. We intend to cater for the differing requirements of specific use cases by using Knowledge Patterns (KPs), similar to the conceptual models in Fig. 1 coupled with linguistic frames, to represent requirements that make sure the resulting causal model enable the required type of reasoning or predictions.

4 Proposed Approach: Generating Causal KGs from Text

The overarching goal of our research is to support the integration of ML/DL and Knowledge Representation, for improving both accuracy and interpretability of downstream AI applications. As discussed previously, we believe that KGs can play a crucial role in this integration, but then the KG construction bottleneck needs to be resolved. Therefore, we propose to develop new methods and algorithms for KG generation from text, which (a) explicitly take KG requirements into account, e.g. allowing to flexibly specify the required schema of the output graph, and (b) automate the KG curation process to radically improve the quality of resulting KGs with minimal human effort. In order to fulfil a specific set of KG requirements, as well as to achieve a sufficient level of accuracy, we argue that the notion of Knowledge Patterns (KPs) [**?**] as formalisations of KG requirements, is a crucial concept. We here specifically focus on KPs and KGs targeting causal relations, since causal models and causal reasoning are one of the main challenges for ML approaches today. However, the approach we outline is generic, and by exchanging the KPs used, it can be used to target any type of complex relation that can be expressed in natural language. The proposed approach is a novel combination of methods from ML/DL for NLP, with recent advancement in Knowledge Representation, such as KGs and KPs.

As can be seen in Fig. 2, we propose a continuous process that iteratively improves its ML/DL models based on feedback from a curation step. As initial input (1), the process needs a set of KPs rep-

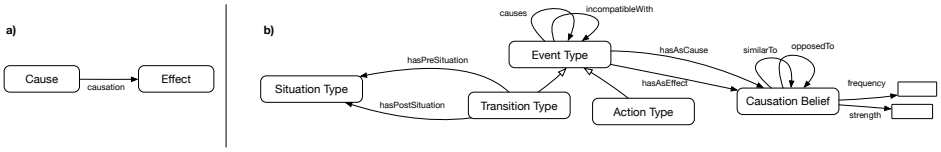


Figure 1. Abstract (conceptual) illustration of two different KPs (here called a and b) for expressing causation, where the patterns produce models at different levels of detail and complexity, hence, targeted at different use cases of the resulting KG. Notation in the figure is informal, but the models could be expressed using an ontology language, such as OWL (as in [7]), in which case the boxes with rounded edges would represent classes, the unfilled arrows subclass relations, and the filled arrows would be object or datatype properties attached to classes based on domain and range restrictions.

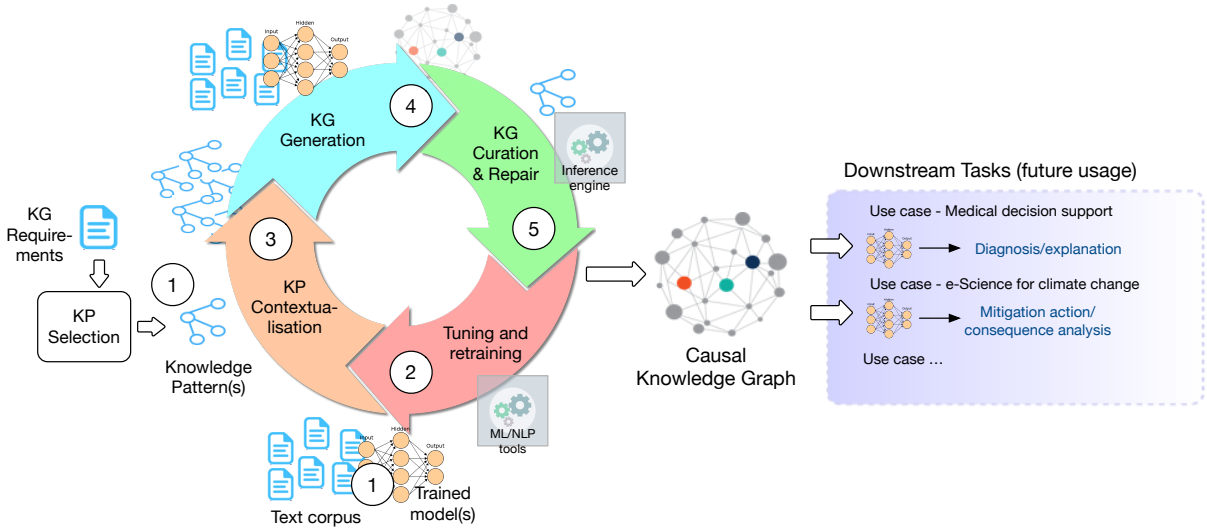


Figure 2. We propose to use KPs to guide the iterative ML process for extracting a Knowledge Graph from unstrutured texts, as well as automating the curation process using a semantic referee. The generated Knowledge Graph, can later be used to support a ML method to derive causal relations from observational data.

resenting the requirements of the output KG, one or more language models, as well as a text corpus from which to extract the KG. The language models then need to be tuned (2) to the relations expressed by the specific KPs at hand. Next, initial instantiations of the KPs, i.e., the linguistic frames they represent, are detected in the text corpus and formalised using the KP as a "schema" (3), whereafter these KP instantiations are merged into an initial KG (4). The initial KG is then subjected to an automated curation and repair process (5), where the formalisation of the KPs is used by a semantic referee to detect postential mistakes in the extracted KG, and suggest repair actions. The result of this curation process is not only a high-quality KG, but also feedback sent back in order to tune, or even retrain, the language models, and to iteratively extend the KG by continuously running the overall process. Below, we go into more details of the ML/DL-based NLP methods to be used, the role and nature of the KPs, and the semantic referee used for curation, respectively.

4.1 Relation Extraction from Text

Causal relations can be extracted from running text by exploiting linguistic cues and then the detected relations can be formalized, for instance, in the form of simple facts (triples). For example, the causal relation in the sentence *COVID-19 is caused by the SARS-CoV-2 virus*, can be formalized as the fact `<SARS-CoV-2 causes COVID-19>`, which could be represented as a triple in a standard Knowledge Representation language, such as RDF[4]. However, in many cases more complex representations are needed, such as including unknown variables, as introduced by Pearl [24], in the notion of Structural Causal Models. Creating a relation such as: *COVID-19:=f(SARS-CoV-2, randomness)*, which means that the appearance of the COVID-19 disease depends on the virus and some other random vairable(s) independent from the virus, e.g. environmental factors, and features of the person in question [5]. An illustration of two conceptual models for representing causal relations were already given in Fig. 1. To instantiate such models (i.e., such KPs), linguistic expessions of causation such as *caused by*, *cause*, *as a result*, *for this reason*, *due to the fact*, *consequently* and similar are cues identified by NLP tools, such as Part-of-Speech Taggers, Dependency Parsers, lexicons, and the like [19, 9].

The NLP task that will contribute to our envisioned approach is mainly Information Extraction (IE), or more specifically Relation Extraction (RE). Recent work in this field includes [30], who describe an innovative approach for relation learning, based on the pretraining of a huge language model, such as BERT [10], passing sentences through its encoder to obtain an abstract notion of a relation, and then fine-tuning on a certain schema, like Wikidata or DBpedia, mainly containing simple binary relations. Our aim is similar, although we intend to develop a slightly different method that can be tuned to a (combination of) a set of smaller, abstract, KPs, targeting more complex relations. An interesting aspect is that [30] are also able to extract generic relations, i.e., potential schema exten-

[4] `https://www.w3.org/RDF/`

[5] The notation is again informal, but the symbol := is here used to indicate the causal relation, and f() represents a function.

sions, which might be a valuable addition in our proposed curation and feedback step. Earlier work on frame detection in text [14, 12], and generation of KGs from this, may also be relevant for comparison, especially since [14] also applied the notion of KPs related to the frames detected, however, they did not allow for the frames to be preselected as the KG requirements, or exchanged.

Further, NELL [23] targets the learning of common facts, extracted from natural language texts. Although their approach does not target a specific output structure or relation, i.e., specific KPs, the continuous improvement process is similar to our proposal. In other recent studies, such as by [25], KGs are also generated from natural language text, but they do not target complex relations such as causality, and the approaches use a fixed output schema.

Very little research exists on extracting more complex relations, i.e. relations that cannot be expressed as single facts (triples), and in particular causal relations, directly from text. One study that generated causal KGs from text is [26]. The difference to our envisioned approach is mainly the types of input data, as well as that [26] targets one fixed logical structure of the output, i.e., a single fixed KP expressing simple direct relations between diagnoses and symptoms. To learn a more complex formalisation of causality, we may also need more complex learning, such as suggested by [18], who proposed a method for extracting a relation graph directly from natural language, where the relations express entailment rules rather than simple facts (triples).

Another area where NLP has been widely used is KG completion, e.g., link and relation prediction in an existing KG. Although we intend to generate a KG "from scratch", the KG generation from instantiated KPs, as well as the subsequent curation process, have some similarities with link and relation prediction. Hence, inspiration may come from work such as [33], who propose to use pretrained language models for knowledge graph completion, scoring candidate triples for addition through their KG-BERT model. This is similar to how we envision to assess potential links between the instantiated KPs, when generating the overall KG. Another approach was recently proposed by [6], where language models such as GPT-2 are combined with a seed KG, allowing the learning of its structure and relations, whereafter the language model can generate new nodes and edges. However, our KPs are abstract and do not contain concrete facts, which is a main difference to the seed KGs they used.

4.2 Knowledge Patterns

The use of patterns in developing knowledge representation models has a long tradition in AI, starting from the idea of Minsky in his proposal of frames [21], and continued towards the notion of ontology design patterns (ODP) in modern ontology engineering [5, 4]. ODPs have also been generalised into KPs [13], where a KP may represents both a linguistic frame that can be detected in text [3], but also the representation of that frame in a desired output formalism. However, in [13], KPs are described and defined informally, and there is currently no concrete formalism for representing and applying KPs specifically for KGs.

In order to capture specific types of knowledge from text, supporting a specific task, such as medical decision support, the knowledge extraction process needs to be carefully guided by the requirements of the intended task of the resulting KG. Tasks may include different types of queries, prediction, applying specific graph pattern matching algorithms, or reasoning. To address this challenge we argue for applying KPs as both a representation of the KG requirements, as well as acting as a "schema" for the resulting KG. In short, as shown in Figure 2, we propose to tune the language models to detect the specific KPs required, and further generate a KG from the instantiated KPs.

Using KPs to guide the learning process makes it possible to capture different possible contextual situations separately, and target different causal models, each focused on a certain specific downstream task. Depending on the relations that are found in the text, KPs will also allow us to calculate more precise certainty values for each captured cause, similar to how we have used knowledge representations as a referee for ML methods in our previous work[1]. This also allows us to filter out extracted knowledge that does not make sense, or is otherwise of questionable quality.

However, this also introduces new challenges, because although KPs have been studied to some extent for ontologies and the Semantic Web, there is so far no formal definition of a KP that can be used operationally (technically) by a system, in particular for KGs. For this purpose we need to operationalise the definition in [13], by expanding on the connection between linguistic frames and ODPs, for use within our KG extraction framework.

4.3 Semantic Referee

Related to the integration of ML/DL and symbolic models, and using knowledge representation to verify and repair results of ML/DL algorithms, we rely on the idea of a semantic referee introduced in our previous work [1]. In that work, we demonstrated the benefit of a semantic referee applied upon a causal model in the form of an ontology (OntoCity) for improving a satellite imagery data classifier. In particular, the ontology together with a reasoning process acted as a semantic referee to guide the ML method (i.e, the classifier). Using causal information represented in the ontology, the semantic referee was able to explain the causes behind errors, and send the explanations as feedback to the classifier. In this way, the ML method is able to know the causes behind its mistakes and therefore better learn from them [1]. We argue that this previous work, will be highly useful, when integrated as step (5) in our KG extraction framework, illustrated earlier in Fig. 2.

5 Conclusion

In this paper, we propose a possible approach to capturing causal knowledge, in a scalable fashion, and representing it as a shared KG. We argue that the advantage of constructing causal KGs is the integration of causality in reasoning and prediction processes, such as the medical diagnosis process, to improve the accuracy and reliability of existing ML/DL-based diagnosis methods, by producing transparent justifications and explanations of the output.

More specifically we focus on KGs as a means for providing background knowledge and reasoning capabilities to ML/DL-based AI systems, and target the KG creation bottleneck. In particular, we recognise the challenge related to causal relations, where the capability of performing causal reasoning is often lacking in pure ML-based systems. Therefore we propose to generate causal KGs from textual information, to then be used as the basis for causal models. Our novel framework is based on using a set of formal KPs as input, acting both as the requirements of the KG as well as the means for formalising the extracted knowledge and curate it through logical reasoning.

REFERENCES

[1] Marjan Alirezaie, Martin Längkvist, Michael Sioutis, and Amy Loutfi, 'Semantic referee: A neural-symbolic framework for enhancing geospatial semantic segmentation', **10**, 863–880, (2019).

[2] Collin F. Baker, Charles J. Fillmore, and John B. Lowe, 'The berkeley framenet project', in *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 1*, ACL '98/COLING '98, p. 86–90, USA, (1998). Association for Computational Linguistics.

[3] Collin F Baker, Charles J Fillmore, and John B Lowe, 'The berkeley framenet project', in *Proceedings of the 17th international conference on Computational linguistics-Volume 1*, pp. 86–90. Association for Computational Linguistics, (1998).

[4] Eva Blomqvist, Karl Hammar, and Valentina Presutti, 'Engineering ontologies with patterns - the extreme design methodology', in *Ontology Engineering with Ontology Design Patterns - Foundations and Applications*, eds., Pascal Hitzler, Aldo Gangemi, Krzysztof Janowicz, Adila Krisnadhi, and Valentina Presutti, volume 25 of *Studies on the Semantic Web*, 23–50, IOS Press, (2016).

[5] Eva Blomqvist and Kurt Sandkuhl, 'Patterns in ontology engineering: Classification of ontology patterns', in *ICEIS 2005, Proceedings of the Seventh International Conference on Enterprise Information Systems, Miami, USA, May 25-28, 2005*, eds., Chin-Sheng Chen, Joaquim Filipe, Isabel Seruca, and José Cordeiro, pp. 413–416, (2005).

[6] Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi, 'Comet: Commonsense transformers for automatic knowledge graph construction', *arXiv preprint arXiv:1906.05317*, (2019).

[7] V. Carretta Zamborlini, *Knowledge Representation for Clinical Guidelines: with applications to Multimorbidity Analysis and Literature Search*, Ph.D. dissertation, Vrije Universiteit Amsterdam, 2017.

[8] Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad, 'Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission', in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, p. 1721–1730, New York, NY, USA, (2015). Association for Computing Machinery.

[9] Tirthankar Dasgupta, Rupsa Saha, Lipika Dey, and Abir Naskar, 'Automatic extraction of causal relations from text using linguistically informed deep neural networks', in *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pp. 306–316, Melbourne, Australia, (July 2018). Association for Computational Linguistics.

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, 'Bert: Pre-training of deep bidirectional transformers for language understanding', in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, (2019).

[11] Michael Färber, Frederic Bartscherer, Carsten Menne, and Achim Rettinger, 'Linked data quality of dbpedia, freebase, opencyc, wikidata, and YAGO', *Semantic Web*, **9**(1), 77–129, (2018).

[12] Marco Fossati, Emilio Dorigatti, and Claudio Giuliano, 'N-ary relation extraction for simultaneous t-box and a-box knowledge base augmentation', *Semantic Web*, **9**(4), 413–439, (2018).

[13] Aldo Gangemi and Valentina Presutti, 'Towards a pattern science for the semantic web', *Semantic Web*, **1**(1-2), 61–68, (2010).

[14] Aldo Gangemi, Valentina Presutti, Diego Reforgiato Recupero, Andrea Giovanni Nuzzolese, Francesco Draicchio, and Misael Mongiovì, 'Semantic web machine reading with fred', *Semantic Web*, **8**(6), 873–893, (2017).

[15] Thomas A. Glass, Steven N. Goodman, Miguel A. Hernán, and Jonathan M. Samet, 'Causal inference in public health', (March 2013).

[16] Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard de Melo, Claudio Gutierrez, José Emilio Labra Gayo, Sabrina Kirrane, Sebastian Neumaier, Axel Polleres, Roberto Navigli, Axel-Cyrille Ngonga Ngomo, Sabbir M. Rashid, Anisa Rula, Lukas Schmelzeisen, Juan Sequeda, Steffen Staab, and Antoine Zimmermann. Knowledge graphs, 2020.

[17] Pearl J. and Mackenzie D., *The book of why: the new science of cause and effect*, Basic Books, 2018.

[18] Mohammad Javad Hosseini, Nathanael Chambers, Siva Reddy, Xavier R Holt, Shay B Cohen, Mark Johnson, and Mark Steedman, 'Learning typed entailment graphs with global soft constraints', *Transactions of the Association for Computational Linguistics*, **6**, 703–717, (2018).

[19] Christopher S. G. Khoo, Syin Chan, and Yun Niu, 'Extracting causal knowledge from a medical database using graphical patterns', in *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pp. 336–343, Hong Kong, (October 2000). Association for Computational Linguistics.

[20] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer, 'Dbpedia - A large-scale, multilingual knowledge base extracted from wikipedia', *Semantic Web*, **6**(2), 167–195, (2015).

[21] Marvin Minsky, 'A framework for representing knowledge', *MIT-AI Laboratory Memo 306*.

[22] Riccardo Miotto, Fei Wang, Shuang Wang, Xiaoqian Jiang, and Joel Dudley, 'Deep learning for healthcare: review, opportunities and challenges', *Briefings in bioinformatics*, **19 6**, 1236–1246, (2018).

[23] Tom Mitchell, William Cohen, Estevam Hruschka, Partha Talukdar, Bishan Yang, Justin Betteridge, Andrew Carlson, Bhanava Dalvi, Matt Gardner, Bryan Kisiel, et al., 'Never-ending learning', *Communications of the ACM*, **61**(5), 103–115, (2018).

[24] JUDEA PEARL, 'Causal diagrams for empirical research', *Biometrika*, **82**(4), 669–688, (12 1995).

[25] Anderson Rossanez and Julio Cesar dos Reis, 'Generating knowledge graphs from scientific literature of degenerative diseases', (2019).

[26] Maya Rotmensch, Yoni Halpern, Abdulhakim Tlimat, Steven Horng, and David Sontag, 'Learning a health knowledge graph from electronic medical records', *Scientific reports*, **7**(1), 1–11, (2017).

[27] J. Runge, S. Bathiany, E. Bollt, G. Camps-Valls, D. Coumou, E. Deyle, M. Glymour, C. andKretschmer, M.D. Mahecha, E.H. van Nes, J. Peters, R. Quax, M. Reichstein, B. Scheffer, M. Schölkopf, P. Spirtes, G. Sugihara, J. Sun, Ka. Zhang, and J. Zscheischler, 'Inferring causation from time series with perspectives in earth system sciences', *Nature Communications*, (2019).

[28] Edward W. Schneider, 'Course modularization applied: The interface system and its implications for sequence control and data analysis', (1973).

[29] Peter Schulam and Suchi Saria, 'Reliable decision support using counterfactual models', in *Advances in Neural Information Processing Systems 30*, eds., I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, 1697–1708, Curran Associates, Inc., (2017).

[30] Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski, 'Matching the blanks: Distributional similarity for relation learning', in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 2895–2905, (2019).

[31] Peter Spirtes, 'Introduction to causal inference', *J. Mach. Learn. Res.*, **11**, 1643–1662, (August 2010).

[32] Denny Vrandecic and Markus Krötzsch, 'Wikidata: a free collaborative knowledgebase', *Commun. ACM*, **57**(10), 78–85, (2014).

[33] Liang Yao, Chengsheng Mao, and Yuan Luo, 'Kg-bert: Bert for knowledge graph completion', *arXiv preprint arXiv:1909.03193*, (2019).

Assessing the clinicians' pathway to embed artificial intelligence for assisted diagnostics of fracture detection

Carlos Francisco Moreno-García[1], Truong Dang[1], Kyle Martin[1], Manish Patel[2], Andrew Thompson[2], Lesley Leishman[1], Nirmalie Wiratunga[1]

Abstract. Fracture detection has been a long-standing paradigm on the medical imaging community. Many algorithms and systems have been presented to accurately detect and classify images in terms of the presence and absence of fractures in different parts of the body. While these solutions are capable of obtaining results which even surpass human scores, few efforts have been dedicated to evaluate how these systems can be embedded in the clinicians and radiologists working pipeline. Moreover, the reports that are included with the radiography could also provide key information regarding the nature and the severity of the fracture. In this paper, we present our first findings towards assessing how computer vision, natural language processing and other systems could be correctly embedded in the clinicians' pathway to better aid on the fracture detection task. We present some initial experimental results using publicly available fracture datasets along with a handful of data provided by the National Healthcare System from the United Kingdom in a research initiative call. Results show that there is a high likelihood of applying transfer learning from different existing and pre-trained models to the new records provided in the challenge, and that there are various ways in which these techniques can be embedded along the clinicians' pathway.

Keywords: Fracture detection, natural language processing, convolutional neural networks, clinicians' pathway

1 Introduction

In recent years, fracture detection has been one of the most cited challenges in medical imaging analysis, evidenced both by public competitions [18] and clinical trials [9] alike. The design of a system which aids clinicians in the automatic detection of fractures is of paramount to reduce the workload of the front line staff and allow them more time to focus on the most urgent cases. To address this issue, the Scottish Government, Opportunity North East (ONE) and the Small Business Research Initiative (SBRI) announced a challenge to carry out a project towards looking at this problem in the healthcare system in the northeast of Scotland[3]. A team comprised of members from the industry (Jiva.ai) and academia (Robert Gordon University) was formed to look at the problem and design a solution.

In addition, a key contribution of this work is the modelling of the clinician's pathway, exploring the current process of radiology imaging for fracture treatment. This was built through a series of co-creation sessions with a reporting radiographer, then verified by two clinicians and two other reporting radiographers. By designing this pathway, we identified three key stakeholders (clinician, radiologist, patient) and four sub-processes: (1) requesting radiology images; (2) acquiring radiology images; (3) reporting on radiology images; and (4) decision support. By understanding the current approach to radiology imaging for fracture treatment, we consider that we are capable of offering a valuable contribution that will allow research groups to pinpoint opportunities for smart automation of this process in future.

The contributions of this paper are as follows:

1. Identify the current processes involved with the various procedures affected by radiology imaging for fractures.
2. Explore where these processes can be improved through the implementation of AI.
3. Identify what form of AI would be most applicable in order to maximise the obtained benefits by the affected stakeholder.
4. Given the limited amount of samples provided by the challenge proposer, perform initial proof of concept tests using baseline methods to identify the potential of transfer learning in this domain.

2 Related Work

2.1 Image recognition

Only a handful of demonstrations of machine learning, computer vision and natural language processing for bone fracture detection appear in scientific literature. Lindsey et al. [9] demonstrated that a deep neural network trained on 256'000 x-rays could detect fractures with a similar diagnostic accuracy to a sub-specialised orthopaedic surgeon. Also, Olczak et al. [10] applied deep learning to analyse 256'458 x-rays, and concluded that artificial intelligence methods are applicable for radiology screening, especially during out-of-hours healthcare or at remote locations with poor access to radiologists

[1] Robert Gordon University, Garthdee Road, Aberdeen, Scotland, UK, Contact email: c.moreno-garcia@rgu.ac.uk

[2] Jiva.ai Ltd, 132 Street Lane, Leeds, UK, Contact email: info@jiva.ai

[3] http://cfmgcomputing.blogspot.com/2019/11/artificial-intelligence-for-fracture.html

or orthopedists. Smaller scale studies using tens to low thousands of images include Lawn et al. [8], Kim and MacKinnon [5], Tomita et al. [14], Dimililer [3] and Bandyopadhyay et al. [1], amongst others.

Three technical frames have been described as applicable for ML in radiology: image segmentation, registration and computer-aided detection and diagnosis [17]. Out of these, graph models, such as Bayesian networks and Markov random fields, have been identified as the two most widely used techniques for fracture modelling. However, recent advances in generative deep models (e.g. variational autoencoders) [6] have been applied to annotate both images and text; which are yet to be exploited in radiology and related applications. Similarly, multi-modal learners [16] have been used to learn from image and text to improve recognition of objects; unlike single modality learners these can combine mixed embedding spaces that unite different modalities. These have been applied to public text and images but digital health applications are yet to emerge. Given the relevance of both image and text to clinical radiology we expect to adapt these algorithms to create an innovative unified embedding suited to automated annotation by deep generative algorithms. Indeed, we can use state-of-the-art translation algorithms such as transformers [15] which exploit similarity and capitalise on adjacency information, to generate reports from both radiograms and clinical text.

3 Modelling a Clinician's Pathway

Any artificially intelligent solution which is suggested to resolve some problem within the field of radiology should be rooted in deep understanding of the domain and user requirements. With this in mind, we have received input from domain-experts to develop a clinician's pathway, detailing current process of radiology imaging for fracture treatment. In this paper, we aim to demonstrate the opportunities which offer potential for smart automation in this field. In particular, we aim to highlight the areas where the application of artificial intelligence would be impactful for increasing efficiency, improving patient experience and decreasing cost.

3.1 Co-Creation of Clinician's Pathway

We performed a multi-stage co-creation process to model the clinician's pathway which was indicative of real-world radiology practice. These stages were divided into a design phase, where we aimed to understand and model the current processes for the acquisition and reporting of radiology images, and a validation phase, in which we obtained unbiased feedback on our model from personnel external to our co-creation. The result is a clinician's workflow which has been developed alongside a reporting radiographer, and verified by two clinicians (one consultant radiologist and one senior accident & emergency doctor) and two reporting radiographers from within the National Health Service (NHS) Scotland. We are therefore confident in its accuracy and its suitability to describe real radiology processes. Although this pathway has been built with input from British radiologists, we suggest it can be generalised to wider radiology practices (within reason).

During the design phase, we organised two separate co-creation sessions. In the first session, we met with a reporting radiographer to discuss the complete journey of a patient who was given an appointment for radiology imaging for a suspected fracture. This session was useful to establish the process start-points and end-points. In the second session, we observed a reporting radiographer reporting on a series of x-ray images for suspected fractures. This session was intended to identify relevant technologies and the role they played in reporting on radiology images. The outcome of this two stage design phase was an initial draft of the clinician's pathway which could be validated by domain experts.

We then organised three sessions for the validation of the developed workflow. In the first session, we obtained feedback upon the pathway from the reporting radiographer who was directly involved in its formation. This allowed any errors which had arisen due to misunderstanding aspects of the design phase to be corrected. In the second session, a member of the research team met with a clinician and a reporting radiographer to explain and discuss the draft pathway and obtain feedback. This session was designed to ensure that the pathway could be generalised to more than just the single radiographer with whom it had been co-designed. In particular, the session resulted in a number of updates to the role of the clinician as an actor in the process. Finally, we used the third session as an opportunity to obtain blind feedback on the developed pathway as a form of litmus test regarding its accuracy to radiology practice. We presented the pathway to a new clinician and reporting radiographer, and requested feedback on any areas where they felt (a) that the pathway was not indicative of real-world practice and (b) that there were opportunities for artificial intelligence to make the process more efficient.

The results of the validation sessions were very valuable for the design process. As a methodology, by performing our co-creation of the clinician's pathway in this manner, we are confident it is accurate to real-world practice, and generalisable beyond simply an individual's viewpoint. In the final validation session, the clinician did not highlight any areas of the workflow which were not indicative of real-world practice, while the reporting radiologist suggested only a minor amendment to terminology. Furthermore, the areas which both of the participants suggested were suitable for artificial intelligence to make a process improvement very closely overlapped with our own findings as researchers. We discuss this in more detail in Section 3.3. In the following subsection, we will introduce and discuss the developed clinician's pathway in detail.

3.2 Resulting Clinician's Pathway

In modelling the process of radiology imaging for fracture treatment, we identified three key stakeholders (clinician, radiologist, patient) and four sub-processes: (1) requesting radiology images; (2) acquiring radiology images; (3) reporting on radiology images; and (4) decision support. The complete figure can be accessed via this link[4] and can be seen in Figure 1. We summarise our pathway using a workflow diagram which we will break down into respective processes in Figures 2, 3 and 4.

[4] https://www.dropbox.com/s/3gx7bicf43bn0lx/wrk_flw_comp_c.pdf?dl=0

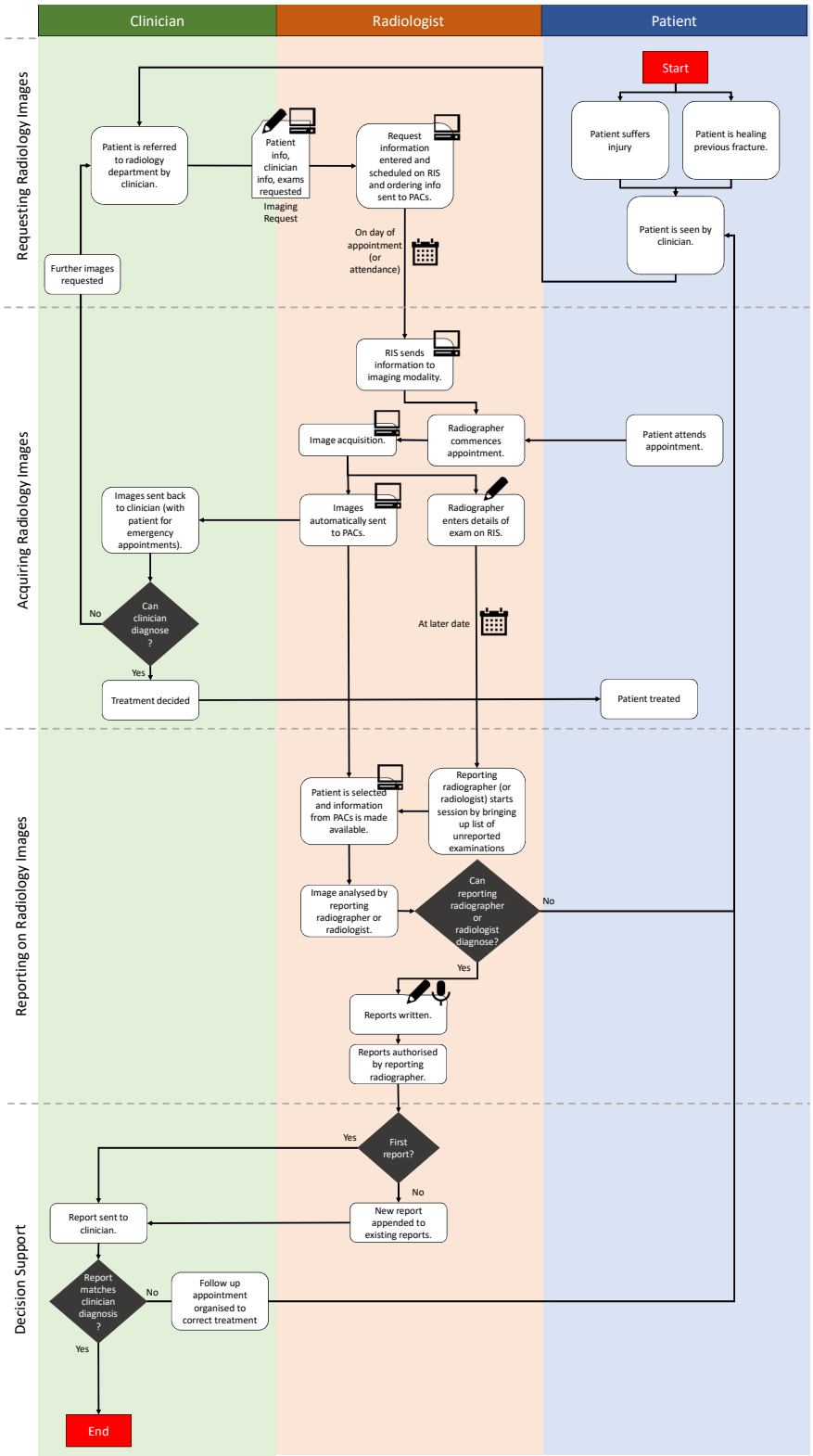


Figure 1. A complete depiction of the current workflow for requesting, acquiring and reporting upon radiology images

1. Requesting Radiology Images: When requesting radiography images for a patient, a clinician sends an imaging request to the radiology department. This request contains information on the patient's demographic (including a medical history), information about the clinician making the request and the examinations requested. It allows an appointment to be scheduled on the Radiology Information System (RIS), and the ordering information is dispatched to Picture Archiving

and Communications System (PACS). PACS is an end-to-end system that supports the process of acquiring radiography images of a patient from referral of the patient until diagnosis and subsequent treatment are agreed. Contained within PACS is the RIS, where the textual components of patient information is stored.

2. Acquiring Radiology Images: Each day, the RIS automatically generates a work list for each imaging modality. This work list gives the radiologist (or the reporting radiographer) access to the request information created by the original referring clinician, helping them to understand the imaging requirements. This enables the radiologist to perform the requested imaging. After the imaging has been performed, the medical equipment will generate images in Digital Imaging and Communication in Medicine (DICOM) standard, and load them into a graphic user interface where they are made available for the radiologist to annotate and report. A graphical representation of this is displayed in Figure 2. DICOM defines an image standard and format for medical images. Images are high resolution and are linked directly with other patient data (such as name, gender and age). It was developed by the National Electrical Manufacturers Association (NEMA) as part of a set of standards that define best practice and inform the international standard for the capture, retrieval, storage and transmission of medical imaging data. DICOM is currently the most commonly used standard across the world for medical imaging, and is implemented in most radiology, cardiology and radiotherapy devices, as well as devices in other medical domains such as dentistry.

3. Reporting on Radiology Images: Radiologists can access these images by calling up a list of unreported examinations. For each patient, previous images and reports are also available to support diagnosis. PACS enables radiologists to annotate the images to highlight areas of interest or identify supporting evidence for their diagnosis. These annotations include the ability to perform simple measurements (length of objects, angles of intersections, etc) and to mark a Region of Interest (ROI) on the image. This allows the radiologist to use tools to capture metadata about the ROI, including its area, average pixel values, standard deviation, and range of pixel values.

The radiologist will then generate a textual report to summarise and describe their findings. These reports have no set template or length, but generally include a statement of whether a fracture has been detected, what type of fracture it is, where it is located, and the seriousness of the breakage. Furthermore, the reports may be appended to existing documentation on the patient (if previous radiology records exist) or may be used to begin a radiology record (if no previous visits have been recorded). Many countries then require the reports to be authorised by a radiologist before being released to a clinician. For example, within the United Kingdom the standards for fail-safe communication of radiology reports are governed by the Royal College of Radiologists (RCR)[5]. The result of these factors is that the reports are a complex textual data source with limited uniformity and describing a broad range of diagnosis and observations. We represent this information as part of the workflow displayed in Figure 3.

4. Decision Support In the existing pathway, decision support occurs after the radiology reports are generated. This is non-optimal; often for accident and emergency fracture cases (which make up the majority of fractures in a hospital) the clinician will attempt to read and comprehend the generated radiology images without any input from an expert radiologist. This can be seen in the current workflow in Figures 2 and 4. This occurs because experts can be unavailable - not all radiology staff are sufficiently trained to report on acquired images. As a result the clinician is forced to make a diagnosis and organise follow up treatment on the basis of their individual knowledge. This can lead to misdiagnosis, if the clinician's findings are not consistent with the radiologist's, which can have an impact on the patient's health as well as financial consequences for the hospital involved.

Having developed an understanding of the current procedure of radiology imaging for fracture treatment, we are motivated to make some recommendations where artificial intelligence could make improvements to this process.

3.3 Opportunities for Artificial Intelligence

The key outcome of this work is highlighting the applicability of artificial intelligence in two places: to reduce burden on radiologists by (1) autonomously classifying radiology images and (2) generating understandable and accurate medical reports to describe the intelligent system's findings. Applications of artificial intelligence to fill these gaps presents an opportunity to improve decision-support for clinicians by giving them access to the information immediately. This is a key factor that is missing from much of the research literature on this topic; although an artificial intelligence method for fracture recognition should enhance the efficiency of radiologists, it should also improve the decision-making of clinicians. Therefore, it should be of a suitable form to be absorbed by that user group.

This outcome is supported by the verification obtained from our test group of clinicians and radiologists. Based on their feedback, we have highlighted the most impacted area of the current clinical pathway in Figure 5.

As seen in our discussion of related work, there has been much exploration of autonomous classification of fracture images throughout the literature [1, 3, 5, 8, 9, 10, 14]. However, few works have considered how this could be integrated with existing medical processes. It is clear that from a clinician's perspective, it would be desirable to have the classification of the image and the report in order to support their decision-making. This suggests an ecosystem of artificial intelligence processes would be much more suitable than a standalone method.

4 Experiments on Image Classification

The main purpose of the experimental framework was to test the learning capabilities of different baseline algorithms and settings to classify the images provided in the challenge as fracture/no fracture. To do so, the first task consisted of having a specialist re-annotate the data provided by splitting it into fracture and no fracture labels, based both on the visual aspect of the radiography and on the information provided by the text reports. It was discovered that while most of the

[5] http://bit.ly/rcr-standards

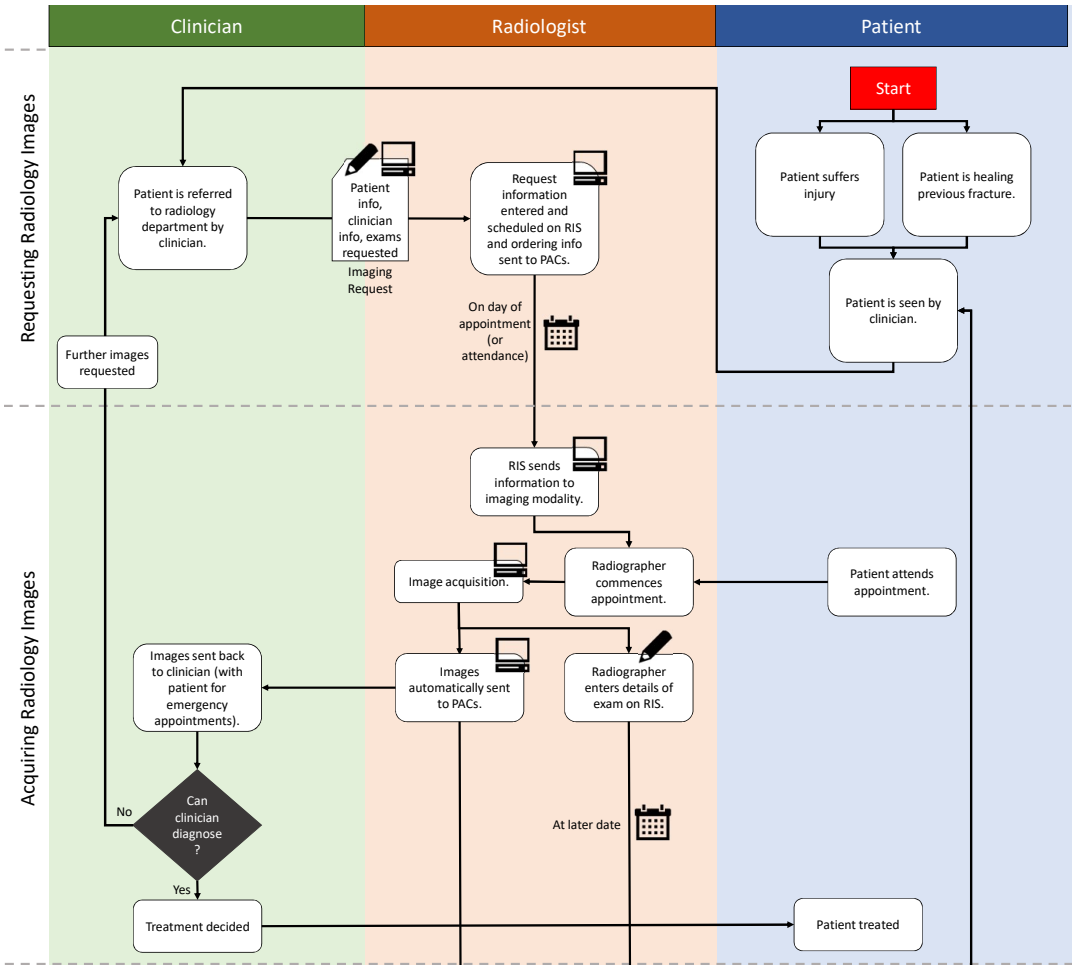


Figure 2. The co-created workflow for requesting and acquiring radiology images.

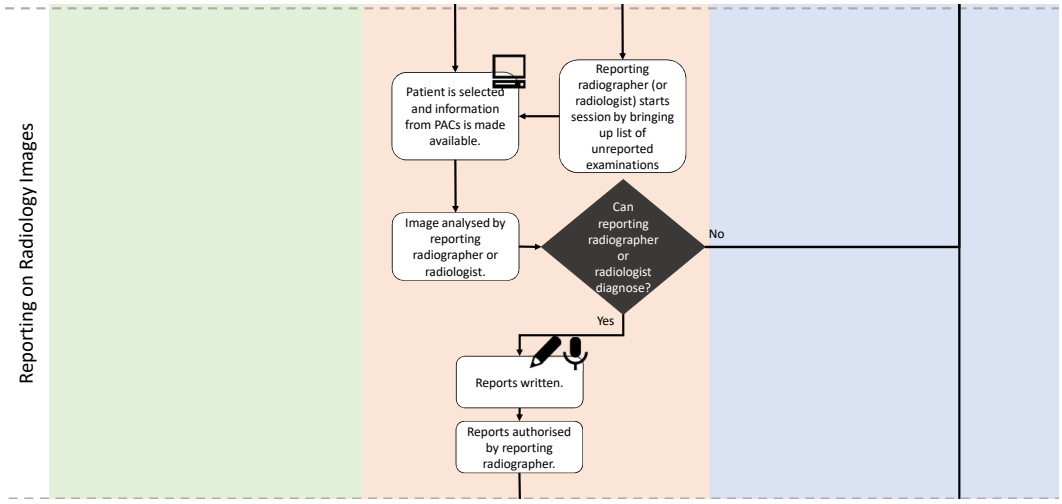


Figure 3. The co-created workflow for reporting on radiology images.

images corresponded to "regular" scenarios (where the purpose is to assess whether the patient has suffered a fracture or not), some other cases also contained follow-up reports (identified as POP) where the issue is not to identify the presence/absence of a fracture, but rather to give a follow up for a patient which already has had the fracture identified in a previous visit. We labelled 73 images as positive (i.e. with fracture) and 138 negative (i.e. no fracture). Moreover, six examples were POP fractures, and thus were not included in our experiments.

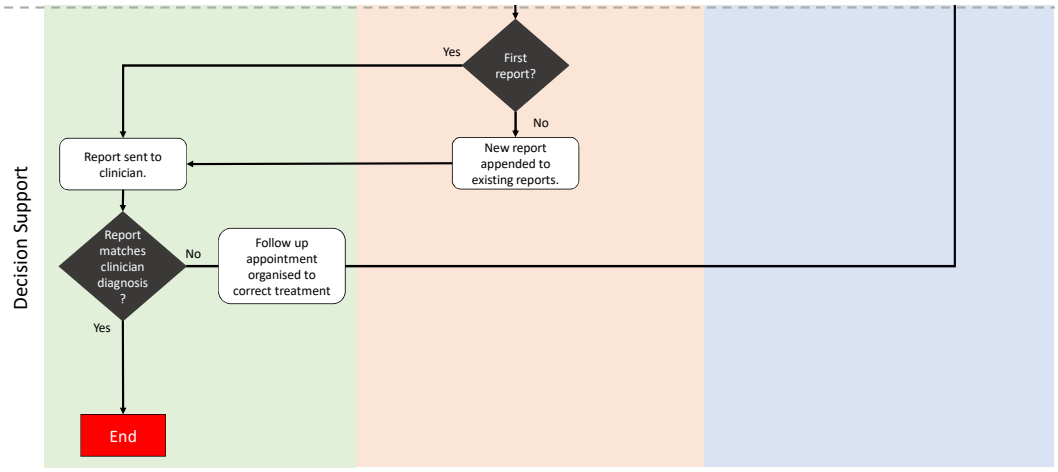


Figure 4. The co-created workflow for using radiology images for decision support.

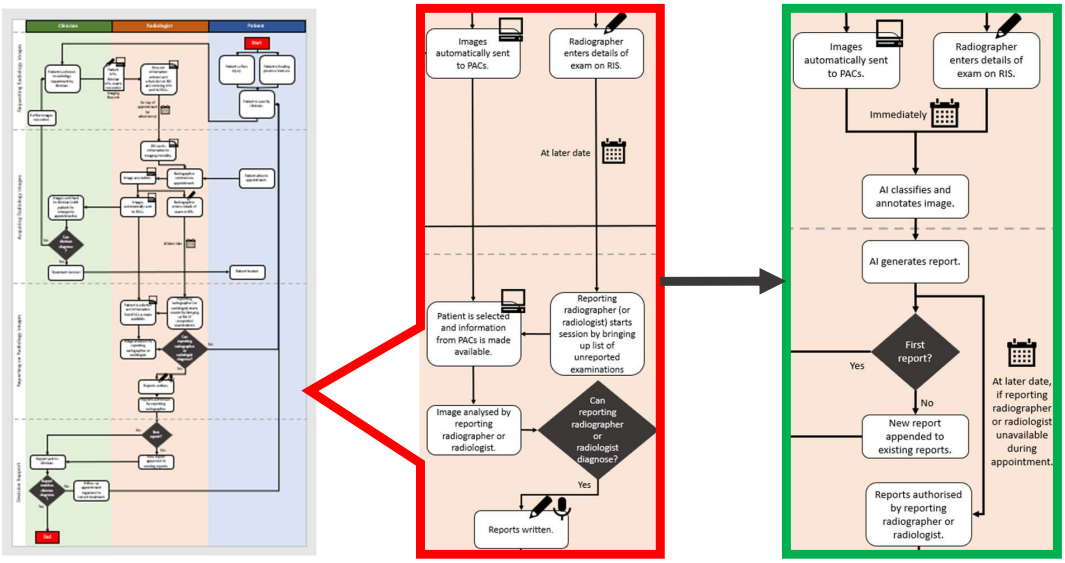


Figure 5. AI provides an opportunity to provide decision support much more quickly by classifying an acquired image and generating a suitable report to describe its findings.

4.1 Datasets

To demonstrate the potential of transfer learning capabilities of the selected algorithms towards the NHS provided records, we used the following publicly available dataset.

MURA: The MURA (MUsculoskeletal RAdiographs) dataset is a large dataset of bone X-rays. Algorithms are tasked with determining whether an X-ray study is normal or abnormal. It consists of X-ray scans on elbow, finger, forearm, hand, humerus, shoulder and wrist. The training set consists of 14'873 positive cases and 21'939 negative cases while the validation set has 1'530 positive examples and 1'667 negative examples. Among them, forearm and wrist are close to our problem, which consists of 7'443 negative and 5'094 positive examples. Images from this dataset can be accessed here[6].

[6] https://stanfordmlgroup.github.io/competitions/mura/

4.2 Image Preprocessing

To increase the likelihood of classification and the training sample size, We applied the following preprocessing techniques, which are the most commonly used in related literature [18]:

- horizontal flip, width shift by 0.1,
- height shift by 0.1,
- shearing with range 0.1,
- zoom with range from 0.9 to 1.25 and
- random rotation from 0 to 15 degrees.

4.3 Architecture Details

These followings baseline architectures were used in our experiments:

A **VGG 16** [11] is a Convolutional Neural Network (CNN) model proposed by Simonyan and Zisserman. The model

achieves 92.7% top-5 test accuracy in ImageNet, which is a dataset of over 14 million images belonging to 1'000 classes. It is an improvement over the classical AlexNet [7] by replacing large kernel-sized filters (11 and 5 in the first and second convolutional layer, respectively) with multiple 3×3 kernel-sized filters one after another. The original VGG16 was trained for weeks and was implemented using NVIDIA Titan Black GPU's.

B **Resnet 50** [4] is a CNN architecture of 50 layers deep, each of which is formulated as learning residual functions with reference to the layer inputs, instead of learning unreferenced functions. Because of these residual modules, the architecture can become very deep. This architecture won the 1^{st} place on the ILSVRC 2015 classification challenge.

C **Inception V3** [12, 13] is a CNN architecture which achieved improved utilisation of the computing resources inside the network by carefully crafted design that allows for increasing the depth and width of the network while keeping the computational budget constant. To optimize quality, the architectural decisions were based on the Hebbian principle and the intuition of multi-scale processing. The authors also proposed ways to scale up networks in ways that aim at using the added computation as efficiently as possible by suitably factorised convolutions and aggressive regularization. Tests were made on the ILSVRC 2012 dataset, in which with an ensemble of four models and multi-crop evaluation, authors reported 3.5% top-5 error on the validation set (3.6% error on the test set) and 17.3% top-1 error on the validation set.

4.4 Experiment Details

To have different points of comparison, we tested the results of using the three aforementioned classifiers to classify images from the MURA dataset. We distinguished between the following four configurations:

1. When the networks were pre-trained with ImageNet [2].
2. When they were initialised randomly.
3. When the networks were initialised randomly, trained on all the MURA dataset (except wrist and arm images) and then retrained on wrist and arm images.
4. When the networks were pre-trained from ImageNet randomly, trained on all of MURA (except wrist and arm) and then retrained on wrist and arm images.

The accuracy result can be seen in Table 4.4 and the run time in Table 4.4:

Table 1. Accuracy results for cases from (1) to (4)

	Case (1)	Case (2)	Case (3)	Case (4)
VGG16	0.82	0.535417	0.535417	0.798958
Resnet50	0.8083	0.535417	0.535417	0.783333
InceptionV3	0.677083	0.535417	0.535417	0.536458

The results showed that case 1 with VGG 16 and ResNet 50 delivered the best accuracy overall (82% and 80% respectively), implying that it is possible to obtain good accuracy provided that we can train the systems with sufficient data, regardless of its origin. Moreover, case 2 with these same architectures also showed good performance, (79% and 78% respectively), but even with the retraining on wrist and arm images, results were slightly worse than training only with ImageNet images. This may be due to the fact that some wrist/arm images had to be used for such retraining instead of testing. In terms of run time, we also found out that case 1 overall is faster to train and test.

Table 2. Running time (in seconds) for cases from (1) to (4)

	Case (1)	Case (2)	Case(3)	Case (4)
VGG16	1869.48	2024.83	5557.05	5374.36
Resnet50	1776.34	1308.95	11406.28	7378.9
InceptionV3	3128.09	3086.32	11429.92	8208.37

After this initial validation, we tested the transfer learning capability from MURA to the newly acquired images. We tested the following three cases:

5. When networks were pre-trained on ImageNet, trained on MURA and tested on the new dataset.
6. Mixing MURA and new images to generate both training and test sets (70% train, 30% test).
7. Same as the previous case, however the test set was composed of 70% of MURA images and 30% from the new dataset.

The accuracy results are shown in Table 4.4:

Table 3. Accuracy results for cases from (5) to (7)

	Case (5)	Case (6)	Case (7)
VGG16	0.668293	0.809524	0.704918
Resnet50	0.673171	0.76112	0.606557
InceptionV3	0.673171	0.727106	0.754098

In contrast to what was expected from the previous test, we observed that for case 5, all CNNs were unable to learn how to classify the new images. In contrast, it was more likely to obtain higher accuracy rates for case 6 and VGG 16 (81%), although this is a direct result of images from the MURA dataset being mixed within the test set. Meanwhile, case 7 and Inception V3 obtained 75% accuracy, but keeping in mind that the test set is only composed of new images, this was a clear indication that it is possible to transfer a model using a larger amount of images. In terms of run time, we discovered that it was faster to train networks through case 6, followed by case 6 and case 7 respectively. The complete run time results can be seen in Table 4.4.

Table 4. Running time (in seconds) for cases from (5) to (7)

	Case (5)	Case (6)	Case (7)
VGG16	2727.67	2700.84s	3718.74
Resnet50	1889.82	3495.72s	4280.42
InceptionV3	2763.39	10388.83s	9656.18

5 Conclusion

In this paper, we have presented a first step towards assessing the most proper way to embed machine learning, computer vi-

sion and natural language processing into the clinicians' pathway to improve assisted diagnostics of fracture detection. We have reviewed the most significant literature and designed a pipeline where we have annotated the most relevant action points where artificial intelligence can be used to improve the current practices. In addition, we have carried out some initial experiments to verify how current methods and transfer learning perform on identifying fractures in a reduced dataset provided by the British public health service. Results show that there is a great likelihood of being able to apply transfer learning for these purposes, and in the case that more images are provided by the challenge setter, then the accuracy can vastly improve.

We will continue this partnership to explore more ways in which we can further improve our findings and including other technologies to enhance the existing results. Finally, we will keep working with clinicians and radiographers to correctly assess their pathways and effectively applying these technologies in commercial settings.

Acknowledgment

We would like to acknowledge the Scottish Government's Small Business Research Initiative (SBRI) and Opportunity North East (ONE) for supporting this work, and to the Data Safe Haven (DaSH) from the National Health Service (NHS) for granting access to the test data.

REFERENCES

[1] Oishila Bandyopadhyay, Arindam Biswas, and Bhargab B. Bhattacharya. Long-bone fracture detection in digital X-ray images based on digital-geometric techniques, 2016.

[2] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[3] Kamil Dimililer. IBFDS: Intelligent bone fracture detection system. In *Procedia Computer Science*, 2017.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv 1512.03385*, pages 770–778, 06 2016.

[5] D. H. Kim and T. MacKinnon. Artificial intelligence in fracture detection: transfer learning from deep convolutional neural networks. *Clinical Radiology*, 73(5):439–445, 2018.

[6] Diederik P. Kingma, Danilo J. Rezende, Shakir Mohamed, and Max Welling. Semi-supervised learning with deep generative models, 2014.

[7] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[8] Brian Lawn. *Fracture of brittle solids*. Cambridge university press, 1993.

[9] Robert Lindsey, Aaron Daluiski, Sumit Chopra, Alexander Lachapelle, Michael Mozer, Serge Sicular, Douglas Hanel, Michael Gardner, Anurag Gupta, Robert Hotchkiss, and Hollis Potter. Deep neural network improves fracture detection by clinicians. *Proceedings of the National Academy of Sciences of the United States of America*, 115(45):11591–11596, nov 2018.

[10] Jakub Olczak, Niklas Fahlberg, Atsuto Maki, Ali Sharif Razavian, Anthony Jilert, André Stark, Olof Sköldenberg, and Max Gordon. Artificial intelligence for analyzing orthopedic trauma radiographs: Deep learning algorithms are they on par with humans for diagnosing fractures? *Acta Orthopaedica*, 88(6):581–586, 2017.

[11] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv 1409.1556*, 09 2014.

[12] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going Deeper with Convolutions. *arXiv:1409.4842 [cs]*, September 2014. arXiv: 1409.4842.

[13] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567, 2015.

[14] Naofumi Tomita, Yvonne Y. Cheung, and Saeed Hassanpour. Deep neural networks for automatic detection of osteoporotic vertebral fractures on CT scans. *Computers in Biology and Medicine*, 98(February):8–15, 2018.

[15] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc., 2017.

[16] O. Vinyals, D. Bohus, and R. Caruana. Learning Speaker, Addressee and Overlap Detection Models from Multimodal Streams. In *ICMI*, 2012.

[17] Xingwei Wang, Lihua Li, Wei Liu, Weidong Xu, Dror Lederman, and Bing Zheng. An interactive system for computer-aided diagnosis of breast masses. *Journal of Digital Imaging*, 25:570–579, 2012.

[18] Xulei Yang, Zeng Zeng, Sin G. Teo, Li Wang, Vijay Chandrasekhar, and Steven Hoi. Deep learning for practical image recognition: Case study on kaggle competitions. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '18, pages 923–931, New York, NY, USA, 2018. Association for Computing Machinery.

The OhioT1DM Dataset for Blood Glucose Level Prediction: Update 2020

Cindy Marling and **Razvan Bunescu** [1]

Abstract. This paper documents the OhioT1DM Dataset, which was developed to promote and facilitate research in blood glucose level prediction. It contains eight weeks' worth of continuous glucose monitoring, insulin, physiological sensor, and self-reported life-event data for each of 12 people with type 1 diabetes. An associated graphical software tool allows researchers to visualize the integrated data. The paper details the contents and format of the dataset and tells interested researchers how to obtain it.

The OhioT1DM Dataset was first released in 2018 for the first Blood Glucose Level Prediction (BGLP) Challenge. At that time, the dataset was half its current size, containing data for only six people with type 1 diabetes. Data for an additional six people is being released in 2020 for the second BGLP Challenge. This paper subsumes and supersedes the paper which documented the original dataset.

1 INTRODUCTION

Accurate forecasting of blood glucose levels has the potential to improve the health and wellbeing of people with diabetes. Knowing in advance when blood glucose is approaching unsafe levels provides time to proactively avoid hypo- and hyper-glycemia and their concomitant complications. The drive to perfect an artificial pancreas [2] has increased the interest in using machine learning (ML) approaches to improve prediction accuracy. Work in this area has been hindered, however, by a lack of real patient data; some researchers have only been able to work on simulated patient data.

To promote and facilitate research in blood glucose level prediction, we have curated the OhioT1DM Dataset and made it publicly available for research purposes. To the best of our knowledge, this is the first publicly available dataset to include continuous glucose monitoring, insulin, physiological sensor, and self-reported life-event data for people with type 1 diabetes.

The OhioT1DM Dataset contains eight weeks' worth of data for each of 12 people with type 1 diabetes. These anonymous people are referred to by randomly selected ID numbers. All data contributors were on insulin pump therapy with continuous glucose monitoring (CGM). They wore Medtronic 530G or 630G insulin pumps and used Medtronic Enlite CGM sensors throughout the 8-week data collection period. They reported life-event data via a custom smartphone app and provided physiological data from a fitness band. The first cohort of six individuals wore Basis Peak fitness bands. Data for this cohort was released in 2018. The second cohort of six individuals wore the Empatica Embrace. Data for this cohort is included in the 2020 release. Table 1 shows the gender, age range, insulin pump model, and sensor band type for each data contributor, by cohort.

[1] Ohio University, USA, email: {marling,bunescu}@ohio.edu

Table 1. Gender, age range, insulin pump model, and sensor band type for each data contributor, by cohort

ID	Gender	Age	Pump Model	Sensor Band	Cohort
540	male	20–40	630G	Empatica	2020
544	male	40–60	530G	Empatica	2020
552	male	20–40	630G	Empatica	2020
567	female	20–40	630G	Empatica	2020
584	male	40–60	530G	Empatica	2020
596	male	60–80	530G	Empatica	2020
559	female	40–60	530G	Basis	2018
563	male	40–60	530G	Basis	2018
570	male	40–60	530G	Basis	2018
575	female	40–60	530G	Basis	2018
588	female	40–60	530G	Basis	2018
591	female	40–60	530G	Basis	2018

The dataset includes: a CGM blood glucose level every 5 minutes; blood glucose levels from periodic self-monitoring of blood glucose (finger sticks); insulin doses, both bolus and basal; self-reported meal times with carbohydrate estimates; self-reported times of exercise, sleep, work, stress, and illness; and data from the Basis Peak or Empatica Embrace band. The data for individuals who wore the Basis Peak band includes 5-minute aggregations of heart rate, galvanic skin response (GSR), skin temperature, air temperature, and step count. The data for those who wore the Empatica Embrace band includes 1-minute aggregations of GSR, skin temperature, and magnitude of acceleration. Both bands indicated the times they detected that the wearer was asleep, and this information is included when available. However, not all data contributors wore their sensor bands overnight.

Data for the first six individuals was released in 2018 for the first Blood Glucose Level Prediction (BGLP) Challenge, which was held in conjunction with the 3rd International Workshop on Knowledge Discovery in Healthcare Data, at IJCAI-ECAI 2018, in Stockholm, Sweden. Data for six additional people is being released in 2020 for the second BGLP Challenge, to be held at the 5th International Workshop on Knowledge Discovery in Healthcare Data, at ECAI 2020, in Santiago de Compostela, Spain. This paper subsumes and supersedes the paper which documented the original 2018 dataset [3]. In order to provide a unified overview of the entire dataset, this paper incorporates most of the original paper verbatim.

The following sections of this paper provide background information, detail the data format, describe the OhioT1DM Viewer visual-

ization software, and tell how to obtain the OhioT1DM Dataset and Viewer for research purposes.

2 BACKGROUND

We have been working on intelligent systems for diabetes management since 2004 [1, 4, 5, 6, 7, 8, 10, 11]. As part of our work, we have run five clinical research studies involving subjects with type 1 diabetes on insulin pump therapy. Over 50 anonymous subjects have provided blood glucose, insulin, and life-event data so that we could develop software intended to help people with diabetes and their professional health care providers.

Our most recent study was designed so that de-identified data could be shared with the research community. All data contributors to the OhioT1DM Dataset signed informed consent documents allowing us to share their de-identified data with outside researchers. This agreement clearly delineated what types of data could be shared and with whom. The data in the dataset was fully de-identified according to the Safe Harbor method, a standard specified by the Health Insurance Portability and Accountability Act (HIPAA) Privacy Rule [9]. To protect the data contributors and to ensure that the data is used only for research purposes, a Data Use Agreement (DUA) must be executed before a researcher can obtain the data.

3 OhioT1DM DATA FORMAT

For each data contributor, there is one XML file for training and development data and a separate XML file for testing data. This results in a total of 24 XML files, two for each of the 12 contributors. Table 2 shows the number of training and test examples for each contributor.

Table 2 also indicates the BGLP Challenge for which the data was released. For the 2018 BGLP Challenge, the number of test examples was equal to the number of data points in the XML testing file. However, for the 2020 BGLP Challenge, the first hour of data in each XML testing file is excluded from the set of points used for evaluation. This is to allow unbiased comparison of prediction models using all training data to predict each test point, as the first test points would otherwise be too close chronologically to the training data. Thus, for the 2020 BGLP Challenge, there are 12 more data points in each XML testing file than the number of test examples shown in Table 2.

Table 2. Number of training and test examples per data contributor

ID	BGLP Challenge	Training Examples	Test Examples
540	2020	11947	2884
544	2020	10623	2704
552	2020	9080	2352
567	2020	10858	2377
584	2020	12150	2653
596	2020	10877	2731
559	2018	10796	2514
563	2018	12124	2570
570	2018	10982	2745
575	2018	11866	2590
588	2018	12640	2791
591	2018	10847	2760

Each XML file contains the following data fields:

1. <**patient**> The patient ID number and insulin type. Weight is set to 99 as a placeholder, as actual patient weights are unavailable.
2. <**glucose_level**> Continuous glucose monitoring (CGM) data, recorded every 5 minutes.
3. <**finger_stick**> Blood glucose values obtained through self-monitoring by the patient.
4. <**basal**> The rate at which basal insulin is continuously infused. The basal rate begins at the specified timestamp *ts*, and it continues until another basal rate is set.
5. <**temp_basal**> A temporary basal insulin rate that supersedes the patient's normal basal rate. When the value is 0, this indicates that the basal insulin flow has been suspended. At the end of a temp_basal, the basal rate goes back to the normal basal rate, <basal>
6. <**bolus**> Insulin delivered to the patient, typically before a meal or when the patient is hyperglycemic. The most common type of bolus, normal, delivers all insulin at once. Other bolus types can stretch out the insulin dose over the period between *ts_begin* and *ts_end*.
7. <**meal**> The self-reported time and type of a meal, plus the patient's carbohydrate estimate for the meal.
8. <**sleep**> The times of self-reported sleep, plus the patient's subjective assessment of sleep quality: 1 for Poor; 2 for Fair; 3 for Good.
9. <**work**> Self-reported times of going to and from work. Intensity is the patient's subjective assessment of physical exertion, on a scale of 1 to 10, with 10 the most physically active.
10. <**stressors**> Time of self-reported stress.
11. <**hypo_event**> Time of self-reported hypoglycemic episode. Symptoms are not available, although there is a slot for them in the XML file.
12. <**illness**> Time of self-reported illness.
13. <**exercise**> Time and duration, in minutes, of self-reported exercise. Intensity is the patient's subjective assessment of physical exertion, on a scale of 1 to 10, with 10 the most physically active.
14. <**basis_heart_rate**> Heart rate, aggregated every 5 minutes. This data is only available for people who wore the Basis Peak sensor band.
15. <**basis_gsr**> Galvanic skin response, also known as skin conductance or electrodermal activity. For those who wore the Basis Peak, the data was aggregated every 5 minutes. Despite this attribute's name, it is also available for those who wore the Empatica Embrace. For these individuals, the data is aggregated every 1 minute.
16. <**basis_skin_temperature**> Skin temperature, in degrees Fahrenheit, aggregated every 5 minutes, for those who wore the Basis Peak, and every 1 minute, for those who wore the Empatica Embrace.
17. <**basis_air_temperature**> Air temperature, in degrees Fahrenheit, aggregated every 5 minutes. This data is only available for people who wore the Basis Peak sensor band.
18. <**basis_steps**> Step count, aggregated every 5 minutes. This data is only available for people who wore the Basis Peak sensor band.
19. <**basis_sleep**> Times when the sensor band reported that the subject was asleep. For those who wore the Basis Peak, there is also a numeric estimate of sleep quality.
20. <**acceleration**> Magnitute of acceleration, aggregated every 1 minute. This data is only available for people who wore the Empatica Embrace sensor band.

Note that, in de-identifying the dataset, all dates for each individual were shifted by the same random amount of time into the future. The days of the week and the times of day were maintained in the new timeframes. However, the months were shifted, so that it is not possible to consider the effects of seasonality or of holidays.

4 THE OhioT1DM VIEWER

The OhioT1DM Viewer is a visualization tool that opens an XML file from the OhioT1DM Dataset and graphically displays the integrated data. It aids in developing intuition about the data and also in debugging. For example, if a system makes a poor blood glucose level prediction at a particular point in time, viewing the data at that time might illuminate a cause. For example, the subject might have forgotten to report a meal or might have been feeling ill or stressed.

Figure 1 shows a screenshot from the OhioT1DM Viewer. The data is displayed one day at a time, from midnight to midnight. Controls allow the user to move from day to day and to toggle any type of data off or on for targeted viewing.

The bottom pane shows blood glucose, insulin, and self-reported life-event data. CGM data is displayed as a mostly blue curve, with green points indicating hypoglycemia. Finger sticks are displayed as red dots. Boluses are displayed along the horizontal axis as orange and yellow circles. The basal rate is indicated as a black line. Temporary basal rates appear as red lines. Self-reported sleep is indicated by blue regions. Life-event icons appear at the top of the pane as dots, squares, and triangles. The data in the bottom pane is clickable, so that additional information about any data point can be displayed. For example, clicking on a meal (a square blue icon) displays the timestamp, type of meal, and carbohydrate estimate.

The top pane displays sensor band data. Blue regions in the top pane are times the sensor band detected that the wearer was asleep. The step count is indicated by vertical blue lines. The curves show heart rate (red), galvanic skin response (green), skin temperature (gold), air temperature (cyan), and magnitude of acceleration (black).

5 OBTAINING THE DATASET AND VIEWER

The original 2018 OhioT1DM Dataset and the OhioT1DM Viewer are now available to researchers. The full 2020 OhioT1DM Dataset is currently being released to participants in the second BGLP Challenge. The second BGLP Challenge will take place June 9, 2020, in conjunction with the 5th International Workshop on Knowledge Discovery in Healthcare Data at ECAI 2020, in Santiago de Compostela, Spain.

After the completion of the BGLP Challenge, the entire dataset will be made available to other researchers. To protect the data contributors and to ensure that the data is used only for research purposes, a Data Use Agreement (DUA) is required. A DUA is a binding document signed by legal signatories of Ohio University and the researcher's home institution. As of this writing, researchers can request a DUA at http://smarthealth.cs.ohio.edu/OhioT1DM-dataset.html. Once a DUA is executed, the OhioT1DM Dataset and Viewer will be directly released to the researcher.

6 CONCLUSION

The OhioT1DM Dataset was developed to promote and facilitate research in blood glucose level prediction. Accurate blood gluocose level predictions could positively impact the health and well-being of people with diabetes. In addition to their role in the artificial pancreas project, such predictions could also enable other beneficial applications, such as decision support for avoiding impending problems, "what if" analyses to project the effects of different lifestyle choices, and enhanced blood glucose profiles to aid in individualizing diabetes care. It is our hope that sharing this dataset will help to advance the state of the art in blood glucose level prediction.

ACKNOWLEDGEMENTS

This work was supported by grant 1R21EB022356 from the National Institutes of Health (NIH). The OhioT1DM Viewer was originally implemented by Hannah Quillin and Charlie Murphy, and further refined by Robin Kelby and Jeremy Beauchamp. The authors gratefully acknowledge the contributions of Emeritus Professor of Endocrinology Frank Schwartz, MD, a pioneer in building intelligent systems for diabetes management. We would also like to thank our physician collaborators, Aili Guo, MD, and Amber Healy, DO, our research nurses, Cammie Starner and Lynn Petrik, and our past and present graduate and undergraduate research assistants. We are especially grateful to the 12 anonymous individuals with type 1 diabetes who shared their data, enabling the creation of this dataset.

REFERENCES

[1] R. Bunescu, N. Struble, C. Marling, J. Shubrook, and F. Schwartz, 'Blood glucose level prediction using physiological models and support vector regression', in *Proceedings of the Twelfth International Conference on Machine Learning and Applications (ICMLA)*, pp. 135–140. IEEE Press, (2013).

[2] Juvenile Diabetes Research Foundation (JDRF). Artificial Pancreas, 2019. Available at http://www.jdrf.org/impact/research/artificial-pancreas/, accessed January, 2020.

[3] C. Marling and R. Bunescu, 'The OhioT1DM dataset for blood glucose level prediction', in *The 3rd International Workshop on Knowledge Discovery in Healthcare Data*, Stockholm, Sweden, (July 2018). Available at http://ceur-ws.org/Vol-2148/paper09.pdf, accessed January, 2020.

[4] C. Marling, J. Shubrook, and F. Schwartz, 'Toward case-based reasoning for diabetes management: A preliminary clinical study and decision support system prototype', *Computational Intelligence*, **25**(3), 165–179, (2009).

[5] C. Marling, M. Wiley, R. Bunescu, J. Shubrook, and F. Schwartz, 'Emerging applications for intelligent diabetes management', *AI Magazine*, **33**(2), 67–78, (2012).

[6] C. Marling, L. Xia, R. Bunescu, and F. Schwartz, 'Machine learning experiments with noninvasive sensors for hypoglycemia detection', in *IJCAI 2016 Workshop on Knowledge Discovery in Healthcare Data*, New York, NY, (2016).

[7] S. Mirshekarian, R. Bunescu, C. Marling, and F. Schwartz, 'Using LSTMs to learn physiological models of blood glucose behavior', in *Proceedings of the 39th International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC 2017)*, Jeju Island, Korea, (2017).

[8] S. Mirshekarian, H. Shen, R. Bunescu, and C. Marling, 'LSTMs and neural attention models for blood glucose prediction: Comparative experiments on real and synthetic data', in *Proceedings of the 41st International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC 2019)*, Berlin, Germany, (2019).

[9] Office for Civil Rights. Guidance regarding methods for de-identification of protected health information in accordance with the Health Insurance Portability and Accountability Act (HIPAA) privacy rule, 2012. Available at https://www.hhs.gov/sites/default/files/ocr/privacy/hipaa/understanding/coveredentities/De-identification/hhs_deid_guidance.pdf, accessed January, 2020.

[10] K. Plis, R. Bunescu, C. Marling, J. Shubrook, and F. Schwartz, 'A machine learning approach to predicting blood glucose levels for diabetes management', in *Modern Artificial Intelligence for Health Analytics: Papers Presented at the Twenty-Eighth AAAI Conference on Artificial Intelligence*, pp. 35–39. AAAI Press, (2014).

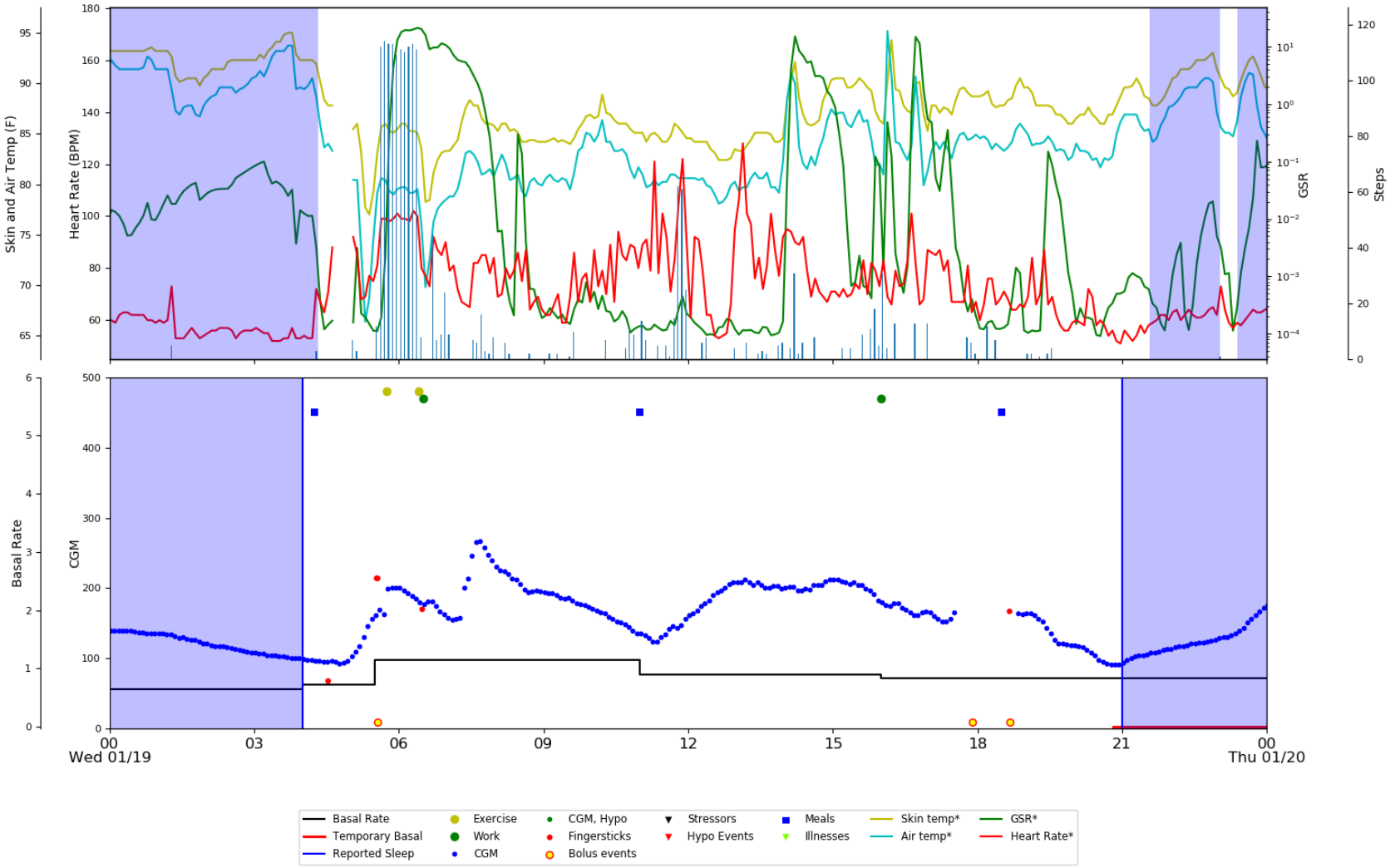


Figure 1. Screenshot from the OhioT1DM Viewer

[11] F. L. Schwartz, J. H. Shubrook, and C. R. Marling, 'Use of case-based reasoning to enhance intensive management of patients on insulin pump therapy', *Journal of Diabetes Science and Technology*, **2**(4), 603–611, (2008).

A Personalized and Interpretable Deep Learning Based Approach to Predict Blood Glucose Concentration in Type 1 Diabetes

Giacomo Cappon[1], Lorenzo Meneghetti[1], Francesco Prendin[1], Jacopo Pavan[1], Giovanni Sparacino[1], Simone Del Favero[1], Andrea Facchinetti[1]

Abstract.

The management of type 1 diabetes mellitus (T1DM) is a burdensome life-long task. In fact, T1DM individuals are request to perform every day tens of actions to adapt the insulin therapy, aimed at maintaining the blood glucose (BG) concentration as much as possible into a safe range coping with the day-to-day variability of their life style. The recent availability of continuous glucose monitoring (CGM) devices and other low-cost wearable sensors to track important vital and activity signals, is stimulating the development of decision support systems to lower this burden. Modern deep learning models, trained using rich amount of information, are a suitable and effective instrument for such purpose, especially if used to predict future BG values. However, the high accuracy of deep learning approaches is often obtained at the expense of less interpretability.

To surpass this limit, in this work we propose a new deep learning method for BG prediction based on a personalized bidirectional long short-term memory (LSTM) equipped with a tool that enables its interpretability. The OhioT1DM Dataset was used to develop a model targeting future BG at 30 and 60 minute prediction horizons (PH). The accuracy of model predictions was evaluated in terms of root mean square error (RMSE), mean absolute error (MAE), and the time gained (TG) to anticipate the actual glucose concentration.

The obtained results show fairly good prediction accuracy (for PH = 30/60 min): RMSE = 20.20/34.19 mg/dl, MAE = 14.74/25.98 mg/dl, and TG = 9.17/18.33 min. Moreover, we showed, in a representative case, that our algorithm is able to preserve the physiological meaning of the considered inputs.

In conclusion, we built a model able to provide reliable glucose performance ensuring the interpretability of its output. Future work will assess model performance against other competitive strategies.

1 INTRODUCTION

Diabetes is a chronical metabolic disease in which patients are no longer able to effectively control blood glucose (BG) concentration [2]. In particular, type 1 diabetes mellitus (T1DM) is characterized by an autoimmune attack on the pancreatic β-cells resulting to impaired insulin production. As a consequence, people with T1DM are required to manage their glycemia to keep it within the safe range (i.e. BG $\in$ [70, 180] mg/dl) without incurring in dangerous complications induced by hypoglycemia (BG $<$ 70 mg/dl) and hyperglycemia (BG $>$ 180 mg/dl). Such a burdensome process can be eased by integrating in T1DM therapy newly developed decision support algorithms [15] [3]. Specifically, methodologies based on deep learning aimed to predict future BG levels [6] represent a unique way to equip people with T1DM with an effective tool to proactively tackle the shortcoming of adverse events.

The increasing amount of data that can be easily collected by sensors continuously monitoring BG levels (CGM), insulin infusion, and physical activity, just to mention a few, enables researchers to build new BG prediction algorithms that are effective, personalized, and able to empower T1DM management [4]. In particular, in 2020, Marling et al. [13] started the second edition of the Blood Glucose Level Prediction (BGLP) Challenge, i.e., an open competition aimed to promote and facilitate research in this field. Alongside with the competition, the second version of the so-called OhioT1DM Dataset was released. In particular, by including CGM recordings, insulin infusion logs, daily event reporting, and patient vital parameters' monitoring, this dataset represents a unique source of data that can be used for the purpose.

In this paper, we present a new BG level prediction method based on deep learning that we developed and submitted to the second BGLP Challenge. Specifically, given the complexity of the problem at hand and the "temporal" nature of the feature set, here we trained a long-short term memory (LSTM) [11] neural network targeting future BG levels. Even if recurrent neural networks such as LSTMs are known to achieve good performance for the specific task of BG prediction [14], they lack of interpretability. In fact, when developing models for T1DM decision support, there is the need of providing transparent models able to produce reliable but also interpretable predictions [1]. To the best of our knowledge, current state-of-the-art algorithms for BG prediction based on LSTMs have never been interpreted to explain the model "rationale" behind its outcomes. As such, the aim being equipping our model with this feature, we exploited SHapley Additive exPlanations (SHAP), i.e., a newly developed approach to interpret deep learning model predictions [12]. This represents a novelty in the field and offers useful insights on the use of recurrent neural networks for T1DM management.

2 DATASET PREPARATION

2.1 Dataset description and preprocessing

The model was trained and evaluated on data obtained from the updated OhioT1DM Dataset developed by Marling et al. [13]. In the specific, data from 6 people with T1DM were provided. These

[1] University of Padova, Department of Information Engineering, Padova Italy, email: {cappongi, meneghet, prendinf, pavanjac, gianni, sdelfave, facchine}@dei.unipd.it

anonymous people (numbered as 540, 544, 552, 567, 584, and 596) wore Medtronic 530G and 630G insulin pumps and Medtronic Enlite CGM sensors during an 8-week data-collection period. They reported their meals and other life-event data (time of exercise, sleep, work, stress, and illness) via a custom smartphone app. Furthermore, additional physiological data were collected by a Empatica fitness band, including galvanic skin response, skin temperature, and magnitude of acceleration.

In the training dataset, several intervals of missing values were observed. Such discontinuities reduce the number of training data available but also compromise the dynamical structure of the data, thus causing a bad impact on the training procedure. Because of this, a first order interpolation was performed, on the training set only, on the missing portions that were shorter than 30 minutes.

The data were re-sampled onto a uniform time grid with regular intervals of 5 minutes for training and testing the model. Each sample is placed in the new grid at the closest timestamp with respect to its original timestamp. The final prediction obtained was then realigned to the original timestamps by reassigning every predicted sample to the original timestamps, inverting the re-sampling procedure.

2.2 Feature extraction

Deep learning models, such as the one used in this work, are able to deal with raw data without resorting to manual feature engineering. However, this is in general true when large amount of data are used for their training. Therefore, given the limited size of the dataset at hand, we resorted to manual feature engineering. This is furtherly substantiated by several tests that we performed during our study (not reported here for the sake of simplicity), which confirmed that, using the extracted features described in the following, we were able to improve model performance.

An initial observation of the data revealed that the information registered by the fitness band were partial or incomplete in the majority of the people. Therefore, we decided to discard these signals. As such, along with the CGM measurements, we considered the following signals as input to our predictive algorithm: the injected insulin as reported by the pump, the reported meals and the self-reported physical exercise.

Since whenever a meal is consumed, an insulin bolus is injected to counter the post-prandial hyperglycemic excursion the two signals (meals and insulin) tend to be highly correlated. Therefore, to try to overcome this problem, we generated a new signal consisting of only the correction boluses (INS_C), determined as the injections of insulin that are administered at a time of minimum 90 minutes after a meal.

A consumed meal or an injected insulin bolus do not impact the BG levels immediately. Instead, their effect can only be observed after a minimum time of 30-60 minutes. Similarly, the impact of physical activity has a delayed effect on the BG levels [16]. Because of this, the signals of injected insulin (INS), INS_C, reported meals (MEA) and physical activity (PA) are transformed to better account for the underlying physiological dynamics. The transformation consisted of a 2^{nd} order low-pass filtering with impulse response $h(t) = \lambda t e^{-\lambda t}$, where we set λ=0.02. This procedure has been adopted in literature to produce feature sets for the development of ML algorithms for T1DM decision support [3] [15]. Additionally, a transformation of the CGM signal is obtained using the dynamic risk [7], which empowers the model with additional features that capture the dynamics of the CGM signal (e.g., glycemic variability).

In summary, the following features were considered: CGM, DR, INS, INS_C, MEA, and PA.

3 METHODS

3.1 A Bidirectional LSTM to Predict Future BG

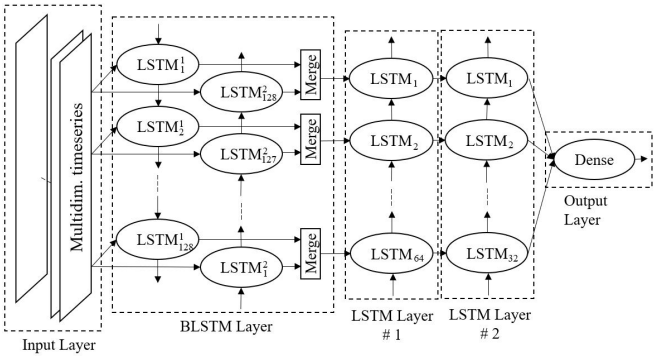


Figure 1. Scheme of the implemented bidirectional LSTM.

As introduced, BG level prediction is a very challenging and complex task. By analyzing the nature of our dataset, it is natural to think that proper modeling of the temporal between-feature dependencies is crucial to effectively solve the problem at hand. For this reason, in this work we decided to adopt an LSTM-based model architecture since LSTMs are well-known in the literature to be the ideal choice to build a predictive model for time series [9]. An LSTM consists of a set of recurrently connected blocks, known as LSTM memory cells. Each LSTM cell consists of an input gate, an output gate, and a forget gate. Each of the three gates can be thought of as a neuron, and each gate achieves a particular function in the cell. In particular, LSTMs are able to exploit learned temporal dependencies to predict the future output according to their previous states, thus well-fitting the purpose of this work. A common drawback of LSTM networks is that, by processing the input in a temporal order, they tend to produce as output, something that is strongly based on forwards dependencies only. To solve this issue, a bidirectional LSTM can be exploited [8]. Briefly, it consists of presenting, to two parallel LSTMs, each training sequence forwards and backwards and then merging the LSTMs outputs to obtain the resulting target estimate. As such, this allows to learn potentially richer representations and capture patterns that may have been missed by the chronological-order version alone. Moreover, the use of bidirectional LSTMs for BG level prediction allowed to obtained promising results in several seminal works [17][18].
The final model architecture, shown in Figure 1 and hereafter labeled as BLSTM, consists of a four-layer neural network: a bidirectional LSTM input layer composed of 128 cells having a look back period of 15 minutes (i.e. 3 samples), two LSTM layers respectively composed of 64 and 32 cells, and a fully connected layer consisting of a single neuron computing the BG level prediction at two different prediction horizons (PH), i.e. 30 and 60 min. BLSTM architecture, hyperparameters, and look back period have been chosen by trial-and-error to compromise between model complexity and accuracy. The BLSTM is implemented in Python using the Keras library [5].

3.2 Equipping BLSTM with interpretability

New algorithms for decision support in T1D management require to be interpretable [1] to avoid potentially adverse or even life-threatening consequences. Unlike traditional physiological-based strategies, deep learning models (such as LSTMs) are black-boxes,

meaning that their high accuracy is often achieved by learning complex relationships that even experts struggle to interpret. For black box models to be adopted in the field of T1D, it is thus desirable to understand whether or not they retain the physiological significance of the inputs they use.

In this work, we aim to overcome the issue of interpretability by analysing our BLSTM with a novel unified approach to interpret model predictions, SHAP [12]. SHAP is a newly developed game theoretical approach to explain how much a given feature impacts on model prediction (compared to if we made that prediction at some baseline value of that feature). By this method, we were able to fully interpret the BLSTM. Indeed, SHAP allowed to both visualize the feature importance and what is driving it.

3.3 Software framework

For each subject and considered PH we trained, thus personalized, a different BLSTM model. The training of each BLSTM has been performed through the gradient descent RMSprop algorithm applied in a mini-batch mode [10]. In particular, as schematized in Figure 2, we developed an *ad-hoc* software framework to automatically perform both model training and tuning. In details, in block A, the

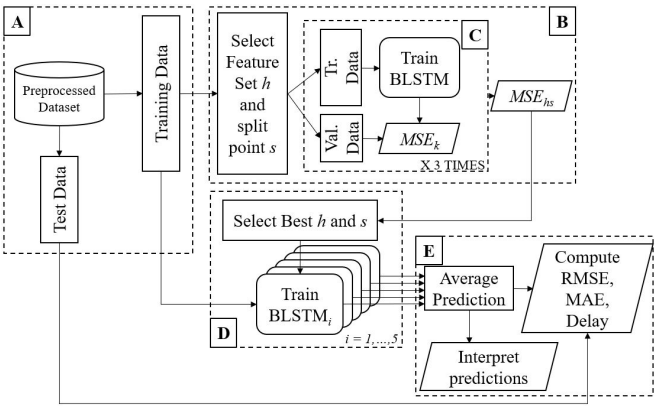


Figure 2. Scheme of the experimental framework.

preprocessed data have been divided into training and test data, respectively. Then, in block B, to optimally tune the BLSTM, feature selection is performed. To do so, we generated the power set of $S = \{\text{DR}, \text{INS}, \text{INS}_C, \text{CHO}, \text{PA}\}$, i.e. the set of all subsets of S, including the empty set and S itself. Then, given its obvious impact on model performance, we constrained each feature subset in the power set to also contain the CGM feature. As a result, we exhaustively examined all the possible sub-sets of features, each containing CGM and other, possibly useful, input features. Block B also splits data into training and validation set. Here, we explored multiple "split points" s, thus assigning $\{50, 60, 70, 80\}\%$ of the data to the training data and the remaining $\{50, 40, 30, 20\}\%$ to the validation data (used to early stop the training of BLSTM in block C to avoid overfitting). For each feature set h in the above-described power set and each considered split point s, the performance of the BLSTM is assessed in terms of mean squared error (MSE_{hs}). To prevent such evaluation from being affected by the random initialization of the BLSTM weights, the whole training and evaluation process is repeated, in block C, three times per feature set. In turn, for each feature set h and split point s we computed MSE_{hs} as:

$$MSE_{hs} = \frac{1}{3}\sum_{k=1}^{3} MSE_k \tag{1}$$

where subscript $k = 1, \ldots, 3$ refers to the repetition at hand. In block D, the best feature set h and split point s are selected as the h and s that obtained the minimum MSE_{hs}. Then, five BLSTMs, namely BLSTM$_i$ $i = 1, \ldots, 5$ are trained on the entire patient/prediction horizon-specific training set. Finally, in block E, we evaluated the model performance by comparing the true BG values in the test set against the respective predictions obtained by averaging each BLSTM$_i$ estimate and we interpret model predictions through SHAP.

4 ASSESSMENT OF BLSTM PERFORMANCE

For the BGLP challenge, the considered metrics for evaluating the accuracy of the obtained prediction are the Root Mean Squared Error, (RMSE) and the Mean Absolute Error (MAE). Considering the prediction error $e(n) = y(n) - \hat{y}(n)$, where $y(n)$ and $\hat{y}(n)$ are the CGM measurements and the computed prediction, respectively, the RMSE and MAE are obtained as follows:

$$RMSE = \sum_N \sqrt{e(t)^2}, \qquad MAE = \text{mean}(|e(t)|)$$

where N is the number of total points.

In this paper, we considered an additional performance metric: the Time Gain (TG), which quantifies the time gained thanks to the prediction. A measure of the average TG is obtained as:

$$TG(y, \hat{y}) = PH - \text{delay}(y, \hat{y})$$

where PH is the prediction horizon used to perform the prediction $\hat{y}$ and the delay$(y, \hat{y})$ between the original and the predicted profiles quantified by the temporal shift k that minimizes the distance between y and $\hat{y}$:

$$\text{delay}(y, \hat{y}) = \underset{k}{\text{argmin}} \sum_{i=1}^{N} \left(y(i) - \hat{y}(i-k)\right)^2$$

5 RESULTS

5.1 BLSTM performance in terms of prediction accuracy

In Figure 3, we present an example of the prediction obtained on a representative subject (544). In the top panel, we report in blue the actual CGM measurements and in red the prediction performed by the BLSTM; in the bottom panel, we report the consumed meals (in grams) as reported by the subject. Albeit affected by the CGM signal noise, the prediction is able to follow the CGM measurements during the post-prandial rises with minor delay. Predicting hypoglycemic episodes with high accuracy resulted to be one of the harder task (an example of inaccurate prediction can be seen at around 12:20). A possible explanation for this is that hypoglycemic episodes are sporadic events which do not happen often, therefore the BLSTM may not have enough training data to learn how to predict similar patterns occurring in the test set.

In Table 1, we report the optimal feature sets that were identified on the training set, in block C, for each subject and PH. The

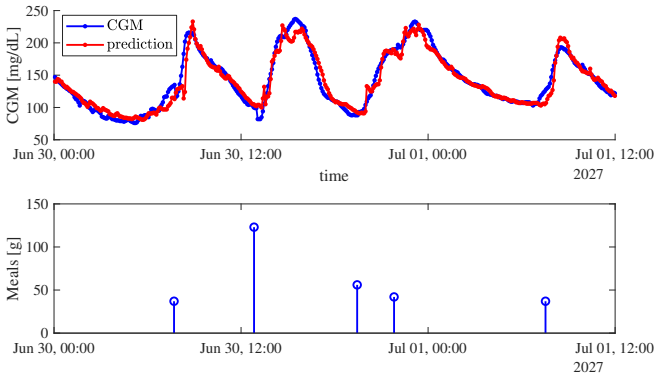


Figure 3. Example of prediction obtained on subject 596. In the top panel, the CGM measurements (blue) and the respective prediction (red). In the bottom panel, the consumed meals reported by the patient.

Table 1. Optimal feature set selected on the training set in block C.

ID	PH = 30 min	PH = 60 min
540	CGM, INS	CGM, DR, INS
544	CGM, DR, INS, MEA, INS_C	CGM, MEA, INS_C
552	CGM, INS, MEA	CGM, INS, INS_C
567	CGM, DR, INS	CGM, INS, PA
584	CGM, DR, INS_C	CGM, INS_C
596	CGM, INS, MEA	CGM, INS, MEA

CGM feature is included in every set by default as described earlier in Section 3.3. The feature INS is adopted in almost every case, expect some where it is replaced by the feature INS_C. The feature MEA is adopted less often, especially in patients where we observed a lower consistency in reporting meals. The feature PA was selected only once, denoting its limited effectiveness in improving the performance of the BLSTM. In general, different PH lead to different features sets for the same patient. This is due to the fact that some features, e.g. MEA, might be relevant, in a specific patient, for PH = 30 min and not for PH = 60 min given their impact on BG level in the very short-term.

In Table 2 we report the RMSE, the MAE and the TG obtained for each subject and PH. A mean RMSE = 20.20 mg/dl is obtained

Table 2. Results obtained on the test-set.

	PH = 30 min			PH = 60 min		
ID	RMSE	MAE	TG	RMSE	MAE	TG
540	23.19	17.33	10	41.41	31.77	20
544	18.88	13.23	15	31.06	22.54	30
552	17.97	13.50	10	31.20	24.48	20
567	21.18	15.20	10	37.40	28.50	20
584	21.91	16.38	5	35.95	27.59	5
596	18.09	12.81	5	28.13	20.99	15
mean	20.20	14.74	9.17	34.19	25.98	18.33

for PH = 30 min, together with a value of MAE = 14.74 mg/dl and TG = 9.17 min. For PH = 60 min, a mean RMSE = 34.19 mg/dl was obtained, together with MAE = 25.98 mg/dl and TG = 18.33 min.

Table 3 reports the number of samples predicted per patient and the percentage of predicted samples over the total CGM samples available. Except for one case, the BLSTM was able to compute a prediction for more than 90% of the samples.

Table 3. Predicted samples per subject.

		PH = 30 min		PH = 60 min	
ID	Total samples	predicted	%	predicted	%
540	2884	2820	97.78	2697	93.52
544	2704	2586	95.64	2638	97.56
552	2352	2275	96.73	2235	95.03
567	2377	2157	90.74	2232	93.90
584	2653	2354	88.73	2473	93.22
596	2731	2683	98.24	2647	96.92

5.2 Model interpretation

As discussed in Section 3.2, thanks to SHAP we are able to interpret each trained BLSTM. The plot in Figure 4 reports the application of SHAP to the BLSTM obtained for patient 596 for a PH of 60 min. This plot is made of many dots. Each dot has three characteristics:

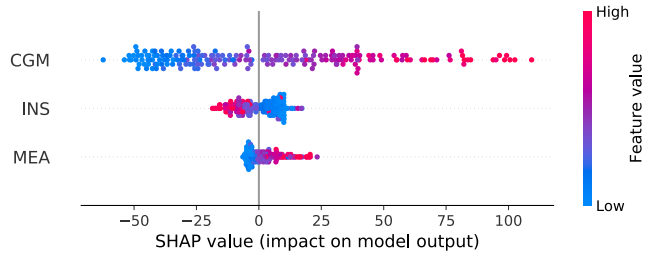


Figure 4. Impact of each input feature on model output obtained via SHAP in patient 596 with PH = 60 min.

vertical location shows what feature it is depicting, the color shows whether that feature assumed an high or low value for that row of the dataset, horizontal location shows whether the effect of that value caused a higher or lower prediction of future BG levels. Results show that high values of CGM translate in high predicted CGM values. On the other hand, high INS impacts negatively on model output mirroring the actual impact of insulin on BG dynamics. Parallely, high MEA induces an increase on predicted glucose values, correctly accounting for the effect of meal intakes on BG level. As such, the physiological meaning of all input features is preserved by the considered representative BLSTM.

For brevity, we do not report the results obtained on other patients, being very similar and consistent with that previously showed.

6 CONCLUSION

The possibility of collecting important vital and activity signals from low-cost wearable sensors in patients with T1DM is calling for the development of individualized proactive decision support systems to lower the daily burden in the application of BG control therapy. In this work, the aim being providing patients with reliable BG predictions, we leveraged the OhioT1DM Dataset to build a new deep learning-based approach for the scope that we submitted for the second edition of the BGLP Challenge. The novelty here is that, beside obtaining fairly good BG predictions considering both a 30 min and a 60 min-long PH, our algorithm is also interpretable. Indeed, the integration of SHAP in our procedure allowed to obtain a "transparent" model where the impact of each feature on model output is explicitly expressed.

The presented study has some limitations that need to be addressed in future work. In particular, we will concentrate on two main issues. First, to fully evaluate its performance, BLSTM will be assessed against other competing baseline and state-of-the-art BG prediction methodologies, e.g., neural networks, random forests, and vanilla LSTMs. Then, we will tackle the limitation represented by the dataset length. In fact, methodologies like LSTMs usually benefit from having more data to be used for their training and tuning. For this purpose, we will investigate the potential advantage of using longer datasets on BLSTM performance.

ACKNOWLEDGEMENTS

Part of this work was supported by MIUR (Italian Minister for Education) under the initiative "Departments of Excellence" (Law 232/2016).

CODE

A repository of the code used in this paper is available online [2].

REFERENCES

[1] M. A. Ahmad, C. Eckert, A. Teredesai, and G. McKelvey, 'Interpretable machine learning in healthcare', *2018 IEEE International Conference on Healthcare Informatics (ICHI), New York, NY*, 447, (2019).

[2] American Diabetes Association, 'Diagnosis and classification of diabetes mellitus: : Standards of medical care in diabetes', *Diabetes Care*, **43**, S14–S31, (2020).

[3] G. Cappon, A. Facchinetti, G. Sparacino, P. Georgiou, and P. Herrero, 'Classification of postprandial glycemic status with application to insulin dosing in type 1 diabetes—an in silico proof-of-concept', *Sensors*, **19**(14), 3168, (2019).

[4] G. Cappon, M. Vettoretti, G. Sparacino, and A. Facchinetti, 'Continuous glucose monitoring sensors for diabetes management: A review of technologies and applications', *Diabetes & metabolism journal*, **43**(4), 383–397, (2019).

[5] F. Chollet. Keras. `https://keras.io`, 2015.

[6] I. Contreras and J. Vehi, 'Artificial intelligence for diabetes management and decision support: Literature review.', *Journal of Medical Internet Research*, **20**, e10775, (2018).

[7] C. Fabris, S.D. Patek, and M.D. Breton, 'Exercise and glucose metabolism in persons with diabetes mellitus: perspectives on the role for continuous glucose monitoring', *Journal of Diabetes Science and Technology*, **14**(1), 50–59, (2015).

[8] A. Graves, S. Fernandez, and J. Schmidhuber, 'Bidirectional lstm networks for improved phoneme classification and recognition', *Artificial Neural Networks: Formal Models and Their Applications – ICANN 2005*, 799–804, (2015).

[9] A. Graves, A.R. Mohamed, and G. Hinton, 'Speech recognition with deep recurrent neural networks', *IEEE International Conference on Acoustics, Speech and Signal Processing*, 6645–6649, (2013).

[10] G. Hinton, N. Srivastava, and K. Swersky, 'Overview of mini-batch gradient descent.', *Neural Network for Machine Learning. Coursera Course. Available at: http://www.cs.toronto.edu/ tijmen/csc321/ slides/lecture_slides_lec6.pdf*, (2017).

[11] S. Hochreiter and J. Schmidhuber, 'Long short term memory', *Neural Computation*, **9**, 1735–1780, (1997).

[12] S.M. Lundberg and S.I. Lee, 'A unified approach to interpreting model predictions', *Advances in neural information processing system.*, **30**, 4765–4774, (2017).

[13] C. Marling and R. Bunescu, 'The ohiot1dm dataset for blood glucose level prediction: Update 2020.', *in The 5th International Workshop on Knowledge Discovery in Healthcare Data*, (2020).

[14] J. Martinsson, A. Schliep, B. Eliasson, and O. Mogren, 'Blood glucose prediction with variance estimation using recurrent neural networks', *Journal of Healthcare Informatics Research*, **4**, 1–18, (2020).

[15] L. Meneghetti, G.A. Susto, and S. Del Favero, 'Detection of insulin pump malfunctioning to improve safety in artificial pancreas using unsupervised algorithms', *Journal of Diabetes Science and Technology*, **13**(6), 1065–1076, (2019).

[16] M. Riddell and B.A. Perkins, 'Exercise and glucose metabolism in persons with diabetes mellitus: perspectives on the role for continuous glucose monitoring', *Journal of Diabetes Science and Technology*, **3**(4), 914–923, (2009).

[17] Q. Sun, M. V. Jankovic, L. Bally, and S. G. Mougiakakou, 'Predicting blood glucose with an lstm and bi-lstm based deep neural network', *in 14th Symposium on Neural Networks and Applications (NEUREL)*, 1–5, (2018).

[18] T. Wang and W. Li, 'Blood glucose forecasting using lstm variants under the context of open source artificial pancreas system', *in 53rd Hawaii International Conference on System Sciences*, 3256–3563, (2020).

[2] https://github.com/meneghet/BGLP_challenge_2020

Neural Multi-class Classification Approach to Blood Glucose Level Forecasting with Prediction Uncertainty Visualisation

Michael Mayo[1] and **Tomas Koutny**[2]

Abstract. A machine learning-based method for blood glucose level prediction thirty and sixty minutes in advance based on highly multiclass classification (as opposed to the more traditional regression approach) is proposed. An advantage of this approach is the possibility of modelling and visualising the uncertainty of a prediction across the entire range of blood glucose levels without parametric assumptions such as normality. To demonstrate the approach, a long-short term memory-based neural network classifier is used in conjunction with a blood glucose-specific data preprocessing technique (risk domain transform) to train a set of models and generate predictions for the 2018 and 2020 Blood Glucose Level Prediction Competition datasets. Numeric accuracy results are reported along with examples of the uncertainty visualisation possible using this technique.

1 INTRODUCTION AND BACKGROUND

Maintaining blood glucose level (BGL) in the normoglycemic range is a significant challenge for patients with type 1 diabetes (T1D). Traditionally, patient BGL self-management is achieved using finger stick blood samples, testing strips and glucose meters (see [13] for an overview), combined with bolus insulin dosing to approximate proper insulin delivery in the body of non-diabetic person. However, with the recent development of continuous glucose monitors (CGMs) and semi- and fully closed-loop artificial pancreas (AP) systems [14], much finer grained control of patient BGL is now possible. Additionally, significantly greater volumes of BGL data is also available when these devices are used. AP technology has been show to improve patient outcomes [5].

In this paper, the problem of forecasting BGL thirty and sixty minutes in advance is considered using the 2020 BGL Prediction Challenge [3] as a testbed. Although several past systems have considered machine learning techniques for BGL forecasting (see [16] for a comprehensive survey), most approaches take a regression approach to solving the problem. In other words, each "forecast" is a numeric point prediction (such as BGL at some point in the future), and overall system accuracy is a measurement of the error between the forecast and the actual future BGL. Accuracy metrics may be statistical (e.g. mean absolute error) or clinical (e.g. Clarke error grid analysis [12]). Regardless, the focus is usually on point predictions.

Here, an alternative approach is taken: instead of treating BGL forecasting as a regression problem, it is instead viewed as a classification problem. This is achieved by dividing the range of possible BGL values into 100 bins equally spaced in the risk domain [9]. Each bin is mapped to a class, and therefore a given forecast is generated by predicting the probability of each class and computing the expected value across all of the classes. An advantage of this approach is that the probabilities associated with the forecast can be visualised across the BGL range. This could be useful for patients, since it enables the patient to take the reliability of the forecast into consideration when making a decision. Additionally, the probability distribution can be used to estimate the chance of significant events such as hypoglycemic episode. Although a similar idea was explored in the context of regression recently [11], the underlying assumption there was that the uncertainty distribution was Gaussian, whereas in the classification approach presented here, no assumptions need be made about the distribution.

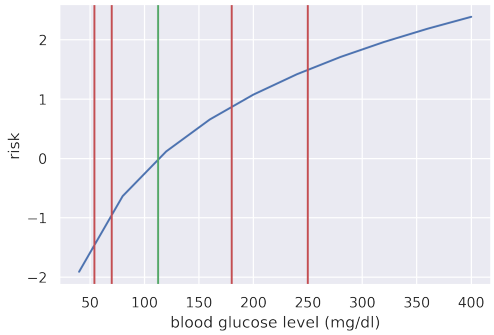


Figure 1. Risk function. The vertical green line at 112.5 mg/dl is the point of least risk while vertical red lines represent (from left to right) the thresholds [6] for level 2 and level 1 hypoglycemia, normoglycemia, and level 1 and level 2 hyperglycemia respectively.

In order to transform the problem of BGL forecasting into a classification problem, a method of breaking the BGL range into sensible classes is required. This is not trivial because the range of BGL values is continuous, and the sizes of clinically-relevant subranges varies non-linearly. For example, a small change in the hypoglycemic part of the BGL range may be highly significant clinically but an equivalent change in the hyperglycemic part of the range may be considered insignificant.

One option is to use the five ranges proposed by Danne et al. [6] as the classes. These ranges are: levels I and II hypoglycemic, normoglycemia, and levels I and II hyperglycemia. In this case, the size and split points of each range are defined. However, this would amount

[1] School of Computing and Mathematical Sciences, University of Waikato, New Zealand, email: michael.mayo@waikato.ac.nz
[2] NTIS – New Technologies for the Information Society, University of West Bohemia, Czech Republic, email: txkoutny@kiv.zcu.cz

to only five classes, and predictions in such a case may lack accuracy within a class. Another option is to arbitrarily divide the BGL range into a much larger number of bins (e.g. 100) which both increases the number of classes considerably (making the machine learning more challenging) but simultaneously increases the granularity of the predictions so that better probability distributions can be produced. In this paper, the latter approach is taken, however this in turn leads to the necessity to decide how the bins should be defined/split across the range of BGL values.

Because of the inherent non-linearity of the BGL range, an approach called the risk domain transformation, first proposed by Kovatchev et al. [9], is utilised. The idea is to define a non-linear transformation function (and by implication, its inverse) that shifts a CGM sensor reading from the blood glucose domain to a new "risk" domain that is better suited for subsequent analysis. This transformation function is illustrated by Figure 1. Also shown by the figure are the breakpoints for the five ranges defined by Danne et al. [6].

As can be observed in the figure, risk domain values typically spans a range from approximately -2 to just over 2, and most normoglycemic readings lie more or less in the range range $[-0.9, 0.9]$. A risk value of 0.0 corresponds to the BGL of 112.5 mg/dl, which is considered the point of least risk. An advantage of the risk domain is that the hypo- and hyperglycemic ranges now have equal size and significance, which reduces the chance of bias in statistical analysis (e.g. due to larger absolute error sizes in the hyperglycemic range).

$$r(x_t) = y_t = 1.509 \left(\texttt{log}(x_t)^{1.084} - 5.381 \right) \tag{1}$$

$$r^{-1}(y_t) = x_t = \texttt{exp} \left(\left(\frac{y_t}{1.509} + 5.381 \right)^{\frac{1}{1.084}} \right) \tag{2}$$

The exact definitions of the risk domain transformation and its inverse are given in Equations 1 and 2 where x_t is a CGM reading at time t and y_t is its corresponding risk value.

2 METHOD

For dividing the BGL range into classes, the following exact procedure is used. The risk domain range $[-2, 2]$ is considered and 100 bin midpoints are placed on it. The bin midpoints are denoted $y_1^*, y_3^*, y_3^* \ldots y_{100}^*$ with $y_1^* = -2$ and $y_{100}^* = 2$. The remaining bin midpoints are equally spaced along the risk range between y_1^* and y_{100}^*. This ensures that for the smaller hypoglycemic range, the bin sizes will be scaled properly in size and that there will be a proportionate number of bins (and therefore classes) in each subrange of the BGL scale.

Next, in order to a assign a new CGM reading x (in mg/dl) to a bin, its corresponding risk value $y = r(x)$ is computed using Equation 1. The reading is then assigned the bin with the closest midpoint. This means that the split points between bins do not need to be calculated explicitly, and if a reading is outside the risk range (either $x < -2$ or $x > 2$) then it will be assigned to one of the bins at the ends of range. However, this tends not happen often since the significant majority of readings in the competition datasets lie on the $[-2, 2]$ range.

Finally, once its class is determined, the reading x is transformed into a one-hot encoded vector $(0, 0 \ldots 0, 1, 0 \ldots 0, 0)$ of length 100 where the single 1 in the vector corresponds to the bin that x is assigned to.

To summarise the process, the predictors X for the model comprise a time series of risk-transformed CGM readings, and the target Y is a one-hot encoded class vector of dimension 100. Predictions are therefore numeric vectors, e.g. $(0, 0 \ldots 0.25, 0.6, 0.3 \ldots 0, 0)$. Note that whichever type of model is used, the values should be positive and sum to unity so that they can be interpreted as normal probabilities.

To evaluate this idea, 100-class classification experiments using a neural network as a predictive model were performed. Figure 2 depicts the particular neural network architecture used here.

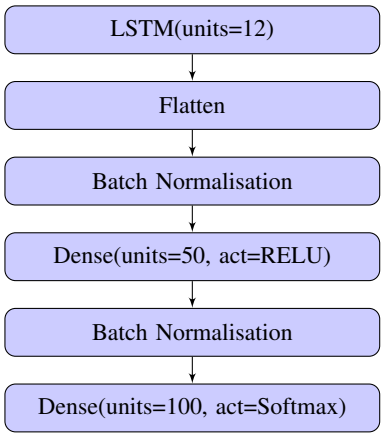


Figure 2. Architecture of the neural network used in the experiments.

The inputs to the neural network are twelve risk value readings, representing the past sixty minutes of BGL variation as sensed by the CGM (assuming that readings occur every five minutes). Since this is a time series, a long-short term memory (LTSM) layer with twelve units is used for initial input processing. Next, the LSTM output is flattened and passed through two dense fully connected layers for further processing. Both of these layers apply batch normalisation first, which ensures faster convergence times and stability during training. Finally, the last dense layer predicts the class and applies a softmax activation function to ensure that the output is a probability vector.

To train each instance (one per patient) of this neural network, the ADAM optimiser [7] was used with a learning rate of 0.0001, 10k epochs, batch size of 32, and validation data set to a random 15% subset of training data. The loss function utilised was categorical cross-entropy, which is commonly used for multiclass classification problems. Early stopping during training was permitted if no improvement in loss was observed for 100 epochs. All other settings were identical to those used in `keras` v. 2.3.1 [4] with `tensorflow` v. 2.1.0 [2] as a base neural network system. The neural network parameters and architecture decisions were made as a result of single-run experiments using data from the first patient in the 2018 competition dataset [10].

To generate data for training the neural network, a strict approach was taken towards missing data and all examples with gaps or time discrepancies (e.g. readings not exactly five minutes apart) exist were excluded. Therefore, to generate one example for the thirty minute $(t+30)$ forecasting problem, it is required that nine consecutive readings (from the start of the example up to and including the prediction target) exist, and for the sixty minute problem $(t + 60)$, twelve consecutive readings were required. No missing value imputation was performed. As a result, the number of test examples varies slightly depending on the forecasting horizon: i.e. the number of 2020 dataset test examples is 2,743, 2,579, 2,177, 2,185, 2,393 and 2,624 for the

30-minute horizon and 2,689, 2,531, 2,111, 2,113, 2,297 and 2,582 for the 60 minute horizon respectively.

With the neural network architecture and training data construction approach described, the final aspect of methodology to be described is the way that numeric point predictions were generated for the competition purposes (which require point predictions). A simplistic approach is allow only bin midpoints (i.e. $y_1^* \dots y_{100}^*$) to be predictions, and select the bin/class with the highest probability. Initial tests showed that this technique had low accuracy. Instead, a more sophisticated approach is to calculate the expected BGL value, as described by the following equation:

$$\begin{aligned} \hat{x}_{t+n} &= f(m_n, y_{t-55}, y_{t-50} \dots y_{t-5}, y_t) \\ &= \sum_{i=1}^{100} p(y_i^* | m_n, y_{t-55} \dots y_t) r^{-1}(y_i^*) \end{aligned} \tag{3}$$

where $n \in \{30, 60\}$, m_n is the neural network for the current patient with forecasting horizon n, y_t is the risk-transformed BGL level at time t ($y_t = r(x_t)$), $y_1^* \dots y_{100}^*$ are risk bin midpoints, $f()$ represents the application of m_n to the observed CGM values, and $\hat{x}_{t+n}$ is the expected value or prediction at time $t + n$. Since the probabilities across the 100 bins sum to 1, the resulting point prediction will be scaled correctly. Initial experiments showed that this expected value approach produced accurate estimates. Source code is available [1].

3 RESULTS

Two rounds of experiments were performed. In the first round, neural networks models were independently trained and tested for each of the twelve patients from both the 2018 and 2020 competition datasets. There was no sharing of information between models. In the second round, models for the 2020 patients only were trained, and the training data for each patient included all of the training data from the 2018 competition in addition to the specific 2020 patient's training data. An individual patient's own training data was therefore a small subset of his/her full training dataset. To account for this imbalance in the second round, samples from the target patient were re-weighted by a factor of six compared to the sample weights from the other patients.

Results for the first round of experiments are given in Tables 1 and 2. The first table is mean absolute error (MAE) results, and the second table gives root mean squared error (RMSE) results. Units are mg/dl. Using both metrics, predictive performance is comparable between the two datasets, with patient 540 from the 2020 dataset being the most "difficult" patient to predict. Conversely, the patient with lowest forecast error is patient 570 from the 2018 dataset.

Results for the second round of experiments are given in Tables 3 and 4. It can be observed that the additional training data leads to a very slight improvement in accuracy. Performing a paired t-test across the six 2020 patients reveals that the improvement in MAE is significant even albeit small (average improvement for thirty minute forecasting is 0.45 with significance $p = 0.000056$, and for sixty minute forecasting it is 0.82 with significance $p = 0.008096$).

More interesting are the prediction plots that can be generated when making a forecast using the classification approach. Figures 3-6 depict some probability densities produced by the model when making four different predictions. For each figure, the point prediction generated using the expected value computation (Equation 3) is shown as red line.

The tidiest example is Figure 3, which depicts a single-peaked distribution with the forecast coinciding with the peak of the distribu-

Patient ID	MAE $t+30$	std	MAE $t+60$	std
559	14.7	0.2	26.2	0.1
570	12.2	0.4	21.0	0.4
588	14.0	0.1	23.4	0.2
563	13.6	0.1	22.5	0.2
575	15.0	0.1	25.9	0.2
591	16.0	0.1	26.3	0.1
Avg.	14.3		24.2	
540	16.8	0.1	30.8	0.1
544	12.9	0.2	23.3	0.2
552	12.5	0.1	23.0	0.1
567	14.9	0.1	27.9	0.2
584	16.7	0.1	28.0	0.2
596	12.6	0.1	21.7	0.0
Avg.	14.4		25.8	

Table 1. Mean absolute error (MAE, mg/dl) results by patient and prediction horizon, averaged over five runs.

Patient ID	RMSE $t+30$	std	RMSE $t+60$	std
559	21.6	0.2	35.5	0.2
570	17.2	0.5	28.3	0.3
588	19.1	0.1	31.8	0.2
563	20.6	0.3	31.2	0.4
575	23.8	0.4	36.4	0.3
591	21.8	0.2	33.7	0.1
Avg.	20.7		32.8	
540	23.0	0.2	40.6	0.14
544	17.4	0.2	30.5	0.17
552	16.9	0.0	30.2	0.10
567	20.9	0.2	36.9	0.22
584	23.0	0.1	36.6	0.16
596	17.8	0.1	29.5	0.03
Avg.	19.8		34.0	

Table 2. Root mean squared error (RMSE, mg/dl) results by patient and prediction horizon, averaged over five runs.

Patient ID	MAE $t+30$	std	MAE $t+60$	std
540	16.4	0.1	29.8	0.1
544	12.3	0.1	22.9	0.2
552	12.2	0.1	22.2	0.2
567	14.5	0.2	27.4	0.4
584	16.1	0.2	26.4	0.1
596	12.2	0.3	21.3	0.1
Avg.	13.9		25.0	

Table 3. Mean absolute error (MAE, mg/dl) results by patient and prediction horizon, averaged over five runs for the 2020 dataset only. Training data includes the entire 2018 dataset.

Patient ID	RMSE $t+30$	std	RMSE $t+60$	std
540	22.4	0.1	39.5	0.1
544	17.0	0.1	30.1	0.2
552	16.5	0.1	29.3	0.1
567	20.8	0.2	36.9	0.4
584	22.4	0.2	35.9	0.2
596	17.2	0.3	29.0	0.2
Avg.	19.4		33.4	

Table 4. Root mean squared error (RMSE, mg/dl) results by patient and prediction horizon, averaged over five runs for the 2020 dataset only. Training data includes the entire 2018 dataset.

tion. While this class of forecast is common, it is not the only type of distribution that is output from the model.

Figure 4 shows a two-peaked distribution with an expected value between the peaks. While the expected value is a useful point prediction, the dual peaks are also useful information since it can be clearly observed that the two most likely outcomes are normoglycemia vs. a state most likely in level I hyperglycemica, although the model is not certain.

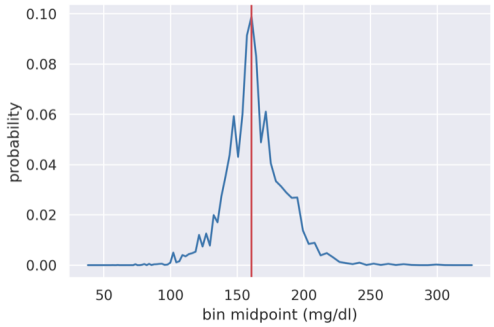


Figure 3. Probability distribution over BGLs for one prediction. In this example, the expected BGL coincides with the distribution peak.

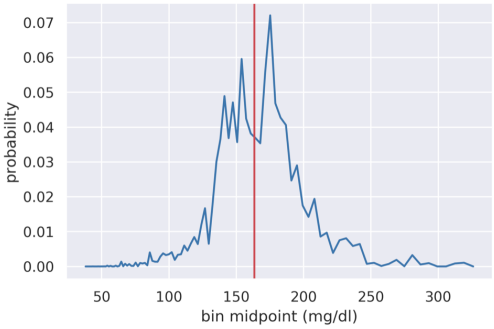


Figure 4. In this probability distribution over BGL levels, two peaks exist in the distribution and the expected BGL lies between the peaks.

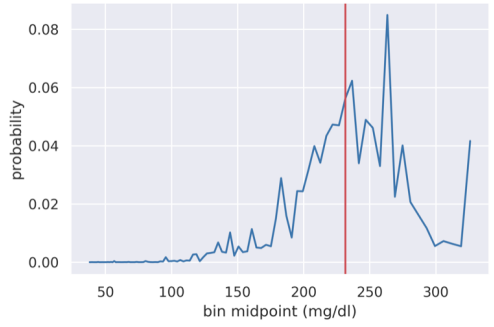


Figure 5. In this example, the distribution is skewed to to the right with the expected BGL quite a distance to the left of the peak.

Figure 5 shows a skewed distribution a significant mass of the probability distribution is at the upper end of the range, but the expected value is closer to the middle of the range (albeit still in the hyperglycemic range). In this case, the expected value underestimates

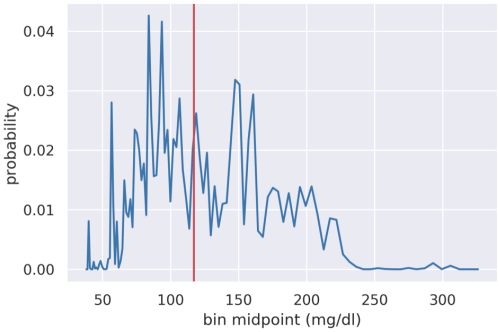


Figure 6. An example of a prediction with considerable uncertainty over the hypoglycemic and normoglycemic ranges.

the true risk to the patient at the current time. Again, this is clearly noticeable.

Finally, Figure 6 depicts a prediction with considerable noise in the distribution. While the forecast is normoglycemic, there is significant mass in the hypoglycemic range. Therefore it could be concluded that although the point prediction is reasonably, there is still significant risk of hypoglycemia.

Further analysis was performed with respect to the variance of the prediction distributions. It was found that for most patients, the distribution of variances is skewed to the left indicating that on average most predictions are more certain (more like Figures 3 and 4) than uncertain. However, more analysis needs to be done on this point.

4 CONCLUSION

This paper describes a system for forecasting BGL at thirty and sixty minutes in advance. This main distinctiveness of this approach is the adoption of a highly multi-class classification-based technique and use of a domain-specific transform for normalising BGL values (opposed to more traditional min/max scaling or standardisation). The ability to visualise non-parametric probability distributions accompanying predictions as a meaningful context is a clear advantage.

To test the proposed method with real patients, we will use a system known as SmartCGMS [8]. SmartCGMS is a continuous glucose monitoring and controlling software framework. It provides infrastructure to connect and develop "building blocks" for an insulin-pump software stack. Principally, the pump developer connects CGM-sensor blocks to computing blocks, which predict BGL and subsequently schedule insulin boluses or adjust the insulin basal rate. Next, another block transforms the results of these computations into insulin-pump control commands. With SmartCGMS, we can close the loop *in-silico*[15] first, before conducting an *in-vivo* experiment to ensure maximum safety.

Our specific approach will be to transform the best trained keras/tensorflow-based neural network into a hard-coded and constant feed-forward neural network in C++. This will enable efficient deployment and computation on low-power devices such as insulin-pump controllers, while we can still train the original neural network using high-performance computers. As a result, a flow in which a neural network is continuously learned from patient BGL measurements, providing personalised BGL predictions, can be established.

Acknowledgment

This publication was partially supported by the project LO1506 of the Czech Ministry of Education, Youth and Sports.

REFERENCES

[1] https://www.cms.waikato.ac.nz/%7Emmayo/kdhcompsource.py

[2] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[3] Marling C. and Bunescu R. The OhioT1DM dataset for blood glucose level prediction: Update 2020. http://smarthealth.cs.ohio.edu/bglp/OhioT1DM-dataset-paper.pdf, 2020.

[4] François Chollet et al. Keras. https://keras.io, 2015.

[5] Xia Dai, Zu-Chun Luo, Lu Zhai, Wen-Piao Zhao, and Feng Huang, 'Artificial pancreas as an effective and safe alternative in patients with type 1 diabetes mellitus: A systematic review and meta-analysis', *Diabetes therapy : research, treatment and education of diabetes and related disorders*, **9**(3), 1269–1277, (06 2018).

[6] Thomas Danne, Revital Nimri, Tadej Battelino, Richard M. Bergenstal, Kelly L. Close, J. Hans DeVries, Satish Garg, Lutz Heinemann, Irl Hirsch, Stephanie A. Amiel, Roy Beck, Emanuele Bosi, Bruce Buckingham, Claudio Cobelli, Eyal Dassau, Francis J. Doyle, Simon Heller, Roman Hovorka, Weiping Jia, Tim Jones, Olga Kordonouri, Boris Kovatchev, Aaron Kowalski, Lori Laffel, David Maahs, Helen R. Murphy, Kirsten Nørgaard, Christopher G. Parkin, Eric Renard, Banshi Saboo, Mauro Scharf, William V. Tamborlane, Stuart A. Weinzimer, and Moshe Phillip, 'International consensus on use of continuous glucose monitoring', *Diabetes Care*, **40**(12), 1631–1640, (2017).

[7] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014.

[8] Tomas Koutny and Martin Ubl, 'Parallel software architecture for the next generation of glucose monitoring', *Procedia Computer Science*, **141**, 279–286, (2018).

[9] Boris P. Kovatchev, Martin Straume, Daniel J. Cox, and Leon S. Farhy, 'Risk analysis of blood glucose data: A quantitative approach to optimizing the control of insulin dependent diabetes', *Journal of Theoretical Medicine*, **3**(1), (2000).

[10] Cindy Marling and Razvan Bunescu, 'The OhioT1DM dataset for blood glucose level prediction', in *Proceedings of the 3rd International Workshop on Knowledge Discovery in Healthcare Data*, 60–63, CEUR Workshop Proceedings (CEUR-WS.org), (2018).

[11] John Martinsson, Alexander Schliep, Bjorn Eliasson, Christian Meijner, Simon Persson, and Olof Mogren, 'Automatic blood glucose prediction with confidence using recurrent neural networks', in *Proceedings of the 3rd International Workshop on Knowledge Discovery in Healthcare Data*, 64–68, CEUR Workshop Proceedings (CEUR-WS.org), (2018).

[12] Himel Mondal and Shaikat Mondal, 'Clarke error grid analysis on graph paper and microsoft excel', *Journal of Diabetes Science and Technology*, **14**(2), 499–499, (2020). PMID: 31777281.

[13] Leann Olansky and Laurence Kennedy, 'Finger-stick glucose monitoring', *Diabetes Care*, **33**(4), 948–949, (2010).

[14] Dawei Shi, Sunil Deshpande, Eyal Dassau, and Francis J. Doyle, 'Chapter 1 - feedback control algorithms for automated glucose management in t1dm: the state of the art', in *The Artificial Pancreas*, eds., Ricardo S. Sánchez-Peña and Daniel R. Cherñavvsky, 1 – 27, Academic Press, (2019).

[15] Martin Ubl and Tomas Koutny, 'SmartCGMS as an environment for an insulin-pump development with FDA-accepted in-silico pre-clinical trials', *Procedia Computer Science*, **160**, 322–329, (2019).

[16] Ashenafi Zebene Woldaregay, Eirik Årsand, Ståle Walderhaug, David Albers, Lena Mamykina, Taxiarchis Botsis, and Gunnar Hartvigsen, 'Data-driven modeling and prediction of blood glucose dynamics: Machine learning applications in type 1 diabetes', *Artificial Intelligence in Medicine*, **98**, 109 – 134, (2019).

Investigating potentials and pitfalls of knowledge distillation across datasets for blood glucose forecasting

Hadia Hameed, Samantha Kleinberg[1]

Abstract. Individuals with Type I diabetes (T1D) must frequently monitor their blood glucose (BG) and deliver insulin to regulate it. New devices like continuous glucose monitors (CGMs) and insulin pumps have helped reduce this burden by facilitating closed-loop technologies like the artifical pancreas (AP) for delivering insulin automatically. As more people use AP systems, which rely on a CGM and insulin pump, there has been a dramatic increase in the availability of large scale patient-generated health data (PGHD) in T1D. This data can potentially be used to train robust, generalizable models for accurate BG forecasting which can then be used to make forecasts for smaller datasets like OhioT1DM in real-time. In this work, we investigate the potential and pitfalls of using knowledge distillation to transfer knowledge from a model learned from one dataset to another and compare it with the baseline case of using either dataset alone. We show that using a pre-trained model to do BG forecasting for OhioT1DM from CGM data only (univariate setting) has comparable performance to training on OhioT1DM itself. Using a single-step, univariate recurrent neural network (RNN) trained on OhioT1DM data alone, we achieve an overall RMSE of 19.21 and 31.77 mg/dl for a prediction horizon (PH) of 30 and 60 minutes respectively.

1 Introduction

Type 1 diabetes (T1D) is a chronic lifelong disease that requires dozens of daily decisions to manage blood glucose (BG). While keeping BG in a healthy range is critical for avoiding complications, it is challenging, as meals and many other factors like exercise and stress can affect BG and insulin sensitivity. Closed-loop technologies, which connect a continuous glucose monitor (CGM) and insulin pump with a control algorithm, could relieve this burden by automatically dosing insulin. This requires an accurate forecast of where glucose is headed so the right amount of insulin can be delivered to keep BG within a target range dynamically.

Prior works include using system identification techniques to model glucose-insulin interactions [18, 3] , using classic autoregressive models for time series forecasting [23, 1, 5, 6] or training deep neural networks to implicitly learn the changing glucose level patterns [16, 17, 4, 24]. Neural network architectures such as LSTM have been used successfully for many time series forecasting problems [10, 8, 7, 19, 15], but require large amounts of training data. This is a challenge for BG forecasting, as it is time consuming and can be infeasible to collect such massive datasets. However, there are now large public datasets created by people with diabetes sharing their own data, which we believe could be leveraged. In particular, the open source artificial pancreas system (OAPS) [11], a collaborative project led by people with T1D, has data donated by individuals using the system. To date, there is open source diabetes data available for more than 100 subjects, collected over a period of 1 – 4 years (more than 1000 days worth of data for some individuals). This patient generated data is self-reported, noisy, heterogeneous, and irregularly sampled, but its much larger than the datasets routinely collected in controlled studies.

We propose that large public datasets like OAPS can be used to pretrain models, allowing deep learning to be used on smaller curated datasets for forecasing BG. In particular, we show by augmenting and distilling knowledge across models trained on data obtained from different sources using RNN, we achieve an accuracy comparable to that achieved by using OhioT1DM dataset alone for univariate setting. We also compare the performance with multi-output setting in which multiple BG values are estimated in the prediction horizon simultaneously. The code is available at https://github.com/health-ai-lab/BGLP_BG_forcasting.

2 Methodology

The task here is to forecast future values for BG. We compare single-step and multi-output forecasting. In the single-step setting, a single glucose value is estimated several minutes into the future, whereas in multi-output forecasting several future values are estimated simultaneously to model the signal trajectory over the prediction horizon. We begin by describing our time series forecasting approach, and later discuss the dataset specific preprocessing.

2.1 Problem setup

We define the feature vector $X_{0:t} = \{x_0, x_1, ..., x_t\} \in \mathbb{R}^n$ with n being the number of variables. We use only raw CGM values and do not incorporate additional features like carbohydrate intake and insulin dosage. We also have a corresponding output time series $X'_{t+1:t+h} = \{x'_{t+1}, x'_{t+2}, \ldots, x'_{t+h}\} \in \mathbb{R}$ representing multiple future glucose values across a given prediction horizon (PH) of 30 and 60 minutes. As CGM data is recorded at a frequency of 5 minutes, a PH of 30 and 60 minutes will lead to $h = 6$ and $h = 12$ samples, respectively. For the single step setting, this target vector becomes $X'_{t+h} = \{x'_{t+h}\}$ estimating only a single value h time instances in the future. Multi-output forecasting, on the other hand, aims to estimate the joint probability $p(X'_{t+1:t+h}|X_{0:t})$ simultaneously. However, root mean square error (RMSE) was calculated by comparing the actual future glucose level and the last future value in the estimated multi-output sequence, to accurately measure the performance of the forecasting model across the two output settings.

[1] Stevens Institute of Technology, USA, email: hhameed@stevens.edu

2.2 Learning Framework

Our proposed approach is to make glucose estimations for a small dataset by pre-training an RNN on a larger dataset and then re-training it using a smaller dataset. We compare four learning approaches for glucose forecasting, as shown in Fig.1: I) training and testing an RNN on OhioT1DM only (red path), II) training an RNN on OAPS dataset and testing on OhioT1DM without any re-training (blue path), III) training an RNN on OAPS dataset, training again OhioT1DM, and then testing on the OhioT1DM (purple path), and IV) the pre-trained RNN model makes intermediate estimates called soft predictions, which are given as target estimates to a student artificial neural network (ANN) model instead of the actual ground truth, as done for a classification task in [2]. As shown in the figure, the black edges from the two datasets to the teacher model show that it is pre-trained using the source data (OAPS here) but uses target data for making final predictions in Approach II, for re-training in Approach III, and making soft estimations in Approach IV (mimic learning), thus always having access to the two datasets.

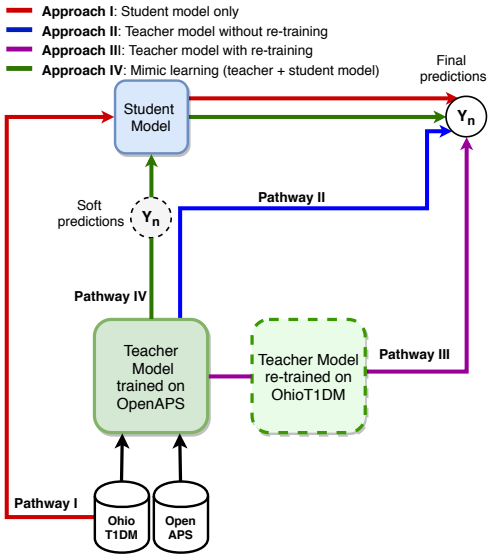


Figure 1: Four learning pipelines to estimate blood glucose levels in OhioT1DM test data.

2.3 Network Architecture

We use a vanilla RNN with a single hidden layer, $H(t)$ with 32 units, followed by a fully-connected output layer $O(t)$. This was used as the teacher model in Approaches II, III and IV and trained on the source patient-generated data. In Approach I where no teacher model was involved, the RNN was used as a student model trained only on the target OhioT1DM dataset to observe the effects of using datasets of different sizes and from different sources with the same network architecture. In Approach IV, a teacher RNN model trained on OAPS was used to teach a student ANN model using OhioT1DM data to study the effects of knowledge distillation between different kinds of networks. We use a simple, fully-connected ANN with a single hidden layer with 32 units. The number of units for both RNN and ANN were chosen after trying and testing [28, 32, 64, 128] and optimizing for the least RMSE. The output layer $O(t)$ predicts the glucose value(s) 30 or 60 minutes into the future depending on the PH and the output setting (single-step or multi-output).

2.4 Training models

The teacher model was trained on OAPS dataset which was pre-processed the same way as OhioT1DM, as discussed in Section 4. Early stopping was used to halt the training process if validation loss was not improving significantly, with the maximum number of epochs being 1000 with a batch size of 248 and 128 for OAPS and OhioT1DM, proportional to size of each dataset. Glorot normal initialization [9] was used to initialize the weight matrix. For the OhioT1DM dataset, the same training configurations (maximum epochs, batch size, initialization technique etc.) were used with all the learning approaches (i.e. student, teacher, retrained teacher, teacher-student) for a fair comparison. The experiments for Approach I, III and IV were repeated 10 times and the average RMSE and MAE was recorded for each subject, along with the standard deviation as presented in the Section 5.

3 Data

We aim to evaluate the impact of using a large noisy dataset for improving forecasting in a smaller more controlled dataset. The larger (source) dataset from OAPS [20] was used to pre-train the model before it was trained on OhioT1DM [12] (target), which is much smaller in terms of the total number of subjects and days for each.

3.1 OAPS

The collection of OAPS data started in 2015 as part of an initiative to make APS technology more accessible and transparent for people with T1D and to enable them to create their own customized AP systems. Participants can voluntarily donate their data, including glucose levels recorded via CGM, insulin basal and bolus rates, carbs intake, physical activity, and other physiological data. Researchers can gain access to this dataset free of charge, provided they share their insights and research findings with the public within a reasonable frame of time [21]. For this work, we used a subset of the dataset from individuals with multiple calendar years of data (55 people total, 320 ± 158.3 days of data on average). Since this data is largely self-reported, it is noisy, irregularly sampled, and heterogeneous in terms of the variables recorded, but because of its sheer size, it is highly useful for pre-training a robust machine learning model for accurate BG forecasting.

3.2 OhioT1DM

The training data consists of 12 subjects: six from the OhioT1DM dataset shared in 2018 for the First BGLP Challenge (Group I)[13], and six from the second BGLP Challenge 2020 (Group II)[12]. The validation and test samples are drawn from the last 10 days of data for subjects in Group I and Group II, respectively. The dataset contains around 8 weeks of data for 20 variables including raw CGM values, insulin basal and boluses, carbohydrate intake, exercise, and sleep.

4 Data pre-processing

For both OAPS and OhioT1DM, we use four recorded variables and one attribute derived from the raw glucose values. The list of features

used in the experiments includes raw CGM values (*glucose_level*, insulin basal rate (*basal* and *temp_basal*), bolus amount (*bolus*), carbs intake (*meal*), and difference between consecutive glucose values calculated during data pre-processing (*glucose_diff*). The first step in data pre-processing was to synchronize the multi-modality data by generating a single timestamp data field based on the timestamps for each of the four fields, generating an irregularly sampled multivariate time series.

In OAPS dataset, there were two types of gaps present in the data, first where both timestamp and glucose values were missing, and second where the timestamp was recorded but the corresponding glucose value was missing. In OhioT1DM, missing glucose values were identified once the multi-modality data was synchronized since basal, bolus, and meals are not recorded at the same 5-minute frequency as glucose levels. When there was missing glucose data for more than 25 consecutive minutes, these times were not used during training. Each data segment (series of points not separated by gap longer than 25 minutes) was then imputed and windowed separately to maintain temporal continuity in the data.

For the rest of the data, which may contain shorter gaps, we used linear interpolation to impute missing glucose values in training data. Missing values in test data were imputed by extrapolation to avoid using data from the future. Basal rates were imputed with forward filling, meaning replacing missing values with the last recorded basal rate, since the value is only recorded when it changes and thus missing values mean the last recorded one is still active. However, if the field "*temp_basal*", recording temporary basal infusion rate, was present for a given set of timestamps, it was used to replace the recorded basal rate [12] by evenly distributing the rate across the time duration which was divided into 5-minute intervals, as implemented in [14, 22]. Bolus rates were imputed in a similar manner by calculating the rate for every 5-minute interval and distributing it evenly across the specified duration, and was set to 0 when it was not recorded, thereby indicating that insulin was not bolused for those time instances. Similarly, the data field "*meal*" which recorded the amount of carbohydrate intake was set to 0 when it was missing.

In addition to missing data, the sensors are also noisy, leading to sudden changes in glucose levels, which can cause high variance in the learned model. To remove these spikes, the signal was passed through a median filter with a window size of 5 samples, as in [25]. This was only done for training data and not for the validation and test sets to test robustness of the model.

A sliding window was used to split the data into fixed sized sequences for further downstream analysis. There are three parameters for the moving window configuration: history window size (number of past samples to use for forecasting), prediction horizon (PH) and output window (how far into the future and how many future values to predict), and stride (number of samples to skip while sliding the window). An hour (12 samples) of past values were used to predict the glucose levels 30 and 60 minutes into the future (PH = 30, 60) with a unit stride, which means overlapping windows were used to partition the data.

In OhioT1DM train and test data, the raw CGM values range from 70 – 275 mg/dl and 75 – 290 mg/dl on average, respectively. To ensure that values of all the features were in the same range, insulin basal, bolus rates and carbs intake were normalized based on the minimum and maximum value of glucose levels using Min-Max Normalization.

5 Experiments

5.1 Experimental set up

The last ten days of data for subjects with ID 559, 563, 570, 575, 588 and 591 were used as validation set and test set was sampled from data for subjects 540, 544, 552, 567, 584, 596. The processing steps for the test data included linear extrapolation for imputing missing values and normalization. The test data was not passed through a median filter like the training set to see how robust the trained models were to unseen, noisy data. We use root mean square error (RMSE) and mean absolute error (MAE) to compare the predicted values with the actual ground truth to evaluate the model. MAE and RMSE can be expressed as,

$$MAE = \frac{1}{n}\sum_{n=1}^{n}|y_i - \hat{y_i}| \quad (1)$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{n=1}^{n}(y_i - \hat{y_i})^2} \quad (2)$$

where y_i is true glucose level and $\hat{y_i}$ is estimated glucose level, both measured in mg/dl. We repeated the experiments 10 times and calculated the average RMSE and MAE for each subject across the ten trials. We also report the best, worst and mean RMSE (MAE) across all the subjects for each of the four pipelines using both single-step and multi-output models.

5.2 Results

Table 1: RMSE (MAE) for single-step forecasting with different learning pipelines for a PH of 30 minutes.

(a) Single-step

Subject ID	I	II	III	IV
540	19.55 (14.00)	20.32 (14.60)	20.36 (14.69)	20.46 (14.81)
544	16.56 (11.51)	17.84 (12.51)	17.50 (12.20)	17.92 (12.53)
552	15.04 (11.14)	16.17 (11.90)	15.72 (11.63)	16.20 (12.06)
567	23.07 (14.67)	24.09 (15.38)	23.91 (15.32)	24.74 (15.65)
584	25.19 (16.16)	26.47 (16.88)	26.97 (16.65)	26.83 (16.84)
596	15.85 (10.98)	17.24 (12.06)	16.50 (11.52)	17.50 (12.12)
Best	15.04 (11.14)	16.17 (11.90)	15.72 (11.63)	16.20 (12.06)
Worst	25.19 (16.16)	26.47 (16.88)	26.97 (16.65)	26.83 (16.84)
Average	**19.21 (13.07)**	20.36 (13.89)	20.16 (13.67)	20.61 (14.00)

(b) Multi-output

Subject ID	I	II	III	IV
540	20.30 (14.64)	20.41 (14.77)	20.36 (14.68)	20.55 (15.18)
544	17.61 (12.19)	18.07 (12.59)	17.68 (12.23)	18.41 (12.91)
552	15.68 (11.57)	15.98 (11.74)	15.66 (11.54)	16.06 (12.08)
567	23.94 (15.29)	24.88 (15.58)	23.66 (15.08)	24.47 (15.55)
584	26.61 (16.65)	26.29 (16.71)	25.82 (16.43)	26.70 (17.01)
596	16.46 (11.43)	17.17 (11.86)	16.54 (16.54)	17.57 (12.21)
Best	15.68 (11.57)	15.98 (11.74)	15.66 (11.54)	17.57 (12.21)
Worst	26.61 (16.65)	26.29 (16.71)	25.82 (16.43)	26.70 (17.01)
Average	20.10 (13.63)	20.46 (13.87)	19.95 (13.57)	20.63 (14.16)

I: Student model only, II: Teacher model without re-training,
III: Teacher model with re-training, IV: Mimic learning (teacher + student model)

The results for a PH of 30 and 60 minutes are shown in Tables 1 and 2, respectively.

Overall, approach I achieved the lowest RMSE (MAE) with 19.21 (13.07) for a PH of 30 minutes and 31.77 (23.09) for PH = 60 minutes. In this approach an RNN was trained only using the OhioT1DM data, using raw CGM values. The worst performance was from approach IV, where estimations made by a teacher model pre-trained on OpenAPS dataset were given as ground truth to student ANN model for training on OhioT1DM, as shown in Tables 1a and 2a. This approach did not improve the forecast accuracy as it did in [2]. It might be because [2] used this technique for a classification task of mortality prediction which involved predicting hard labels and evaluated performance using misclassification error instead of estimating continuous valued deviations from the ground truth as is the case in BG forecasting.

For BG forecasting using multi-output model, all approaches performed equally well, with approach I, II, and IV (student model, teacher and teacher_student model) giving the same RMSE on average. For approach II, the error did not worsen significantly, showing that pre-trained models can be used for making forecasts for OhioT1DM data in real-time, without having to set aside a portion of the dataset for retraining the model, an important consideration for smaller datasets. However, the RMSE improved slightly for approach II when the teacher model was retrained.

Approach IV

Table 2: RMSE (MAE) for single-step forecasting with different learning pipelines for a PH of 60 minutes.

(a) Single-step

Subject ID	I	II	III	IV
540	33.94 (25.40)	35.54 (26.74)	35.16 (26.94)	35.84 (27.35)
544	27.79 (20.34)	31.07 (22.45)	30.79 (22.84)	31.23 (22.69)
552	26.68 (20.15)	28.36 (21.08)	27.99 (21.33)	28.54 (21.53)
567	37.99 (26.50)	39.89 (27.76)	40.63 (28.47)	39.57 (28.09)
584	37.47 (27.00)	39.74 (27.87)	39.40 (27.60)	39.36 (27.85)
596	26.72 (19.12)	28.41 (20.42)	27.89 (20.31)	28.75 (20.83)
Best	26.68 (20.15)	28.36 (21.08)	27.89 (20.31)	28.54 (21.53)
Worst	37.99 (26.50)	39.89 (27.76)	40.63 (28.47)	39.57 (28.09)
Average	**31.77 (23.09)**	33.84 (24.39)	33.64 (24.58)	33.88 (24.72)

(b) Multi-output

Subject ID	I	II	III	IV
540	35.23 (26.94)	35.32 (26.99)	35.33 (27.06)	35.66 (27.05)
544	30.68 (22.83)	31.17 (22.74)	30.73 (22.79)	31.23 (22.84)
552	28.22 (21.57)	28.57 (21.56)	28.13 (21.43)	29.23 (21.98)
567	39.53 (28.03)	39.07 (27.95)	39.32 (21.43)	41.33 (28.57)
584	39.60(27.71)	39.43 (27.91)	39.43 (21.43)	39.07 (27.37)
596	27.92 (20.27)	28.48 (20.55)	28.13 (20.42)	28.81 (20.81)
Best	27.92 (20.27)	28.48 (20.55)	28.13 (20.42)	28.81 (20.81)
Worst	39.60 (27.71)	39.43 (27.91)	39.43 (21.43)	41.33 (28.57)
Average	33.53 (24.56)	33.67 (24.62)	33.516 (24.52)	34.22 (24.77)

I: Student model only, II: Teacher model without re-training,

III: Teacher model with re-training, IV: Mimic learning (teacher + student model)

6 Conclusion

In this work we have compared four different learning strategies for BG forecasting using two different datasets. We have shown that an RNN model pre-trained on a bigger dataset such as OpenAPS can be used directly to do BG forecasting for a smaller dataset like OhioT1DM when using CGM data only. We predicted BG levels 30 and 60 minutes into the future using single-step and multi-output models, using univariate BG data. Overall, a single-step RNN trained only on univariate data from OhioT1DM dataset achieved the least RMSE of 19.21 and 31.77 mg/dl for a PH of 30 and 60 minutes, respectively.

ACKNOWLEDGEMENTS

We would like to thank the referees for their comments, which helped improve this paper considerably. This work was supported in part by the NSF under award number 1915182, NIH under award number R01LM011826, and Fulbright Scholarship.

REFERENCES

[1] Ransford Henry Botwey, Elena Daskalaki, Peter Diem, and Stavroula G Mougiakakou, 'Multi-model data fusion to improve an early warning system for hypo-/hyperglycemic events', in *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 4843–4846. IEEE, (2014).

[2] Zhengping Che, Sanjay Purushotham, Robinder Khemani, and Yan Liu, 'Interpretable deep models for icu outcome prediction', in *AMIA Annual Symposium Proceedings*, volume 2016, p. 371. American Medical Informatics Association, (2016).

[3] C Cobelli, G Nucci, and S Del Prato, 'A physiological simulation model of the glucose-insulin system', in *Proceedings of the First Joint BMES/EMBS Conference. 1999 IEEE Engineering in Medicine and Biology 21st Annual Conference and the 1999 Annual Fall Meeting of the Biomedical Engineering Society (Cat. N*, volume 2, pp. 999–vol. IEEE, (1999).

[4] J Fernandez de Canete, S Gonzalez-Perez, and JC Ramos-Diaz, 'Artificial neural networks for closed loop control of in silico and ad hoc type 1 diabetes', *Computer methods and programs in biomedicine*, **106**(1), 55–66, (2012).

[5] Meriyan Eren-Oruklu, Ali Cinar, Lauretta Quinn, and Donald Smith, 'Estimation of future glucose concentrations with subject-specific recursive linear models', *Diabetes technology & therapeutics*, **11**(4), 243–253, (2009).

[6] Adiwinata Gani, Andrei V Gribok, Yinghui Lu, W Kenneth Ward, Robert A Vigersky, and Jaques Reifman, 'Universal glucose models for predicting subcutaneous glucose concentration in humans', *IEEE Transactions on Information Technology in Biomedicine*, **14**(1), 157–165, (2009).

[7] André Gensler, Janosch Henze, Bernhard Sick, and Nils Raabe, 'Deep learning for solar power forecasting—an approach using autoencoder and lstm neural networks', in *2016 IEEE international conference on systems, man, and cybernetics (SMC)*, pp. 002858–002865. IEEE, (2016).

[8] Felix A Gers, Douglas Eck, and Jürgen Schmidhuber, 'Applying lstm to time series predictable through time-window approaches', in *Neural Nets WIRN Vietri-01*, 193–200, Springer, (2002).

[9] Xavier Glorot and Yoshua Bengio, 'Understanding the difficulty of training deep feedforward neural networks', in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256, (2010).

[10] Nikolay Laptev, Jason Yosinski, Li Erran Li, and Slawek Smyl, 'Time-series extreme event forecasting with neural networks at uber', in *International Conference on Machine Learning*, volume 34, pp. 1–5, (2017).

[11] Dana Lewis, Scott Leibrand, and OpenAPS Community, 'Real-world use of open source artificial pancreas systems', *Journal of diabetes science and technology*, **10**(6), 1411, (2016).

[12] Cindy Marling and Razvan Bunescu, 'The ohiot1dm dataset for blood glucose level prediction: Update 2020', in *KHD@ IJCAI*, (2020).

[13] Cindy Marling and Razvan C Bunescu, 'The ohiot1dm dataset for blood glucose level prediction.', in *KHD@ IJCAI*, pp. 60–63, (2018).

[14] Cooper Midroni, Peter J Leimbigler, Gaurav Baruah, Maheedhar Kolla, Alfred J Whitehead, and Yan Fossat, 'Predicting glycemia in type 1 diabetes patients: experiments with xgboost', *heart*, **60**(90), 120, (2018).

[15] Sadegh Mirshekarian, Razvan Bunescu, Cindy Marling, and Frank Schwartz, 'Using lstms to learn physiological models of blood glucose behavior', in *2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 2887–2891. IEEE, (2017).

[16] Stavroula G Mougiakakou, Aikaterini Prountzou, Dimitra Iliopoulou, Konstantina S Nikita, Andriani Vazeou, and Christos S Bartsocas, 'Neural network based glucose-insulin metabolism models for children with type 1 diabetes', in *2006 International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 3545–3548. IEEE, (2006).

[17] Carmen Pérez-Gandía, A Facchinetti, G Sparacino, C Cobelli, EJ Gómez, M Rigla, Alberto de Leiva, and ME Hernando, 'Artificial neural network algorithm for online glucose prediction from continuous glucose monitoring', *Diabetes technology & therapeutics*, **12**(1), 81–88, (2010).

[18] Fredrik Ståhl and Rolf Johansson, 'Diabetes mellitus modeling and short-term prediction based on blood glucose measurements', *Mathematical biosciences*, **217**(2), 101–117, (2009).

[19] Qingnan Sun, Marko V Jankovic, Lia Bally, and Stavroula G Mougiakakou, 'Predicting blood glucose with an lstm and bi-lstm based deep neural network', in *2018 14th Symposium on Neural Networks and Applications (NEUREL)*, pp. 1–5. IEEE, (2018).

[20] Open Artificial Pancreas System. Openaps. `https://openaps.org/what-is-openaps/`, 2015. [Online; accessed 10-Dec-2019].

[21] Open Artificial Pancreas System. Openaps research application. `https://tinyurl.com/oaps-application`, 2015. [Online; accessed 10-Dec-2019].

[22] Jinyu Xie and Qian Wang, 'Benchmark machine learning approaches with classical time series approaches on the blood glucose level prediction challenge.', in *KHD@ IJCAI*, pp. 97–102, (2018).

[23] Jun Yang, Lei Li, Yimeng Shi, and Xiaolei Xie, 'An arima model with adaptive orders for predicting blood glucose concentrations and hypoglycemia', *IEEE journal of biomedical and health informatics*, **23**(3), 1251–1260, (2018).

[24] Konstantia Zarkogianni, Konstantinos Mitsis, Eleni Litsa, M-T Arredondo, G Fico, Alessio Fioravanti, and Konstantina S Nikita, 'Comparative assessment of glucose prediction models for patients with type 1 diabetes mellitus applying sensors for glucose and physical activity monitoring', *Medical & biological engineering & computing*, **53**(12), 1333–1343, (2015).

[25] Taiyu Zhu, Kezhi Li, Pau Herrero, Jianwei Chen, and Pantelis Georgiou, 'A deep learning algorithm for personalized blood glucose prediction.', in *KHD@ IJCAI*, pp. 64–78, (2018).

Blood Glucose Prediction for Type 1 Diabetes Using Generative Adversarial Networks

Taiyu Zhu[1], **Xi Yao**[2], **Kezhi Li**[3], **Pau Herrero**[4] and **Pantelis Georgiou**[5]

Abstract. Maintaining blood glucose in a target range is essential for people living with Type 1 diabetes in order to avoid excessive periods in hypoglycemia and hyperglycemia which can result in severe complications. Accurate blood glucose prediction can reduce this risk and enhance early interventions to improve diabetes management. However, due to the complex nature of glucose metabolism and the various lifestyle related factors which can disrupt this, diabetes management still remains challenging. In this work we propose a novel deep learning model to predict future BG levels based on the historical continuous glucose monitoring measurements, meal ingestion, and insulin delivery. We adopt a modified architecture of the generative adversarial network that comprises of a generator and a discriminator. The generator computes the BG predictions by a recurrent neural network with gated recurrent units, and the auxiliary discriminator employs a one-dimensional convolutional neural network to distinguish between the predictive and real BG values. Two modules are trained in an adversarial process with a combination of loss. The experiments were conducted using the OhioT1DM dataset that contains the data of six T1D contributors over 40 days. The proposed algorithm achieves an average root mean square error (RMSE) of 18.34 ± 0.17 mg/dL with a mean absolute error (MAE) of 13.37 ± 0.18 mg/dL for the 30-minute prediction horizon (PH) and an average RMSE of 32.31 ± 0.46 mg/dL with a MAE of 24.20 ± 0.42 for the 60-minute PH. The results are compared for clinical relevance using the Clarke error grid which confirms the promising performance of the proposed model.

1 INTRODUCTION

Diabetes is a chronic metabolic disorder that affects more than 400 million people worldwide with an increasing global prevalence [27]. Due to an absence of insulin production from the pancreatic β cells, people living with Type 1 diabetes (T1D) require long-term self-management through exogenous insulin delivery to maintain blood glucose (BG) levels in a normal range. In this regard, accurate glucose prediction has great potential to improve diabetes management, enabling proactive actions to reduce the occurrence of adverse glycemic events, including hypoglycemia and hyperglycemia.

In recent years, empowered by the advances in wearable devices and data-driven techniques, different BG prediction algorithms have been proposed and validated in clinical practice [29]. Among these, continuous glucose monitoring (CGM) is an essential technology that measures BG levels and provides readings in real-time. CGM has produced a vast amount of BG data with its increasing use in the diabetes population. Taking advantage of this, the emergence of deep learning algorithms for BG prediction has achieved recent success and outperformed several conventional machine learning approaches in terms of accuracy [1, 16, 17, 23, 28]. Generally, the major challenge of BG prediction lies in accounting for the intra- and inter-person variability that leads to various glucose responses under different conditions [25]. Furthermore, many external events and factors can influence glucose dynamics, such as meal ingestion, physical exercise, psychological stress, and illness. Deep learning is powerful at extracting hidden representations from large-scale raw data [15], making it suitable for accounting for the complexity of glucose dynamics in diabetes.

In this work, we propose a novel deep learning model for BG prediction using a modified generative adversarial network (GAN). As a recent breakthrough in the field of deep learning, GANs have shown promising performance on various tasks, such as generating realistic images [13], synthesizing electronic health records [4] and predicting financial time series [31]. Normally, a GAN framework is composed of two deep neural networks (DNNs) models as the generator and the discriminator, respectively. They are trained simultaneously through an adversarial process [10]. The proposed generator captures feature maps of the multi-variant physiological waveform data and generates predictive BG samples, while the discriminator is designed to distinguish the real data from generated ones. To model the temporal dynamics of BG data, we adopt a recurrent neural network (RNN) in the generator and a one-dimensional convolutional neural network (CNN) in the discriminator with dilation factors in each DNN layer to expand receptive fields, which have been verified as adequate network structures for BG prediction in our previous works [5, 17, 33].

2 METHODS

2.1 Dataset and Pre-processing

The data that we used to develop the model is the OhioT1DM dataset, provided by the Blood Glucose Level Prediction (BGLP) Challenge [20, 21]. It was produced by collecting BG-relevant data on 12 people with T1D over an eight-week period. The first half of the cohort released for the 2018 BGLP challenge was used for model pre-training, and we focus on the performance of the rest six individuals that numbered 540, 544, 552, 567, 584, and 596. The dataset contains BG levels collected by CGM readings every five minutes, insulin delivery from insulin pumps, self-reported events (such as meal, work, sleep, psychological stress, and physical exercise) via a smartphone app and physical activity by a sensor band. However,

[1] Imperial College London, UK, email: taiyu.zhu17@imperial.ac.uk
[2] Imperial College London, UK, email: x.yao19@imperial.ac.uk
[3] University College London, UK, email: ken.li@ucl.ac.uk
[4] Imperial College London, UK, email: pherrero@imperial.ac.uk
[5] Imperial College London, UK, email: pantelis@imperial.ac.uk

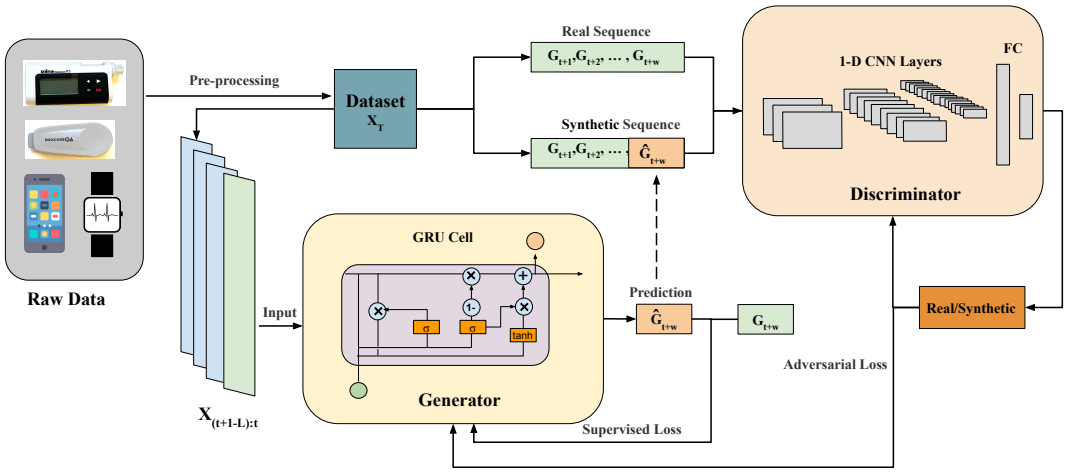


Figure 1: The system architecture of the proposed GAN framework to predict BG levels.

there are unavoidable differences between the collected data and actual physiological states. For example, the CGM sensor measures interstitial fluid glucose level and then estimate BG levels by applying signal processing techniques, such as filtering and calibration algorithms. The meal and insulin are discrete values manually input by users, instead of series of carbohydrates and insulin on board.

It should be noted that the dataset contains many missing gaps and outliers affecting BG levels, both in the training and testing sets, mainly due to CGM signal loss, sensor noise (e.g., compression artifacts), or some usage reasons, such as sensor replacement and calibration. To compensate for some of the missing data, we apply linear interpolation to fill the missing sequences in the training sets, while we only extrapolate missing values in the testing set to ensure that the future information is not involved as partial inputs in the prediction. We then align processed BG samples and other features, e.g. exogenous events, with the same resolution of CGM measurements, and normalize them to form a N-step time series: $\mathbf{X}_N = [\mathbf{x}_1, \dots, \mathbf{x}_N] \in \mathbb{R}^{N \times d}$, where $\mathbf{x}$ is a d-dimensional vector mapping the multivariate data at each timestep.

2.2 Problem Formulation

Considering a target prediction horizon (PH) (e.g. 30 or 60 minutes), the goal of the predictor is to estimate the future BG levels G_{t+w} of individuals given past and current physiological states, where w is the number of timesteps determined by PH and CGM resolution (e.g. 5 minutes). Hence, the objective of predictor is consistent with that of GANs, aiming to learn the DNN approximator $\hat{p}$ from the pattern of glucose dynamics p measured in the human body, which can be expressed by the form of the Kullback-Leibler divergence [30]:

$$\min_{\hat{p}} \mathrm{D}((p(G_{t+w}|\mathbf{X}_{1:t})||(\hat{p}(G_{t+w}|\mathbf{X}_{1:t})) \tag{1}$$

where D is a measurement of the distance between distributions. Thus, we need to select highly-related data features to represent the physiological state. Referring to some previous work and hyper-parameter tuning [16, 17, 22, 23], we use $\mathbf{X} \triangleq [\mathbf{G}, \mathbf{M}, \mathbf{I}]$ as the physiological time series, where $\mathbf{G}$ is pre-processed CGM measurements (mg/dL); $\mathbf{M}$ denotes the carbohydrate amount of meal ingestion (g); and $\mathbf{I}$ is the bolus insulin delivery (U). In order to reduce the bias in the supervised learning, we set the changes of BG levels in PH as the training targets of the generator: $\Delta G_t = G_{t+w} - G_t$. Then the predictive BG level $\hat{G}_{t+w}$ from the generator is defined as follows:

$$\hat{G}_{t+w} = f_{\mathcal{G}}(\mathbf{X}_{t+1-L:t}) + G_t \tag{2}$$

where $f_{\mathcal{G}}$ represents the parameters of the generator. Instead of using the whole series, we divide $\mathbf{X}$ into small contiguous sequences with a length of L as a sliding window, then feed them into the deep generative model in a form of mini-batches, aiming at improving stability and generalization of the model [12]. According to the feature selection in [22] and the model validation, we empirically set $L = 18$ which indicates that the input contains 1.5 hours historical data.

2.3 System Architecture

The RNN-based algorithms performed well in BG level prediction in previous studies [1, 23, 28]. Thus, we instantiate a three-layer RNN with 32 hidden units to build the generator, which can be seen as a typical setup of time series GANs [9, 24, 30]. In general, vanilla RNN architecture faces the problem of gradient vanishing and exploding, making it difficult to capture long-term dependencies. Thus, the gated RNN units are proposed to meet this challenge using element-wise gating functions [7], including long short-term memory (LSTM) units [11] and gated recurrent units (GRUs) [6]. Compared to the vanilla RNN, the gated units are able to control the flow of information inside units by a set of gates, which allows an easier backward propagation process. Compared to the LSTM, the GRU was proposed more recently and removed the memory unit. This cell structure uses less parameters and computes the output more efficiently [32] . During the hyper-parameter tuning, GRU-based algorithms also achieved the best predictive outcomes, so we naturally adopt GRU cells in the RNNs .

As depicted in Figure 1, the multi-dimensional input is fed into a RNN with GRU cells given a state length of L. Then the data is processed by a set of hidden neurons to calculate the last cell state C_t. A fully connected (FC) layer with weights $\mathbf{W}_{FC}$ and a bias b_{FC} are used to model the final scalar output: $\Delta\hat{G}_t = \mathbf{W}_{FC}C_t + b_{FC}$. Finally, after adding the current BG level to predictive glucose change, we obtain the output $\hat{G}_{t+w}$.

In general, the prediction performance degrades with the increase of PH, due to the complicated physiological conditions of people with T1D and the uncertainties of exogenous events between t and $t+w$. For instance, if there was a meal intake with large carbohydrate

20-30 minutes before $t+w$, the BG level would raise fast and make the target ΔG_t suddenly increase. These cases occur frequently in the daytime with a large PH, which could affect a supervised learning model to achieve global optimum. This motivated us to make use of the information between t and $t+w$ during the training process to investigate the contiguous glucose change. Therefore, we append the predictive BG level to the end of series $G_{t+1:t+w-1}$ to form a synthetic sequence $\hat{\mathbf{y}}$ and use $G_{t+1:t+w}$ as the corresponding real sequence $\mathbf{y}$. Then we introduce a CNN-based discriminator to extract features and distinguish the real from synthetic sequences, benefiting from the good classification ability of CNNs [15]. There are three one-dimensional (1-D) causal CNN layers employed with rectified linear unit (ReLU) activation and 32 hidden units to compute the final binary output. The discriminator is expected to classify the real and synthetic sequences by 1 and 0, while the generator is pitted against the discriminator and aims to estimate a BG value that is close to the real BG distribution over the PH. Thus the loss of discriminator is computed by cross-entropy. Consequently, this adversarial training contains two loss functions $\mathcal{L}_\mathcal{G}$ and $\mathcal{L}_\mathcal{D}$ for the generator and the discriminator respectively, which are given by

$$\mathcal{L}_\mathcal{G} = \lambda_1 \mathcal{L}_{SL} + \lambda_2 m \sum_{i=1}^{m} \log(1 - f_\mathcal{D}(\hat{\mathbf{y}}^{(i)})), \tag{3}$$

$$\mathcal{L}_\mathcal{D} = \frac{1}{m} \sum_{i=1}^{m} [-\log f_\mathcal{D}(\mathbf{y}^{(i)}) - (\log(1 - f_\mathcal{D}(\hat{\mathbf{y}}^{(i)})))], \tag{4}$$

where $f_\mathcal{D}$ represents the calculation in the discriminator; $\mathcal{L}_{SL}$ is the means square error loss of supervised learning: $\mathcal{L}_{SL} = \sum_{i=1}^{m} (G_{t+w}^{(i)} - \hat{G}_{t+w}^{(i)})^2$; λ_1 and λ_2 are used to adjust the ratio between supervised loss and adversarial loss [31]; and m stands for the mini-batch size. In practice, we employ two separate Adam optimizer [14] to minimize $\mathcal{L}_\mathcal{G}$ and $\mathcal{L}_\mathcal{D}$ with batch size of 512 and learning rate of 0.0001.

Moreover, we introduce dilation to both the RNN and the CNN layers [3, 26], which has shown the promising performance of BG level prediction in previous work [5, 17, 32, 33]. By skipping certain number connections between neurons, the receptive field of the DNN layers can be exponentially increased, which is helpful to capture long-term temporal dependencies in the BG series. In particular, the dilation of layer l is set to $r^l = 2^{l-1}$, increasing from the bottom layer to the top layer. The computation of DNN layers are defined as follows:

$$h_t^{(l)} = f_N(h_{t-r^l}^{(*)}, in_t^{(l-1)}) \tag{5}$$

where $h_t^{(l)}$ and $in_t^{(l-1)}$ are the output and input of layer l at timestep t; f_N denotes the computation in hidden neurons, referring to convolution and cell operation in CNN and RNN layers, respectively. As a feed-forward neural network, the CNN hidden units fetch all the inputs from the layer at a lower level ($* = l-1$), whereas RNNs skip cell state by $r^l - 1$ timesteps to perform the recursive operation ($* = l$).

2.4 Training and Validation

The training and testing sets are separately provided by the BGLP challenge, which contains the data for around 40 and 10 days, respectively. To tune the hyper-parameters by grid search, we validated the models by the same range of hyper-parameters values as in our previous work [32]. We considered many validation methods, such as simple splitting, k-fold cross-validation, and blocked cross-validation [2]. Due to the temporal dependencies and limited size of the training set, we use the last 20% data of the training set to validate the models and guarantee that future information is not involved in current prediction. The early-stop technique is applied to avoid overfitting; we stop the training process when the validation loss keeps increasing. In particular, we set the maximum number of epochs to 3000 with stopping patience of 50. The data sufficiency and overfitting occurrences are further investigated by means of the learning curves.

2.5 Metrics

A set of metrics is applied to evaluate the performance of the GAN model, including root mean square error (RMSE) (mg/dL), mean absolute error (MAE) (mg/dL), which are denoted as:

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{k=1}^{N}(G_k - \hat{G}_k)^2}, \ \ \text{MAE} = \frac{1}{N}\sum_{k=1}^{N}|G_k - \hat{G}_k|, \tag{6}$$

In addition to the RMSE and MAE metrics, we also use the Clarke error grid (CEG) [8], which is a semi-quantitative tool from the clinical perspective. As shown in Figure 2, there are five zones labeled to intuitively reveal the medical consequence based on the prediction results. In general, the data points (BG pairs) in zone A and B are regarded as positive for medical treatment, while the rest (C, D and E) are considered undesirable.

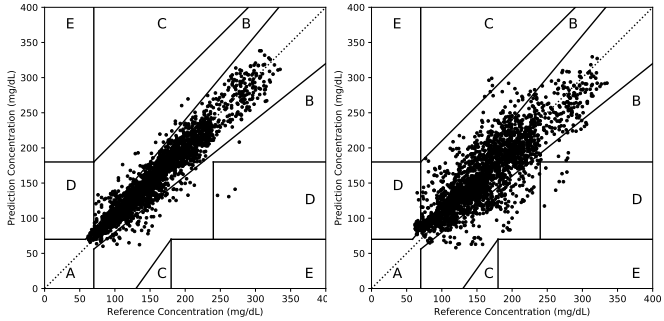
3 RESULTS

After tuning the hyper-parameters, we tested the model on the testing sets. Table 1 shows the RMSE and MAE results for the PH of 30 minutes and 60 minutes. Considering the randomness of the initial weights in DNNs, we conducted 10 simulations and reported results by Mean±SD, where SD is the standard deviation. The average (AVG) RMSE and MAE over all 6 contributors respectively achieve 18.34 ± 0.17 and 13.37 ± 0.18 mg/dL for 30-minute PH, and 32.21 ± 0.46 and 24.20 ± 0.42 for 60-minute PH. The best RMSE and MAE results in experiments are also presented in the last row, which are slightly smaller than the average results. It is noted the standard deviation of multiple simulations is small, which indicates the stability of the model.

Table 1: Prediction performance of the GAN model evaluated on 6 data contributors.

ID	Number	30-minute PH		60-minute PH	
	(#)	RMSE	MAE	RMSE	MAE
540	2884	20.14 ± 0.21	15.22 ± 0.17	38.54 ± 0.46	29.37 ± 0.21
544	2704	16.28 ± 0.11	11.62 ± 0.15	27.64 ± 0.43	20.09 ± 0.38
552	2352	16.08 ± 0.20	12.03 ± 0.22	29.03 ± 0.35	22.47 ± 0.34
567	2377	20.00 ± 0.14	14.17 ± 0.22	35.65 ± 0.41	26.68 ± 0.53
584	2653	20.91 ± 0.08	15.11 ± 0.11	34.31 ± 0.53	25.55 ± 0.52
596	2731	16.63 ± 0.25	12.12 ± 0.23	28.10 ± 0.57	21.06 ± 0.57
AVG		**18.34**	**13.37**	**32.21**	**24.20**
SD		0.17	0.18	0.46	0.42
Best		18.21	13.21	31.64	23.70

To visualize clinical significance between the reference and prediction outcomes, Figure 2 shows the CEG of the contributor 544 that obtains the best statistic performance in Table 1. The specific percentage of the distribution in five regions is presented in Table 2.



(a) 30-minute PH (b) 60-minute PH

Figure 2: The Clarke error grid plots for contributor 544

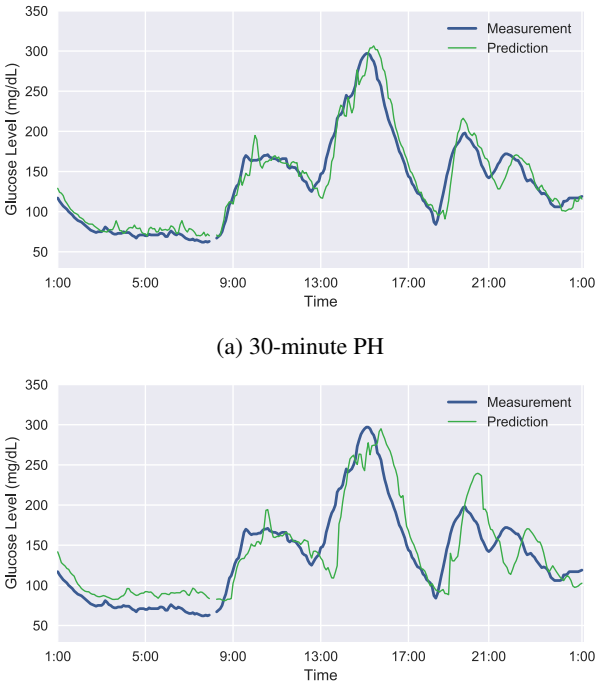
Table 2: The percentage distribution in Clarke error gird (%).

ID	540	544	552	567	584	596
30-minute PH						
CEG_A	86.15	93.91	89.41	89.01	86.75	91.03
CEG_B	12.18	5.76	8.80	10.06	12.26	7.57
CEG_C	0	0	0	0	0	0
CEG_D	1.67	0.33	1.79	0.93	0.98	1.40
CEG_E	0	0	0	0	0	0
60-minute PH						
CEG_A	60.22	79.38	68.01	60.81	69.46	76.60
CEG_B	33.37	19.20	28.91	30.80	28.34	20.78
CEG_C	0.14	0	0	0.25	0.18	0
CEG_D	6.27	1.38	3.08	8.14	2.01	2.62
CEG_E	0	0	0	0	0	0

4 DISCUSSION

As shown in Table 2, the majority of the CEG points are located in zones A and B. These zones signify that the data is within 20% value of the reference, where the treatment suggestions are appropriate regardless of the prediction error. It indicates the high clinical accuracy of the proposed model. The percentage of zone D is small for the 30-minute PH and increases for the 60-minute PH. The points in zone D mean the predictive model missed the hypoglycemia or hyperglycemia events and could lead to poor treatment. In Figure 2b, the most error points are concentrated on the bottom-right corner of the left panel of zone D. It reveals that the model outputs higher predictions when BG levels enter the hypoglycemia region, which is undesirable in the clinical setting. Figure 3 shows the corresponding BG curves for the contributor 544, where the findings from CEG analysis can be validated, and time lags between the predictions and measurements can be observed. The overestimation is observed in several BG regions with low BG levels or a sharp decrease. Aligning the error region with the timesteps, we find that some of the misestimation occurs in nocturnal hypoglycemia. Similar findings are identified by the CEG analysis and BG curves of the other contributors. Therefore, future work will include training and switching between different models for different glucose regions, evaluated by more advanced error grid analysis.

During the experiments, we explored Tikhonov regularization to filter out the outliers in training sets, as described in [1]. However, it was prone to degrade the validation performance but largely reduce the training loss. Then we used the 2018 OhioT1DM dataset [21] and the *in silico* datasets from UVA/Padova T1D simulator [19] for model pre-training. The simulator produced data of an average virtual adult subject with the scenarios defined in [32] over 360 simulated days. The population model was trained by 5 epochs and then fine-tuned



(a) 30-minute PH

(b) 60-minute PH

Figure 3: The comparison between the model predictions and the ground truth of CGM measurements during the first 24-hour period in the testing set of contributor 544. There are three missing BG values between 8:00 and 8:15.

by subject-specific data, but the average validation RMSE slightly increased by around 0.5 mg/dL, compared with the models without pre-training. As shown in Table 1, there are two groups: one including contributors 544, 552, and 596 with better RMSE and MAE performance, and the other including contributors 540, 567 and 584. We introduced the data from the former group to pre-train a population model for the latter group, but the RMSE almost remained unchanged. Thus, one explanation of the pre-training performance is that large inter-person variability exists. For example, in the testing set, contributor 552 has a gap of 1415 missing data points ($\sim$ 5 days), and contributor 567 did not record the meal ingestion, for which we reduced the dimension of the input data. To this end, multiple prepossessing methods are needed to mitigate these missing or incorrect inputs, such as the detection of unannounced meals. In addition, as future work, we consider incorporating personalized physiological and behavioral models [18], such as insulin and carbohydrate on board, to better explain the observed variability.

Compared with the RNN prediction model in our previous work [32], the GAN model achieved better validation performance and smaller RMSE for most of the data contributors in the training process, especially for the 60-minute PH. During the testing phase, the GAN model can output the predictions without using the discriminator. Hence, the complexity of the proposed model is similar to that of the conventional RNN models, which can be easily implemented on smartphone applications [16, 17] to provide real-time predictions and control insulin pump via Bluetooth connectivity. The code corresponding to this work is available at: https://bitbucket.org/deep-learning-healthcare/glugan.

5 CONCLUSION

In this work, a novel deep learning model using a modified GAN architecture is designed to predict BG levels for people with T1D. We developed the personalized models and conducted multiple evaluations for each data contributor in the OhioT1DM dataset. The proposed model achieves promising prediction performance for 30-minute and 60-minute PH in terms of average RMSE and MAE. The CEG analysis further indicates good clinical accuracy, but there are opportunities for enhancement. In particular the model falls short sometimes in capturing a small number of hypoglycemia events. Nevertheless, the model is able capture most of the individual glucose dynamics and has clear potential to be adopted in actual clinical applications.

ACKNOWLEDGEMENTS

The work is supported by EPSRC EP/P00993X/1 and the President's PhD Scholarship at Imperial College London.

REFERENCES

[1] Alessandro Aliberti, Irene Pupillo, Stefano Terna, Enrico Macii, Santa Di Cataldo, Edoardo Patti, and Andrea Acquaviva, 'A multi-patient data-driven approach to blood glucose prediction', *IEEE Access*, **7**, 69311–69325, (2019).

[2] Christoph Bergmeir and José M Benítez, 'On the use of cross-validation for time series predictor evaluation', *Information Sciences*, **191**, 192–213, (2012).

[3] Shiyu Chang, Yang Zhang, Wei Han, Mo Yu, Xiaoxiao Guo, Wei Tan, Xiaodong Cui, Michael Witbrock, Mark A Hasegawa-Johnson, and Thomas S Huang, 'Dilated recurrent neural networks', in *Advances in Neural Information Processing Systems*, pp. 77–87, (2017).

[4] Zhengping Che, Yu Cheng, Shuangfei Zhai, Zhaonan Sun, and Yan Liu, 'Boosting deep learning risk prediction with generative adversarial networks for electronic health records', in *2017 IEEE International Conference on Data Mining (ICDM)*, pp. 787–792. IEEE, (2017).

[5] Jianwei Chen, Kezhi Li, Pau Herrero, Taiyu Zhu, and Pantelis Georgiou, 'Dilated recurrent neural network for short-time prediction of glucose concentration.', in *The 3rd KDH workshop, IJCAI-ECAI 2018*, pp. 69–73, (2018).

[6] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio, 'On the properties of neural machine translation: Encoder-decoder approaches', *arXiv preprint arXiv:1409.1259*, (2014).

[7] Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio, 'Empirical evaluation of gated recurrent neural networks on sequence modeling', in *NIPS 2014 Workshop on Deep Learning, December 2014*, (2014).

[8] William L Clarke, Daniel Cox, Linda A Gonder-Frederick, William Carter, and Stephen L Pohl, 'Evaluating clinical accuracy of systems for self-monitoring of blood glucose', *Diabetes care*, **10**(5), 622–628, (1987).

[9] Cristóbal Esteban, Stephanie L Hyland, and Gunnar Rätsch, 'Real-valued (medical) time series generation with recurrent conditional gans', *arXiv preprint arXiv:1706.02633*, (2017).

[10] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, 'Generative adversarial nets', in *Advances in neural information processing systems*, pp. 2672–2680, (2014).

[11] Sepp Hochreiter and Jürgen Schmidhuber, 'Long short-term memory', *Neural computation*, **9**(8), 1735–1780, (1997).

[12] Elad Hoffer, Itay Hubara, and Daniel Soudry, 'Train longer, generalize better: closing the generalization gap in large batch training of neural networks', in *Advances in Neural Information Processing Systems*, pp. 1731–1741, (2017).

[13] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros, 'Image-to-image translation with conditional adversarial networks', in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1125–1134, (2017).

[14] Diederik P Kingma and Jimmy Ba, 'Adam: A method for stochastic optimization.', *International Conference on Learning Representations 2015*, 1–15, (2015).

[15] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton, 'Deep learning', *nature*, **521**(7553), 436–444, (2015).

[16] Kezhi Li, John Daniels, Chengyuan Liu, Pau Herrero-Vinas, and Pantelis Georgiou, 'Convolutional recurrent neural networks for glucose prediction', *IEEE journal of biomedical and health informatics*, (2019).

[17] Kezhi Li, Chengyuan Liu, Taiyu Zhu, Pau Herrero, and Pantelis Georgiou, 'Glunet: A deep learning framework for accurate glucose forecasting', *IEEE journal of biomedical and health informatics*, (2019).

[18] Chengyuan Liu, Josep Vehí, Parizad Avari, Monika Reddy, Nick Oliver, Pantelis Georgiou, and Pau Herrero, 'Long-term glucose forecasting using a physiological model and deconvolution of the continuous glucose monitoring signal', *Sensors*, **19**(19), 4338, (2019).

[19] Chiara Dalla Man, Francesco Micheletto, Dayu Lv, Marc Breton, Boris Kovatchev, and Claudio Cobelli, 'The uva/padova type 1 diabetes simulator: new features', *Journal of diabetes science and technology*, **8**(1), 26–34, (2014).

[20] C. Marling and R. Bunescu, 'The OhioT1DM dataset for blood glucose level prediction: Update 2020', in *The 5th KDH workshop, ECAI 2020*, (2020). CEUR proceedings in press, available at http://smarthealth.cs.ohio.edu/bglp/OhioT1DM-dataset-paper.pdf.

[21] Cindy Marling and Razvan C. Bunescu, 'The OhioT1DM dataset for blood glucose level prediction', in *The 3rd KDH workshop, IJCAI-ECAI 2018*, pp. 60–63, (2018).

[22] Cooper Midroni, Peter J Leimbigler, Gaurav Baruah, Maheedhar Kolla, Alfred J Whitehead, and Yan Fossat, 'Predicting glycemia in type 1 diabetes patients: experiments with XGBoost', in *The 3rd KDH workshop, IJCAI-ECAI 2018*, pp. 79–84, (2018).

[23] Sadegh Mirshekarian, Hui Shen, Razvan Bunescu, and Cindy Marling, 'Lstms and neural attention models for blood glucose prediction: Comparative experiments on real and synthetic data', in *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 706–712. IEEE, (2019).

[24] Olof Mogren, 'C-rnn-gan: Continuous recurrent neural networks with adversarial training', *arXiv preprint arXiv:1611.09904*, (2016).

[25] Silvia Oviedo, Josep Vehí, Remei Calm, and Joaquim Armengol, 'A review of personalized blood glucose prediction strategies for t1dm patients', *International journal for numerical methods in biomedical engineering*, **33**(6), e2833, (2017).

[26] Tom Le Paine, Pooya Khorrami, Shiyu Chang, Yang Zhang, Prajit Ramachandran, Mark A Hasegawa-Johnson, and Thomas S Huang, 'Fast wavenet generation algorithm', *arXiv preprint arXiv:1611.09482*, (2016).

[27] Pouya Saeedi, Inga Petersohn, Paraskevi Salpea, Belma Malanda, Suvi Karuranga, Nigel Unwin, Stephen Colagiuri, Leonor Guariguata, Ayesha A Motala, Katherine Ogurtsova, et al., 'Global and regional diabetes prevalence estimates for 2019 and projections for 2030 and 2045: Results from the international diabetes federation diabetes atlas', *Diabetes research and clinical practice*, **157**, 107843, (2019).

[28] Qingnan Sun, Marko V Jankovic, Lia Bally, and Stavroula G Mougiakakou, 'Predicting blood glucose with an LSTM and Bi-LSTM based deep neural network', in *2018 14th Symposium on Neural Networks and Applications (NEUREL)*, pp. 1–5. IEEE, (2018).

[29] Ashenafi Zebene Woldaregay, Eirik Årsand, Ståle Walderhaug, David Albers, Lena Mamykina, Taxiarchis Botsis, and Gunnar Hartvigsen, 'Data-driven modeling and prediction of blood glucose dynamics: Machine learning applications in type 1 diabetes', *Artificial intelligence in medicine*, (2019).

[30] Jinsung Yoon, Daniel Jarrett, and Mihaela van der Schaar, 'Time-series generative adversarial networks', in *Advances in Neural Information Processing Systems*, pp. 5509–5519, (2019).

[31] Xingyu Zhou, Zhisong Pan, Guyu Hu, Siqi Tang, and Cheng Zhao, 'Stock market prediction on high-frequency data using generative adversarial nets', *Mathematical Problems in Engineering*, **2018**, (2018).

[32] Taiyu Zhu, Kezhi Li, Jianwei Chen, Pau Herrero, and Pantelis Georgiou, 'Dilated recurrent neural networks for glucose forecasting in type 1 diabetes', *Journal of Healthcare Informatics Research*, 1–17, (2020).

[33] Taiyu Zhu, Kezhi Li, Pau Herrero, Jianwei Chen, and Pantelis Georgiou, 'A deep learning algorithm for personalized blood glucose prediction.', in *The 3rd KDH workshop, IJCAI-ECAI 2018*, pp. 64–78, (2018).

Personalized Machine Learning Algorithm based on Shallow Network and Error Imputation Module for an Improved Blood Glucose Prediction

Jacopo Pavan, Francesco Prendin, Lorenzo Meneghetti, Giacomo Cappon, Giovanni Sparacino, Andrea Facchinetti, Simone Del Favero [1]

Abstract. Real-time forecasting of blood glucose (BG) levels has the potential to drastically improve management of Type 1 Diabetes, a widespread chronic disease affecting the metabolic system. Most notably, if hypo or hyperglycemia episodes (i.e. glycemic excursion below or above a safe range) could be accurately predicted, then the patient could be timely warned, thus enabling proactive countermeasures to avoid these dangerous conditions. In this work, a novel personalized algorithm for the real-time forecasting of BG is developed by combining the output of a shallow feed forward neural network with an error imputation module composed by an ensemble of trees. Past glucose readings as well as insulin, meals and work/sleep time information are carefully handled to train and boost the prediction performance of the algorithm. The root mean square error over the 6 subjects achieves a mean value of 18.69 mg/dL and 32.43 mg/dL for 30- and 60-minute prediction horizon respectively.

1 INTRODUCTION

Type 1 diabetes (T1D) is a metabolic disease characterized by the destruction of the pancreatic cells responsible for insulin production and thus resulting in an impaired Blood Glucose (BG) homeostasis. Diabetes treatment relies on external insulin injection aimed at maintaining BG levels within a physiological range [1], since poor BG regulation is responsible of comorbidities and reduced life expectancy [2]. In particular, concentrations below or above the normal range (called hypo- and hyper-glycemia respectively) can either represent an immediate threat to patient safety or could cause severe long-term complications.

Measures to mitigate/avoid these conditions include the administration of exogenous insulin to reduce hyperglycemic excursion or the consumption of fast-acting carbohydrates for hypoglycemic events. Unfortunately, both insulin injection and carbohydrates consumption affects BG only after a considerable amount of time: 45 minutes for insulin and at least 15 minutes for carbohydrates.

Therefore, accurate BG prediction would be of paramount importance to enable timely corrective actions and thus ensure successful BG control. This holds true both for standard therapy, where corrective actions are manually performed by patients, and in automated and semi-automated systems such as an artificial pancreas [1].

Nevertheless, BG prediction is by no mean a trivial task. BG concentration is the result of complex non-linear dynamical interaction of multiple physiologic subsystems and is influenced by several factors, often hard to measure. They include timing and magnitude of carbohydrates consumption, protein and fat content of the meal, length, intensity and even type (aerobic vs anaerobic) of physical exercise, stress, illness or menstrual cycle. Furthermore, the modeling of BG dynamics is hindered by the large variability in the physiological metabolic response of different individuals. For these reasons, non-linear and personalized models can represent one of the best options to address the task. The introduction of continuous glucose monitoring (CGM) sensors in T1D care has encouraged the use of data-driven models based on past BG readings, whereas the further availability of data offered by infusion pumps and fitness bands opened the possibility of using exogenous inputs as additional features for describing the model [15, 3].

Over the last two decades, several non-linear algorithms have been tested in this framework, including support vector machine [4], gaussian process regression [14], random forest (RF) [5] and several kinds of neural networks (NN) [17, 10, 16, 9], up to deep-learning approaches like long short-term memory networks [13] and convolutional NN [6]. While some works assessed that only past CGM information is actually useful to describe an accurate model [6, 13, 8], many others claims that exogenous inputs play an important role in describing BG dynamics. These extra features include time of the day, insulin administration, food intake, energy expenditure, lifestyle and emotions [5, 4, 6, 10, 18].

Up to the present, none of the these models has stand out from the others in terms of prediction accuracy. This consideration lead our group to focus more on feature manipulation, selection and hyperparameters optimization and to explore the possibility of combining different kinds of non-linear learners. In conclusion, the aim of our work is to synthesize an accurate model of BG dynamics by starting from a simple, feedforward NN and to investigate:

- the impact of hyperparameters optimization and feature selection;
- the improvement achievable by combining the NN with an error imputation module (EIM) based on a regression trees ensemble.

2 DATASET

Our study is based on the real patient data provided by the OhioT1DM Dataset, described in detail in [7]. In its 2020 update, six new patients are introduced in the dataset, with roughly 8 weeks of data each (6 weeks of training set, 10 days of testing set). Each of them wore a CGM device (Medtronic Enlite CGM sensor), and an insulin pump (Medtronic 530G or 630G). Daily life-events (e.g.

[1] Department of Information Engineering, University of Padova, Italy, email: {pavanjac, prendinf, meneghet, cappongi, gianni, facchine, sdelfave}@dei.unipd.it

work/sleep, exercise) are reported via smartphone app while other physiological data (e.g. skin temperature) are provided by using Empatica Embrace fitness wristbands. We will work exclusively on these six new subjects.

Some of these signals are quasi-continuously measured, like CGM, acceleration or basal insulin. Some others are impulse-like, e.g. self-monitoring BG or meal consumption, which are provided only a few number of times along the day. Some others, like work intensity or sleep quality, are instead defined for time windows of the order of some hours.

Some of the impulsive-like features, e.g. insulin boli, have an impact on glucose dynamics that could last up for several hours. In order to include this information in the framework of feedforward NNs, which have no memory about the dynamics of inputs, we had to rely on new features. We described the insulin-on-board (IOB) by convolution of insulin boli and basal with a 6 hours activity curve, whereas the convolution of consumed carbohydrates with an absorption curve (different for slow- and fast-acting carbohydrates) returned the carbohydrates-on-board (COB). These two variables carry information about the dynamics of slow insulin absorption and carbohydrates slow impact on BG, hence they are suitable for being used with feedforward NNs. In a similar way, we described the physical-exercise-on-board by low-pass filtering with second order transfer function the physical activity intensity.

We also introduced the slope of CGM, computed by using the last 2 hours of readings, since the trend of the glycemic profile resulted to be significantly correlated to future CGM readings in the training set. Other tested features include daytime, time and amount of last carbohydrates intake and several filtered version of the original signals.

3 METHODS

As shown in Figure 1, the proposed model is composed by two parts. The first one is a shallow NN, which is the main predictor. It is trained to predict future BG values with a certain prediction horizon (PH). The second predictor is based on an ensemble of trees. It is called error imputation module, since it is trained to predict the error that is committed by the shallow NN. Finally, as shown in Figure 1, the prediction of the proposed algorithm is obtained by combining the output of the shallow neural network $CGM_s(t+PH)$ with the output of EIM, $\hat{e}(t+PH)$, to have an accurate value for the expected glucose concentration.

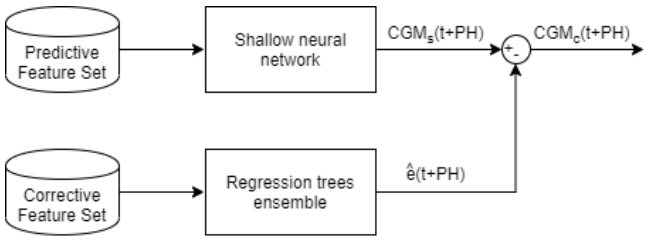


Figure 1: Complete architecture of the predictive model. Variable $CGM_s(t+PH)$ is the original prediction, $CGM_c(t+PH)$ is the corrected prediction and $\hat{e}(t+PH)$ is the predicted error.

3.1 SHALLOW NEURAL NETWORK

The feature set for the shallow NN is manually determined on a population level, i.e. by looking for a unique set of feature which can be used by all subjects. The first criterion for feature selection consists in excluding all those features that presents too many missing values in the training set and hence cannot be considered reliable. For instance, COB was discarded because, in some subjects, the information about meal is missing for a large part of the training set (e.g. subject 567 reported only 31 of the expected 141 meals). A second criterion consists in excluding the features that are expected to have a negligible impact on prediction accuracy, e.g. the state of illness or work. This selection was performed with domain experts and by means of some preliminary evaluation on the training set. The selected features still presented many missing values that could hinder the training procedure. Thus we performed, exclusively on the training set, a first order interpolation on any gap of samples shorter than 30 minutes. Finally, we investigated how many past CGM readings should be used as inputs in order to improve model accuracy. To this purpose we performed a bayesian optimization [12], using 70% of the data in the original training set for training and the remaining 30% for validation. The result was that using the last 16 CGM instants (corresponding to the last 1 hour and 20 minutes of readings) leads to the optimal prediction performances in the six subjects. In conclusion, the resulting feature pool we adopted included present and past CGM readings, CGM slope and IOB.

Similarly, we determined the number of hidden layers and number of neurons in each layer with an exploratory optimization, which was also performed via bayesian optimization and using the same training/validation split we employed for feature selection. The best performances on the validation set were found when using a single hidden layer with a reduced number of inner nodes (only 5).

While the architecture of the net was optimized on a population level and it will be the same for every subject, the weights of the net were trained individually for every subject, meaning that the resulting model is tailored on each of them. The shallow net is trained on the whole original training set, with its target being either the 6- or 12-steps ahead prediction of CGM, i.e. prediction horizons of 30 and 60 minutes, respectively. Inputs and targets of the net are normalized by the mean and standard deviation computed on the training set.

3.2 ERROR IMPUTATION MODULE

We noticed that the prediction of the main NN is affected by a large error when abrupt changes occur in its inputs. For instance, consecutive CGM readings showing a large difference in values are often associated with poor BG predictions. However, other explanatory factors for the prediction error could be found in those features which were not used by the shallow NN. The key idea behind the EIM is to provide an estimate $\hat{e}(t+PH)$ of the true error, $e(t+PH)$, affecting the prediction of the shallow net. The first step to build this module is to create a new pool of feature, named Corrective Feature Set, containing the first order differences, at several time lags, of: CGM, IOB, COB, sleep/work period, skin temperature and acceleration data. Then, to take into account the large inter-individual variability, a feature selection step (based on ReliefF [11]) is applied to the Corrective Feature Set. Then, only the features with highest ranks are used to train the model. A further level of individualization is achieved by optimizing several hyperparameters. For each subject, a bayesian optimization procedure returns the best method to train the ensemble of trees (i.e., Bagging or Boosting), the best number of trees (searched among the range $\{10,500\}$), the best number of leaves for each tree (searched among the log-scaled range $\{1,\max(2,\text{number of training sample}/2)\}$), the best tree-depth (searched among the log-scaled range $\{1,\max(2,\text{number of train-}$

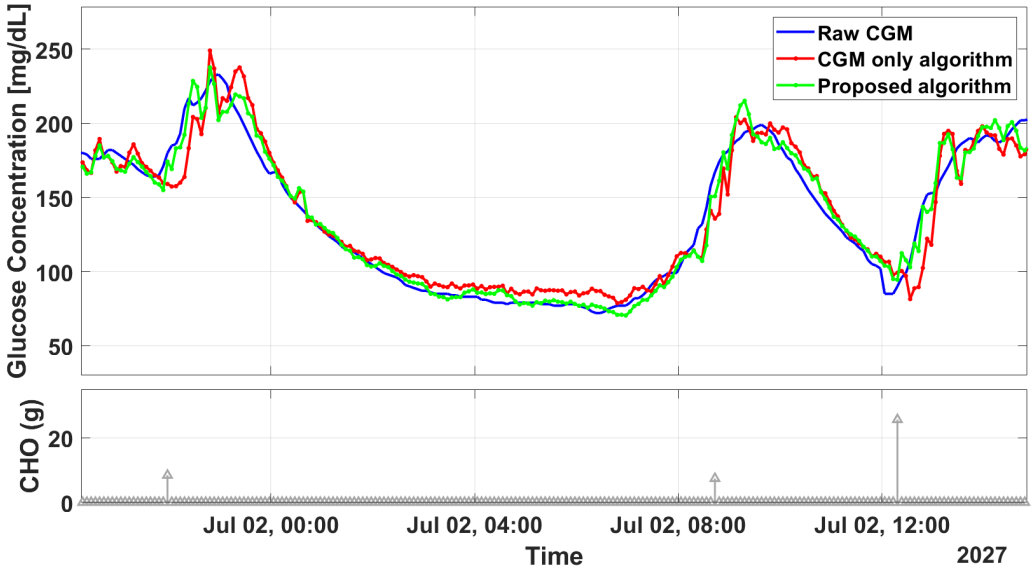


Figure 2: 30-minute prediction for subject 544. The three curves represent real CGM data (blue line), the prediction with CGM-NN (red line), and the prediction with NN-EIM (green line).

ing sample-1)}), the best learning rate (among the log-scaled range $\{0.001,1\}$ if boosting method is chosen). Both the feature selection and the hyperparameters optimization exploits the training set only.

3.3 BENCHMARK NEURAL NETWORK

The effectiveness of the proposed approach is assessed by comparing the predicted profiles with the ones obtained by a benchmark predictor: a shallow neural network based on CGM data only (CGM-NN). This network resorts the structure of the main predictor (i.e. a single hidden layer with 5 inner nodes) but it exploits past CGM data only (16 samples) as input features. This model will also be personalized for each patient.

4 METRICS

The accuracy of the predicted profiles is evaluated by using four metrics. The Mean Absolute Error (MAE) is defined as:

$$MAE = \frac{1}{N}\sum_{t=1}^{N}(y(t) - \hat{y}(t|t-PH))$$

where PH is the prediction horizon, $y(t)$ is the current CGM reading, N is the length of the whole signal y and $\hat{y}(t|t-PH)$ is the the PH-steps ahead prediction using the information available up to instant t. Similarly we can define the Root Mean Square Error (RMSE):

$$RMSE = \sqrt{\frac{1}{N}\sum_{t=1}^{N}(y(t) - \hat{y}(t|t-PH))^2}$$

and Coefficient of Determination (COD):

$$COD = 100 \cdot (1 - \frac{||(y(t) - \hat{y}(t|t-PH))||_2^2}{||(y(t) - \bar{y}(t))||_2^2})$$

where $\bar{y}(t)$ is the average value of y. COD counts for the variance explained by the predictive model with respect to the total variance of the signal. Its maximum value is 100%.

The delay existing between the target signal and the predicted one can be computed as the temporal shift that minimizes the square of the mean quadratic error between these two signals:

$$delay = \underset{j\in[0,PH]}{\arg\min}\left[\frac{1}{N}\sum_{t=1}^{N-PH}((\hat{y}(t|t-PH)+j) - y(t))^2\right].$$

5 RESULTS

Since the shallow net is initialized with random weights, results will be reported in terms of mean and standard deviation of the various metrics on 10 different initialization of the algorithm.

We compared two models: one is the benchmark, shallow NN using exclusively past and present CGM readings (CGM-NN); the other is the shallow net employing the Predictive Feature Set and the error imputation module (NN-EIM). Table 1 and Table 2 reports the results obtained with these two models, on each subject, for 30 and 60 minutes prediction horizons respectively. The last row of the tables averages the mean values of the metrics on every subject.

As aforementioned in Section 3.1, no kind of operation was performed on the testing set in order to impute missing values. Table 3 reports the number of CGM samples available for each subject and the number of those predicted by our CGM-NN and NN-EIM.

NN-EIM achieves better results on each subject for all the evaluation metrics, both for PH = 30 and PH = 60 minutes. On average, the RMSE improves from 19.50 mg/dL with CGM-NN to 18.63 mg/dL with NN-EIM on the 30 minutes PH (p-value=0.031) and from 34.26 mg/dL with CGM-NN to 32.27 mg/dL with NN-EIM (p-value=0.031). The p-values are computed with a Wilcoxon signed rank test and show that the improvement, albeit small ($\sim 5\%$ in both cases), is statistically significant with $1-\alpha = 0.95$ confidence level.

Figure 3 show the boxplots and scatter plots of the average RMSE values for every subject, for a PH of 30 minutes (Figure 3a) and 60 minutes (Figure 3b). The color of the lines linking the scatter plots indicates the magnitude of the difference in RMSE between the two strategies: green-shaded lines mean an improved accuracy from CGM-NN to NN-EIM; viceversa, the lines are red if model accuracy worsen. Both for 30 and 60 minutes ahead predictions, the use

Table 1: Evaluation of RMSE, MAE, COD and delay (mean (± standard deviation)) with CGM-NN and NN-EIM on a 30 minutes PH.

Subj	RMSE		MAE		COD		delay	
	CGM-NN	NN-EIM	CGM-NN	NN-EIM	CGM-NN	NN-EIM	CGM-NN	NN-EIM
540	21.66 (±0.20)	20.42 (±0.27)	14.09 (±0.13)	12.81 (±0.14)	90.29 (±0.18)	91.58 (±0.22)	20 (±0.0)	15.0 (±0)
544	17.67 (±0.08)	16.50 (±0.22)	10.52 (±0.09)	9.57 (±0.13)	89.86 (±0.10)	91.33 (±0.24)	20.0 (±0.0)	15.0 (±0)
552	16.81 (±0.15)	16.51 (±0.12)	6.85 (±0.13)	6.51 (±0.07)	90.72 (±0.16)	91.38 (±0.12)	20.0 (±0.0)	17.5 (±2.6)
567	20.73 (±0.10)	20.18 (±0.14)	10.92 (±0.09)	10.19 (±0.08)	86.89 (±0.13)	88.14 (±0.16)	20.0 (±0.0)	17.5 (±2.6)
584	22.44 (±0.14)	21.29 (±0.33)	12.77 (±0.14)	11.34 (±0.18)	87.69 (±0.15)	89.71 (±0.32)	25.0 (±0.0)	25.0 (±0.0)
596	17.71 (±0.29)	16.89 (±0.24)	10.93 (±0.08)	10.09 (±0.15)	88.28 (±0.38)	89.53 (±0.30)	25.0 (±0.0)	20.0 (±0.0)
All	19.50 (±2.39)	18.63 (±2.22)	11.01 (±2.45)	10.08 (±2.10)	88.95 (±1.54)	90.28 (±1.37)	21.6 (±2.5)	18.3 (±3.7)

Table 2: Evaluation of RMSE, MAE, COD and delay (mean (± standard deviation)) with CGM-NN and EIM-NN on a 60 minutes PH.

Subj	RMSE		MAE		COD		delay	
	CGM-NN	NN-EIM	CGM-NN	NN-EIM	CGM-NN	NN-EIM	CGM-NN	NN-EIM
540	40.34 (±0.82)	37.69 (±0.60)	26.46 (±0.51)	23.83 (±0.36)	66.61 (±1.36)	71.53 (±0.91)	45.0 (±0.0)	40.0 (±0)
544	31.21 (±0.31)	28.74 (±0.32)	19.57 (±0.30)	16.61 (±0.21)	68.55 (±0.64)	73.90 (±0.60)	44.5 (±1.5)	27.0 (±2.58)
552	30.45 (±0.17)	29.56 (±0.23)	12.56 (±0.14)	11.89 (±0.18)	70.12 (±0.33)	72.93 (±0.43)	45.0 (±0.0)	40.0 (±0.0)
567	37.37 (±0.29)	36.28 (±0.40)	20.48 (±0.24)	18.65 (±0.12)	67.13 (±0.65)	62.70 (±0.83)	45.0 (±0.0)	41.0 (±2.1)
584	36.85 (±0.27)	33.84 (±0.26)	21.26 (±0.32)	18.37 (±0.18)	67.13 (±0.49)	74.52 (±0.40)	53.0 (±2.4)	46.0 (±2.1)
596	29.32 (±0.23)	27.51 (±0.57)	18.58 (±0.20)	16.74 (±0.25)	68.31 (±0.51)	72.53 (±1.15)	49.0 (±2.1)	42.5 (±2.6)
All	34.26 (±4.5)	32.27 (±4.25)	19.82 (±4.49)	17.69 (±3.87)	66.55 (±4.08)	71.35 (±4.36)	47.0 (±3.5)	39.4 (±6.4)

Table 3: Number of CGM samples available in the testing set of the OhioT1DM Dataset 2020 and number of CGM samples predicted by CGM-NN and NN-EIM.

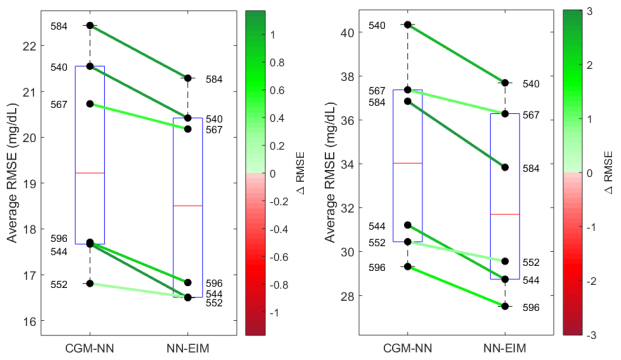
Subj	Available samples	Predicted samples	
		PH=30	PH=60
540	2884	2610	2592
544	2704	2532	2514
552	2352	2061	2020
567	2377	2050	1996
584	2653	2206	2165
596	2731	2551	2521

of exogenous inputs and EIM results in a systematic improvement of performances.

The RMSE of the NN-EIM ranges from 16.50 mg/dL up to 21.29 mg/dL with PH=30 for subject 544 and 584, respectively. This difference might be related to the occurrence of large oscillations in CGM data which cannot be clearly explained by any input variable in the dataset. Considering a PH = 30 min, the lowest improvement in terms of RMSE is for subject 552 (16,81 mg/dL vs 16.51 mg/dL, CGM-NN vs NN-EIM respectively). The largest one is for subject 540 (21.66 mg/dL vs 20.42 mg/dL, CGM-NN vs NN-EIM respectively). One of the main reasons linked to this marginal improvement is the large number of missing values in the testing set for some features (e.g. acceleration and skin temperature).

Nevertheless, whenever the feature set is reliable — as for subject 544 in Figure 2 — the proposed algorithm (green line) reduces the prediction delay as well as the error when CGM data show a positive increment related to any external input (i.e. CHO ingested, in this case). Furthermore, the use of lagged first order differences of past CGM data provides a prediction which is more adherent to the target CGM than the one obtained by CGM-NN. This is confirmed by an enhanced COD for subject 552 (90,72% vs 91,38%, CGM-NN vs NN-EIM and by the delay (20 min vs 17,5 min, CGM-NN vs NN-EIM).

Same conclusions can be found by considering a PH = 60 min. As before, the lowest improvement in terms of RMSE is for subject 552 (30,45 mg/dL vs 29,56 mg/dL, CGM-NN vs NN-EIM). The largest is for subject 584 (36,85 mg/dL vs 33,84 mg/dL, CGM-NN vs NN-EIM). The prediction capabilities of the proposed approach are also confirmed by COD and delay, even when a marginal improvement in terms of RMSE is found. In fact, for subject 552 we found the COD for CGM only NN vs proposed algorithm is 70,12% vs 72,93%. Delay is reduced at 40 minutes, by meaning that the prediction has a useful time anticipation of 20 minutes.



(a) PH=30 minutes (b) PH=60 minutes

Figure 3: Boxplot and scatter plots of average RMSE obtained with CGM-NN and EIM-NN on a 30 minutes (a) and 60 minutes (b) prediction horizons.

6 CONCLUSIONS

In this work we presented a new approach for a real-time forecasting of glucose levels based on a shallow neural network and an error imputation module (NN-EIM). Comparing the performance, at PH = 30 and PH = 60, of the novel algorithm vs CGM-NN, we demonstrated that accurate feature manipulation and selection steps can effectively improve the prediction accuracy and reduce the delay affecting the

predicted profile. However, the presence of many missing values in some variables reduce the improvement brought by the proposed approach. Finally, a further development of this work is the investigation of patterns within the CGM time series and the inclusion of such physiological priors into the proposed algorithm could results in improved performance.

ACKNOWLEDGMENTS

This work was partially supported by MIUR, (Italian Minister for Education, under the initiatives "Departments of Excellence" (Law 232/2016) and "SIR: Scientific Independence of young Researchers", project RBSI14JYM2 "Learn4AP: Patient-Specific Models for an Adaptive, Fault-Tolerant Artificial Pancreas".

CODE

A repository containing the code developed for this work is available at: `https://github.com/jp993/BGPC_2020`

REFERENCES

[1] C. Cobelli, C. Dalla Man, G. Sparacino, L. Magni, G. De Nicolao, and B. Kovatchev, 'Diabetes: Models, signals and control', *IEEE Rev Biomed*, **2**, 54–96, (2009).

[2] L.A DiMeglio, C. Evans-Molina, and R. Oram, 'Type 1 diabetes', *Lancet*, **391**, 2449–62, (2018).

[3] A. Gani, A.V. Gribok, S. Rajaraman, W.K. Ward, and J. Reifman, 'Predicting subcutaneous glucose concentration in humans: data-driven glucose modeling', *IEEE Trans Biomed Eng*, **56**, 246–54, (2009).

[4] E. Georga, V.C. Protopappas, D. Ardigo, M. Marina, I. Zavaroni, D. Polyzos, and D.I. Fotiadis, 'Multivariate prediction of subcutaneous glucose concentration in type 1 diabetes patients based on support vector regression', *IEEE J Biomed Health Inform*, **17**, 71–81, (2013).

[5] E. Georga, V.C. Protopappas, D. Polyzos, and D. Fotiadis, 'A predictive model of subcutaneous glucose concentration in type 1 diabetes based on random forest', *Proceedings of the 34th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2889–92, (2012).

[6] K. Li, J. Daniels, C. Liu, P. Herrero, and P. Georgiou, 'Convolutional recurrent neural networks for glucose prediction', *IEEE J Biomed Health Inform*, **24**, 603–613, (2019).

[7] C Marling and R. Burnescu, 'The ohiot1dm dataset for blood glucose level prediction: Update 2020', (2019).

[8] J. Martinsson, A. Schliep, B. Eliasson, C. Meijner, S. Persson, and O. Mogren, 'Automatic blood glucose prediction with confidence using recurrent neural networks', *Proceedings of the 3rd International Workshop on Knowledge Discovery in Healthcare Data*, 64–68, (2018).

[9] S.M. Pappada, B.D. Cameron, and P.M. Rosman, 'Development of a neural network for prediction of glucose concentration in type 1 diabetes patients', *J Diabetes Sci Technol*, **2**, 792–801, (2008).

[10] C. Perez-Gandia, A. Facchinetti, G. Sparacino, C. Cobelli, E.J. Gḿez, M. Rigla, A. de Leiva, and M.E. Hernando, 'Artificial neural network algorithm for online glucose prediction from continuous glucose monitoring', *Diabetes Technol Ther*, **12**, 81–88, (2010).

[11] M. Robnik-Sikonha and I. Kononenko, 'Theoretical and empirical analysis of relieff and rrelieff', *Machine Learning*, **53**, 23–69, (2003).

[12] J. Snoek, H. Larochelle, and R.P. Adams, 'Practical bayesian optimization of machine learning algorithms', *Advances in Neural Information Processing Systems*, **4**, 2951–2959, (2012).

[13] Q. Sun, M.V. Jankovic, L. Bally, and S. G. Mougiakakou, 'Predicting blood glucose with an lstm and Bi-LSTM based deep neural network', *2018 14th Symposium on Neural Networks and Applications (NEUREL)*, 1–5, (2018).

[14] J. Valletta, A. Chipperfield, and C. Byrne, 'Gaussian process modelling of blood glucose response to free-living physical activity data in people with type 1 diabetes', *Proceedings of the 31st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 4913–6., (2009).

[15] A.Z. Woldaregay, E. Å rsand, W. Stå le, A. David, L. Mamykina, T. Botsis, and G. Hartvigsen, 'Data-driven modeling and prediction of blood glucose dynamics: Machine learning applications in type 1 diabetes', *Artificial Intelligence in Medicine*, **98**, 109–134, (2019).

[16] Z. Zainuddin, O. Pauline, and C. Ardil, 'A neural network approach in predicting the blood glucose level for diabetic patients', *Int J Comput Intell*, **5**, 72–79, (2009).

[17] C. Zecchin, A. Facchinetti, G. Sparacino, and C. Cobelli, 'Jump neural network for online short-time prediction of blood glucose from continuous monitoring sensors and meal information', *Comput Methods Programs Biomed*, **113**, 144–52, (2013).

[18] C. Zecchin, A. Facchinetti, G. Sparacino, and C. Cobelli, 'How much is Short-Term glucose prediction in type 1 diabetes improved by adding insulin delivery and meal content information to cgm data? a Proof-of-Concept study', *J Diabetes Sci Technol*, **10**, 1149–60, (2016).

Experiments in non-personalized future blood glucose level prediction

Robert Bevan [1] and **Frans Coenen** [2]

Abstract. In this study we investigate the need for training future blood glucose level prediction models at the individual level (i.e. per patient). Specifically, we train various model classes: linear models, feed-forward neural networks, recurrent neural networks, and recurrent neural networks incorporating attention mechanisms, to predict future blood glucose levels using varying time series history lengths and data sources. We also compare methods of handling missing time series data during training. We found that relatively short history lengths provided the best results: a 30 minute history length proved optimal in our experiments. We observed long short-term memory (LSTM) networks performed better than linear and feed-forward neural networks, and that including an attention mechanism in the LSTM model further improved performance, even when processing sequences with relatively short length. We observed models trained using all of the available data outperformed those trained at the individual level. We also observed models trained using all of the available data, except for the data contributed by a given patient, were as effective at predicting the patient's future blood glucose levels as models trained using all of the available data. These models also significantly outperformed models trained using the patient's data only. Finally, we found that including sequences with missing values during training produced models that were more robust to missing values.

1 Introduction

Accurate future blood glucose level prediction systems could play an important role in future type-I diabetes condition management practices. Such a system could prove particularly useful in avoiding hypo/hyper-glycemic events. Future blood glucose level prediction is difficult - blood glucose levels are influenced by many variables, including food consumption, physical activity, mental stress, and fatigue. The Blood Glucose Level Prediction Challenge 2020 tasked entrants with building systems to predict future blood glucose levels at 30 minutes, and 60 minutes into the future. Challenge participants were given access to the OhioT1DM dataset [8], which comprises 8 weeks worth of data collected for 12 type-I diabetes patients. The data include periodic blood glucose level readings, administered insulin information, various bio-metric data, and self-reported information regarding meals and exercise.

In the previous iteration of the challenge, several researchers demonstrated both that it is possible to predict future blood glucose levels using previous blood glucose levels only [9], and that past blood glucose levels are the most important features for future blood glucose level prediction [10]. In this study, we aimed to extend this research into future glucose level prediction from historical glucose levels only. Most previous work involved training personalized models designed to predict future blood glucose level data for a single patient [9, 10, 2]. Others used schemes coupling pre-training using adjacent patient's data with a final training phase using the patient of interest's data only [3, 12].

In this work, we investigate the possibility of building a single model that is able to predict future blood glucose levels for all 12 patients in the OhioT1DM data set, and the effectiveness of applying such a model to completely unseen data (i.e. blood glucose series from an unseen patient). We also investigate the impact of history length on future blood glucose level prediction. We experiment with various model types: linear models, feed-forward neural networks, and recurrent neural networks. Furthermore, inspired by advances in leveraging long distance temporal patterns for time series prediction [6], we attempt to build a long short-term memory (LSTM) model that is able to use information from very far in the past (up to 24 hours) by incorporating an attention mechanism. Finally, we compare the effectiveness of two methods for handling missing data during training.

2 Method

2.1 Datasets

As stated above, one of our primary aims was to build a single model that is able to predict future blood glucose levels for each patient in the data set. To this end, we constructed a combined data set containing data provided by each of the patients. Specifically, we created a data set composed of all of the data points contributed by the six patients included in the previous iteration of the challenge (both training and test sets), as well as the training data points provided by the new cohort of patients. This combined data set was split into training and validation sets: the final 20% of the data points provided by each patient were chosen for the validation set. The test data sets for the 6 new patients were ignored during development to avoid bias in the result. For experiments in building patient specific models, training and validation sets were constructed using the patient in question's data only (again with an 80/20 split).

2.2 Data preprocessing

Prior to model training, the data were standardized according to:

$$x = \frac{x - \mu_{train}}{\sigma_{train}} \tag{1}$$

[1] University of Liverpool, UK, email: robert.e.bevan@gmail.com
[2] University of Liverpool, UK, email: coenen@liverpool.ac.uk

Model	Hyper-parameters
Feed-forward	# hidden units $\in \{16, 32, 64, 128, 256, 512, 1024\}$
	# layers* $\in \{1, 2\}$
	activation function $\in \{ReLU\}$
Recurrent	# hidden units $\in \{16, 32, 64, 128, 256, 512, 1024\}$
	recurrent cell* $\in \{LSTM, GRU\}$
	# layers* $\in \{1, 2\}$
	output dropout* $\in \{0, 0.1, 0.2, 0.5\}$
LSTM + Attention	# $\boldsymbol{\alpha_t}$ hidden units $\in \{4, 8, 16, 32, 64, 128\}$

Table 1. Lists of hyper-parameters tuned when training feed-forward, recurrent neural networks, and LSTM with attention networks. Note that not all possible combinations were tried - parameters marked with an asterisk were tuned after the optimal number of hidden units was chosen.

	RMSE		MAE	
Patient ID	**PH=30**	**PH=60**	**PH=30**	**PH=60**
540	21.03 (0.07)	37.37 (0.09)	16.64 (0.1)	30.8 (0.13)
544	16.14 (0.12)	28.4 (0.14)	12.85 (0.11)	23.57 (0.16)
552	15.82 (0.06)	27.6 (0.15)	12.43 (0.12)	22.78 (0.16)
567	20.29 (0.08)	34.28 (0.18)	15.9 (0.12)	28.95 (0.13)
584	20.39 (0.07)	32.97 (0.09)	15.99 (0.03)	27.04 (0.07)
596	15.7 (0.03)	25.99 (0.12)	12.4 (0.04)	21.33 (0.13)
AVG	18.23 (2.36)	31.1 (4.05)	14.37 (1.83)	25.75 (3.43)

Table 2. Root mean square error and mean absolute error (mg/dl) computed using the test points for each patient, at different prediction horizons (30 minutes, and 60 minutes) for a single layer LSTM with 128 hidden units.

where μ_{train}, and σ_{train}, are the mean and standard deviation of the training data, respectively. There is a non-negligible amount of data missing from the training set, which needed to be considered when preprocessing the data. We investigated two approaches to handling missing data: discarding any training sequences with one or more missing data points; and replacing missing values with zeros following standardization. It was hypothesized that the second approach may help the system learn to be robust to missing data.

2.3 Model training

We experimented with linear models, feed-forward neural networks, and recurrent neural networks. Each model was trained to minimize the root mean square error (RMSE):

$$RMSE = \sqrt{\sum_{i=1}^{n} \left(\frac{\hat{y_i} - y_i}{n}\right)^2} \tag{2}$$

where $\hat{y_i}$ is the predicted value, y_i is the true value, and n is the total number of points in the evaluation set. Various hyper-parameters were tuned when training the feed-forward and recurrent neural networks; Table 1 provides a summary. During development, each model was trained for 50 epochs using the Adam optimizer (α=0.001, β_1=0.9, β_2=0.999) [5] and a batch size of 256. The final model was trained with early stopping using the same optimizer settings, and a batch size of 32, for a maximum period of 500 epochs with early stopping and a patience value of 30. Each model was trained 5 times in order to get an estimate of the influence of the random initialization and stochastic training process on the result.

Model selection and hyper-parameter tuning were performed for the 30 minute prediction horizon task. The best performing model was then trained for 60 minute prediction. Experiments were repeated for blood glucose history lengths of 30 minutes, 1 hour, 2 hours, and 24 hours.

2.4 Improving long distance pattern learning with Attention

It can be difficult for recurrent neural networks to learn long distance patterns. The LSTM network was introduced to address this problem [4]. Even so, LSTM networks can struggle to learn very long range patterns. Attention mechanisms - initially introduced in the context of neural machine translation - have been shown to improve LSTM networks' capacity for learning very long range patterns [7, 1]. Attention mechanisms have also been applied to time series data, and have proven to be effective in instances where the data exhibit long range periodicity - for example, in electricity consumption prediction [6]. We hypothesised that blood glucose level prediction using a very long history, coupled with an attention mechanism, could lead to improved performance, due to periodic human behaviours (e.g. eating meals at similar times each day; walking to and from work e.t.c). In order to test this hypothesis, we chose the best performing LSTM configuration trained with a history length of 24 hours, without attention, and added an attention mechanism as per [7]:

$$score(\boldsymbol{h_t}, \boldsymbol{h_i}) = \boldsymbol{h_t^T W h_i} \tag{3}$$

$$\alpha_{ti} = \frac{exp(score(\boldsymbol{h_t}, \boldsymbol{h_i})}{\Sigma_{j=1}^{t} exp(score(\boldsymbol{h_t}, \boldsymbol{h_j}))}) \tag{4}$$

$$\boldsymbol{c_t} = \sum_i \alpha_{ti} \boldsymbol{h_i} \tag{5}$$

$$\boldsymbol{a_t} = f(\boldsymbol{c_t}, \boldsymbol{h_t}) = tanh(\boldsymbol{W_c}[\boldsymbol{c_t}; \boldsymbol{h_t}]) \tag{6}$$

where $score(\boldsymbol{h_t}, \boldsymbol{h_i})$ is an alignment model, $\boldsymbol{c_t}$ is the weighted context vector, and $\boldsymbol{a_t}$ is the attention vector, which is fed into the classification layer (without an attention mechanism, the hidden state $\boldsymbol{h_t}$ is fed into the classification layer). The dimensionality of $\boldsymbol{\alpha_t}$ was chosen using the validation set - see Table 1 for details. We also experimented with attention mechanisms in LSTM networks designed to process shorter sequence lengths: we chose the optimal LSTM model architecture for each history length (30 minutes, 60 minutes, 2 hours), added an attention mechanism, and re-trained the model (again, the optimal $\boldsymbol{\alpha_t}$ dimensionality was chosen using the validation set).

2.5 Investigating the need for personal data during training

In order to investigate the need for an individual's data when training a model to predict their future blood glucose levels, we trained 6 different models - each with one patient's training data excluded from

Patient ID	Patient only	All patients	Patient excluded
540	21.68 (0.04)	**21.03 (0.07)**	21.16 (0.11)
544	17.28 (0.1)	**16.14 (0.12)**	16.22 (0.09)
552	16.87 (0.12)	**15.82 (0.06)**	15.87 (0.09)
567	21.15 (0.3)	**20.29 (0.08)**	20.5 (0.11)
584	22.11 (0.13)	**20.39 (0.07)**	20.46 (0.06)
596	16.16 (0.11)	**15.7 (0.03)**	15.71 (0.02)
AVG	19.21 (2.48)	**18.23 (2.36)**	18.32 (2.4)

Table 3. Root mean square error (mg/dl) computed using the test points for each patient, with a prediction horizon of 30 minutes for a single layer LSTM with 128 hidden units, trained using different data sets. Values listed in the first column correspond to models trained using the individual patient data only; values in the middle column correspond to models trained using data from all patients; values in the final column correspond to models trained using data from all patients except for the patient for which the evaluation is performed.

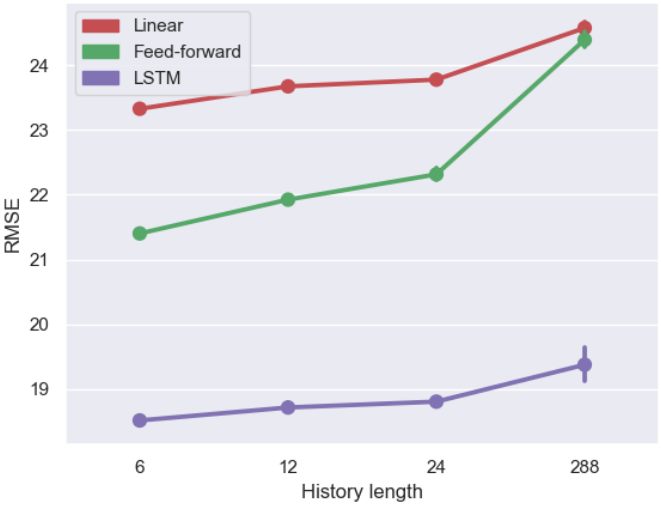


Figure 1. Comparison of validation set scores for linear, feed-forward, and LSTM neural networks as a function of history length.

the training set - using the optimal LSTM architecture determined in previous experiments. The models were then evaluated using the test data for the patient that was excluded from the training set. We also trained 6 patient specific models, each trained using the patient's training data only. We again used the optimal architecture determined in previous experiments, but tuned the number of hidden units using the validation set in order to avoid over-fitting due to the significantly reduced size of the training set (compared with the set with which the optimal architecture was chosen). Each model was trained using the early-stopping procedure outlined in 2.3.

3 Results and Discussion

Our evaluation showed that recurrent models performed significantly better than both linear and feed-forward neural network models, for each history length we experimented with (p=0.05, corrected paired t-test [11]). We also found that feed-forward networks generally outperformed linear models, likely due to their ability to model non-linear relationships. The optimal feed-forward network contained 512 hidden units. We found no difference between LSTM and gated recurrent unit (GRU) networks - remaining evaluations will be performed for LSTM recurrent networks only for simplicity. Figure 1 compares the performance of the different model types as a function of history length. For each model class we observed that performance decreased linearly with increasing history length. The LSTM appeared better able to deal with longer history lengths - the performance degradation was less severe than for the other model classes. We found a history length of 30 minutes to be optimal for each model class. The best performing LSTM model contained a single layer with 128 hidden units, and was trained without dropout. The test set results for this model are listed in Table 2.

Table 3 compares LSTM models (with the same architecture as above) trained with the following data sources: the individual patient's data only, data from all of the available patients, and data from every other patient (excluding data contributed by the patient in question). We observed that models trained using a large amount of data, but excluding the patient's data, outperformed models trained using the patient's data only (p=0.05). We also found no significant difference in performance between models trained using all of the available training data (i.e. including the patient's data) and those trained excluding the patient's data, highlighting the general nature of the models. We found that including sequences with values in the training set produced models that were more robust to missing data, as evidenced by the improved RMSE scores listed in Table 4: RMSE scores were significantly improved for each patient using this approach to training (p=0.05).

Incorporating an attention mechanism further improved performance in most instances: we observed significant improvements for history lengths of 30 minutes, 60 minutes, and 2 hours (p=0.05), but not for history lengths of 24 hours. Figure 3 compares the regular LSTM and LSTM with attention performance as a function of history length. Figure 2 shows partial auto-correlation plots for 4 different patients. Interestingly, two of the patients' blood glucose data - patient 540, and patient 544 - don't show any significant long term correlation, whereas the other two - patient 552, and patient 567 - both exhibit significant correlation at time lags of approximately 6 and 12 hours. We observed this behaviour in half of the patients. We

Patient ID	Exclude missing data	Include missing data
540	21.45 (0.06)	**21.03 (0.07)**
544	16.79 (0.06)	**16.14 (0.12)**
552	16.27 (0.13)	**15.82 (0.06)**
567	21.19 (0.1)	**20.29 (0.08)**
584	21.16 (0.06)	**20.39 (0.07)**
596	16.08 (0.07)	**15.7 (0.03)**
AVG	18.82 (2.45)	**18.23 (2.36)**

Table 4. Root mean square error (mg/dl) computed using the test points for each patient, with a prediction horizon of 30 minutes for a single layer LSTM with 128 hidden units, trained using different methods of handling missing data. Values in the first column correspond to a model trained with full sequences only (any sequences with missing values were discarded). Values in the second column correspond to a model trained with sequences including missing values - missing values were replaced with zeros following standardization.

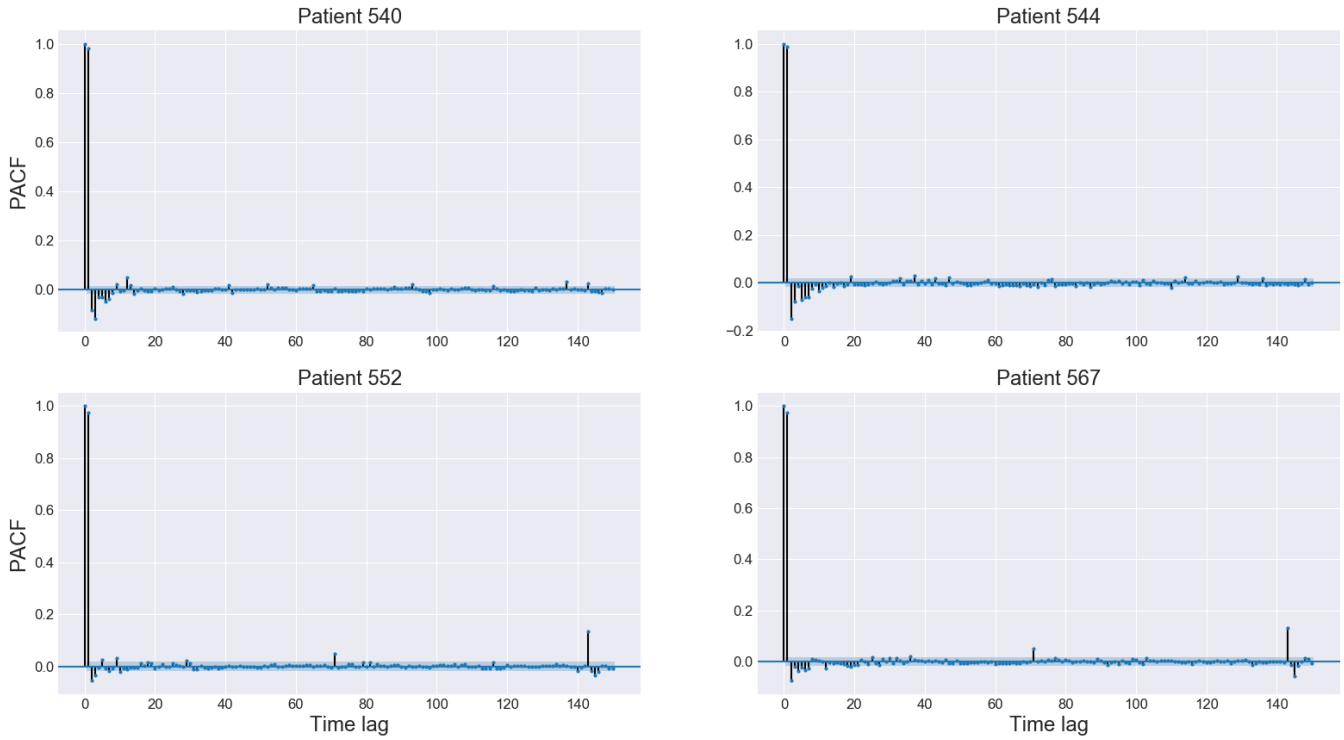


Figure 2. Partial auto-correlation plots for 4 different patients. The patients in the top row exhibit short term patterns only, but those in the bottom row show significant correlations at time lags of approximately 6 and 12 hours.

also observed correlations at even greater time lags, corresponding to multiples of 6 hours. The difference in the patients' partial auto-correlation plots suggests it may be sub-optimal to train an attention mechanism using each patient's data at once, and that training at the patient-level may enable the model to learn very long range patterns. Furthermore, while we were able to train an LSTM model with a history length of 30 minutes that generalized across all patients, it may be the case that short range blood glucose patterns are quite general, and long range patterns are more personalized, and tuning the history length per patient could improve prediction performance. All of the results presented in this section can be reproduced using publicly available code [3].

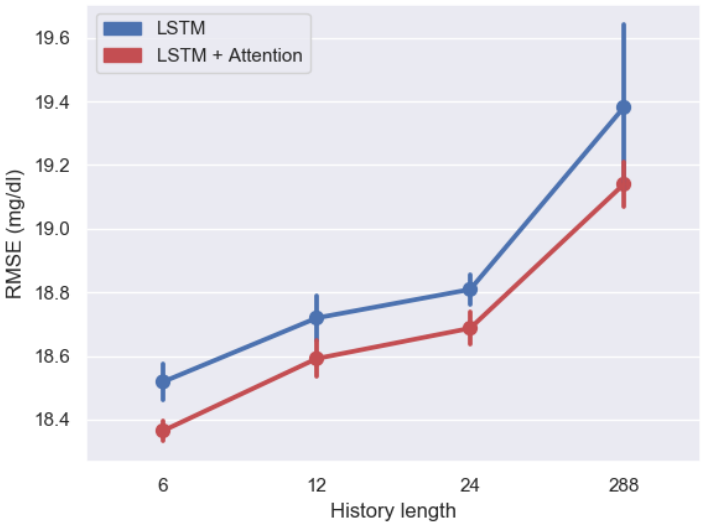


Figure 3. Comparison of validation set RMSE scores (prediction horizon = 30 minutes) for LSTMs and LSTMs incorporating an attention mechanism as a function of history length.

4 Conclusion

In this study we showed that it is possible to train a single LSTM model that is able to predict future blood glucose levels for each of the different patients whose data are included in the OhioT1DM data set. We also demonstrated that an individual patient's data is not required during the training process in order for our model to effectively predict the patient's future blood glucose levels. Furthermore we showed that incorporating an attention mechanism in the LSTM improved performance, and that including sequences with missing values during training produced models that were more robust to missing data.

[3] https://github.com/robert-bevan/bglp

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio, 'Neural machine translation by jointly learning to align and translate', in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, (2015).

[2] Arthur Bertachi, Lyvia Biagi, Iván Contreras, Ningsu Luo, and Josep Vehí, 'Prediction of blood glucose levels and nocturnal hypoglycemia using physiological models and artificial neural networks', in *Proceedings of the 3rd International Workshop on Knowledge Discovery in Healthcare Data*, pp. 85–90.

[3] Jianwei Chen, Kezhi Li, Pau Herrero, Taiyu Zhu, and Pantelis Georgiou, 'Dilated recurrent neural network for short-time prediction of glucose concentration', in *Proceedings of the 3rd International Workshop on Knowledge Discovery in Healthcare Data*, pp. 69–73.

[4] Sepp Hochreiter and Jürgen Schmidhuber, 'Long short-term memory', *Neural computation*, **9**(8), 1735–1780, (1997).

[5] Diederik P. Kingma and Jimmy Ba, 'Adam: A method for stochastic optimization', in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, (2015).

[6] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu, 'Modeling long- and short-term temporal patterns with deep neural networks', *CoRR*, **abs/1703.07015**, (2017).

[7] Thang Luong, Hieu Pham, and Christopher D. Manning, 'Effective approaches to attention-based neural machine translation', in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pp. 1412–1421, (2015).

[8] Cynthia R. Marling and Razvan C. Bunescu, 'The OhioT1DM dataset for blood glucose level prediction', in *Proceedings of the 3rd International Workshop on Knowledge Discovery in Healthcare Data*, pp. 60–63, (2018).

[9] John Martinsson, Alexander Schliep, Bjorn Eliasson, Christian Meijner, Simon Persson, and Olof Mogren, 'Automatic blood glucose prediction with confidence using recurrent neural networks', in *Proceedings of the 3rd International Workshop on Knowledge Discovery in Healthcare Data*, pp. 64–68.

[10] Cooper Midroni, Peter J. Leimbigler, Gaurav Baruah, Maheedhar Kolla, Alfred J. Whitehead, and Yan Fossat, 'Predicting glycemia in type 1 diabetes patients: Experiments with xgboost', in *Proceedings of the 3rd International Workshop on Knowledge Discovery in Healthcare Data*, pp. 79–84.

[11] C. Nadeau and Y. Bengio, 'Inference for the generalization error', *Mach. Learn.*, **52**(3), 239–281, (September 2003).

[12] Taiyu Zhu, Kezhi Li, Pau Herrero, Jianwei Chen, and Pantelis Georgiou, 'A deep learning algorithm for personalized blood glucose prediction', in *Proceedings of the 3rd International Workshop on Knowledge Discovery in Healthcare Data*, pp. 74–78.

Deep Residual Time-Series Forecasting: Application to Blood Glucose Prediction

Harry Rubin-Falcone[1], **Ian Fox**[1] , **and Jenna Wiens**[1]

Abstract. Improved forecasting of blood glucose could aid in the management of diabetes. Recently proposed neural network architectures that use stacked fully connected layers with residual backcasting have achieved state-of-the-art performance on benchmark time-series forecasting tasks. Though promising, previous work ignores opportunities for additional supervision, and the use of fully connected layers fails to account for the temporal nature of the signal. Here, we propose a new architecture that builds on previous work by learning to forecast gradually in stages or blocks. Our updates include replacing the fully connected block structure with a recurrent neural network and adding additional losses to provide auxiliary supervision. In addition, we leverage important context in the form of additional input signals. Applied to the task of glucose forecasting, we find that each of these modifications offers an improvement in performance. On the task of predicting blood glucose values 30 minutes into the future, our proposed approach achieves a mean rMSE of 18.2 mg/dL, versus 21.2 mg/dL achieved by the baseline. These improvements get us closer to predictions that are reliable enough to be used in CGMs or insulin pumps to manage diabetes.

1 Introduction

Accurate blood glucose forecasting would improve diabetes treatment by enabling proactive treatment [11]. To this end, there has been significant interest in developing time-series forecasting methods for predicting blood glucose levels, using a large variety of statistical and machine learning methods [9]. This has inspired the OhioT1DM challenge [6], where participants are tasked with accurately forecasting blood glucose values in individuals with type 1 diabetes. As our entry to this competition, we build on recent work in time-series forecasting that focuses on iterative residual prediction, specifically Neural Basis Expansion for Interpretable Time-Series Forecasting, or N-BEATS [8]. This work uses a neural network architecture consisting of network *blocks* that output both a *forecast* and a *backcast* (*i.e.*, a reconstruction of the block's input). The backcast is subtracted from the block's input forming a *residual* which then serves as input to the following block. At the final layer, the forecasts from each block are combined to form the final prediction. The iterative and residual nature of this architecture aims to encourage gradual signal reconstruction and forecasting. We build upon this idea in several ways (**Figure 1**):

- We replace the fully connected architecture in each block with a recurrent neural network (RNN) [12]. We hypothesize that this will allow the model to more accurately capture important temporal relationships within the input/output.

[1] University of Michigan, USA, email: hrf@umich.edu

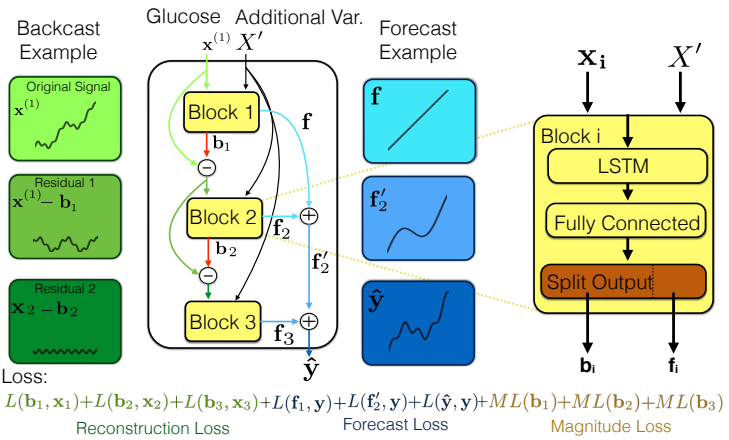


Figure 1. Our glucose forecasting architecture, shown with three blocks. Each residual block contains a bidirectional LSTM with a single output layer that produces the forecast and backcast simultaneously. Additional variables are added as input channels to each block, but residuals are calculated only for the glucose signal. Losses (described in section 2.2) are calculated after each block. Here, *L* is the primary loss function (MSE) and *ML* is magnitude loss, which is a penalization for small signals (the inverse of the norm).

- We include additional variables as input to the model (*e.g.*, bolus insulin data), which we hypothesize will provide valuable context. However, due to the sparse nature of some of these variables, we backcast only on our primary variable of interest: blood glucose.
- We include additional loss terms that act as auxiliary supervision. We hypothesize that these terms will further encourage the model's blocks to gradually learn accurate components of the signal (*i.e.*, backcasts and forecasts) that sum to the correct signal.

Additionally, we study the effects of pre-training with related datasets. Applied to the task of glucose forecasting, these modifications lead to notable improvements over baseline.

2 Methods

Given the strong performance of deep residual forecasting across a range of tasks [8], we build on recent work in this area, tailoring the approach to the task of forecasting blood glucose.

2.1 Problem Setting and Notation

We focus on the task of univariate time-series forecasting in which we aim to predict future values of a single variable, but assume

we have access to additional inputs. Let $X = [\mathbf{x}^{(1)}, ..., \mathbf{x}^{(d)}]$ represent a multivariate time series where d is the number of variables. For each variable i: $\mathbf{x}^{(i)} \in \mathbb{R}^T$ is a sequence of length T. For our problem, $\mathbf{x}^{(1)}$ corresponds to glucose measurements and $X' = [\mathbf{x}^{(2)}, ..., \mathbf{x}^{(d)}]$ represents other variables of interest (*e.g.*, bolus insulin). Given X, we aim to predict the next h glucose measurements $\mathbf{y} = x^{(1)}_{T+1}, x^{(1)}_{T+2}, ..., x^{(1)}_{T+h}$.

2.2 Proposed Approach

We build on a recent univariate time-series forecasting architecture, N-BEATS [8], which consists of a series of n blocks, each composed of a series of fully-connected layers. The i^{th} block takes as input some vector $\mathbf{x_i} \in \mathbb{R}^{\mathbf{T}}$, where for the first block, $\mathbf{x_1}$ is the original input. Each block's output $\Phi(\mathbf{x_i}) \in \mathbb{R}^{h'}$ produces a forecast $\mathbf{f_i} \in \mathbb{R}^h$ and a backcast $\mathbf{b_i} \in \mathbb{R}^T$. The backcast is subtracted from the current block's input before being input to the next block (*i.e.*, $\mathbf{x_{i+1}} = \mathbf{x_i} - \mathbf{b_i}$). The output of the network is the sum of forecasts across all blocks: $\hat{\mathbf{y}} = \sum_{i=1}^{n} \mathbf{f_i}$. The residual nature of the prediction means that each block learns only what the previous block could not. Thus, the model learns to gradually reconstruct the signal, while predicting components of the forecast. We build on this idea by identifying several opportunities for improvement. Our modification are as follows:

Accounting for Temporal Structure. We account for the temporal nature of the input sequence by using a bidirectional LSTM network [3] within each block, in lieu of the fully connected layers. Although using fully connected networks results in a faster training time, we hypothesized that an LSTM would be better able to capture block-level time-varying patterns by enforcing a sequential prior on the signal. The final LSTM hidden state is used as input to an output layer that produces a sequence of length $T + h$ that is then split into the backcast and forecast signal.

Including Additional Input Variables. We include auxiliary variables as input at each block. This provides important context when backcasting and forecasting the main signal of interest. However, rather than simply augmenting the dimensionality of the signal throughout, we backcast/forecast only the primary signal, *i.e.*, each block's input is $[\mathbf{x}^{(1)}_{i-1} - \mathbf{b}_{i-1}, X']$ (**Figure 1**). Backcasting/forecasting was not performed on additional variables due to their sparse nature. We hypothesized that learning to backcast/forecast abrupt carbohydrate and bolus inputs would increase the difficulty of the overall task and decrease performance on the main task.

Auxiliary Tasks and Supervision. In addition to training based on the loss of the final forecast prediction, $MSE(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{h}\sum_{i=1}^{h}(y_i - \hat{y}_i)^2$, we include additional losses with the goal of improving the quality of intermediate representations. We add three auxiliary tasks to our model (shown in **Figure 1**):

- Per-block reconstruction loss ($\mathcal{L}_r$): We hypothesized that explicitly supervising the backcasts could improve their accuracy, and supervising at the block level could improve the intermediate reconstructions. To do this, we add an MSE loss, $MSE(\mathbf{x}^{(1)}_i, \mathbf{b}_i)$ after each block's output. Each block's loss is weighted proportional to its position in the network, encouraging the network to gradually construct a backcast. The final loss $\mathcal{L}_r$ is scaled by the sum of the weights of each block, and summed over n blocks:

$$\mathcal{L}_r = \frac{\sum_{i=1}^{n} i \cdot MSE(\mathbf{x}^{(1)}_i, \mathbf{b}_i)}{\sum_{i=1}^{n} i}$$

- Per-block forecast loss ($\mathcal{L}_f$): To encourage each block to contribute towards an accurate forecast, we calculate the loss on the running forecast after each block. We apply a similar per-block scaling term as in the backcast, but increase the weight using cubic values in order to emphasize the network's later predictions. Let $\mathbf{f'_i}$ denote the sum of all forecasts up to the i^{th} block, ie $\mathbf{f'_i} = \sum_{k=1}^{i} \mathbf{f_k}$. The total forecast loss L_f is:

$$\mathcal{L}_f = \frac{\sum_{i=1}^{n} i^3 \cdot MSE(\mathbf{f'_i}, \mathbf{y})}{\sum_{i=1}^{n} i^3}$$

- Per-block magnitude loss ($\mathcal{L}_m$): Intuitively, deep residual forecasting aims to incrementally learn parts of a signal. To ensure that each block contributes to the backcast, we penalize blocks by the inverse of their output size (measured using an L_1 norm) and scale this penalty by the inverse of their position in the stack, so earlier blocks are encouraged to output larger values. We apply this loss to only the backcast. We do not penalize low-magnitude forecasts because it is possible that a block could account for part of the input signal that is associated with a zero-value forecast. The complete magnitude loss is:

$$\mathcal{L}_m = \frac{\sum_{i=1}^{n} \frac{1}{i} \cdot \frac{1}{|\mathbf{b_i}|}}{\sum_{i=1}^{n} \frac{1}{i}}$$

Loss is calculated as a weighted sum of these three losses:

$$\mathcal{L} = \mathcal{L}_f + \beta\mathcal{L}_r + \gamma\mathcal{L}_m,$$

where $\beta, \gamma \in \mathbb{R}^+$are hyperparameters. Note that the final forecast is included as a part of $\mathcal{L}_f$.

2.3 Ensembling

We ensemble models that use different input lengths. As others have shown, this allows us to account for trends at different time scales [8]. We train a model for each of 6 different input lengths. We use T = 12, 18, 24, 30, 36, and 42 time steps. These values were selected as multiples of the shortest input length, $h = 6$. Shorter backcast windows (*i.e.*, $T < 12$) were not found to be beneficial. When ensembling the predictions, we output the median.

3 Experimental Set Up

Datasets. We evaluate the proposed approach using the 2020 OhioT1DM dataset [6]. This dataset consists of CGM, insulin pump, and other variables (*i.e.*, sleep data, activity levels) for six individuals. We explored using all variables in the dataset, but found only glucose (CGM and finger stick), bolus dose, carbohydrate input, and time of day to be helpful. Each individual has approximately 10,000 samples for training and 2,500 for testing, recorded at 5-minute intervals, where the test data temporally follow the training data. We use the first 80% of each individual's specified training set for training and the remaining 20% for early-stopping validation, holding out the test data. Beyond the 2020 OhioT1DM dataset, we had access to Tidepool data, a large repository of CGM and insulin pump data for approximately 100 participants with a total 15

million time steps [7]. We used these data and data from the 2018 OhioT1DM challenge [6] during pre-training and model selection.

Preprocessing. Bolus dose and carbohydrate input are less frequently recorded compared to CGM. Thus, to be used as input to the model, we align and resample these additional data to length T. In addition, we encode time using sine and cosine embeddings over 24-hour periods. Approximately 15% of all time-steps were missing for CGM. We replace missing values with a value of zero and include an additional variable that indicates missingness. In this way, the model can learn to handle missing values differently [5]. This results in a model input of $X \in \mathbb{R}^{T \times 7}$ (CGM, finger stick glucose, bolus values, carbohydrate inputs, sine and cosine of time, and missingness indicators for CGM values). Time steps with missing glucose values are ignored during all loss calculations and forecast evaluations.

Baseline Architecture Implementation. Our baseline is based on N-BEATS and uses hyperparameters similar to those reported by Oreshkin *et al.*: 10 blocks with 4 layers each, with 512 hidden units output by each layer, with a ReLU activation function between each layer [8]. Though Oreshkin *et al.* originally used 30 blocks, we found 10 blocks worked better and thus report performance against this stronger baseline.

Proposed Architecture Implementation. To learn the model parameters of our proposed approach, we use Adam [4] with a learning rate of 0.0002 (selected to maximize learning speed while maintaining reasonable convergence behavior), and a batchsize of 512. We run the optimization algorithm for up to 300 epochs, or until validation performance does not improve for 20 iterations.

To reduce the chance of overfitting to the 2020 data, we tune all hyperparameters, excluding γ and β, on the six subjects from the previous competition. We performed a grid search to select the number of blocks (range: 5-10, best: 7) and number of hidden LSTM units (range: 50-300, best: 300), optimizing for rMSE of the final time step of the prediction on the held out data. Including more hidden units could perhaps have further improved performance, but we were limited by memory constraints.

During tuning, we measured the added value of including each auxiliary variable (*e.g.*, sleep data), including only those variables that improved performance for the final evaluation. We tuned the block-weighting terms for our auxiliary loss functions, considering inverse, constant, and linear relationships for $\mathcal{L}_r$, and powers of 2 and 3 for $\mathcal{L}_f$. We tuned the auxiliary loss function weights β, and γ using the validation data for the new subjects. We performed a grid search over $[.1, .15, .25, .3, .5, 1, 2, 4]$ for β and $10e5$ times those values for γ, selecting $\beta = .3$ and $\gamma = 10e4$. Grid search analyses were tuned with a subset of backcast lengths ($3f$ and $6f$) across all patients, for efficiency.

For both the 30 and 60 minute prediction horizons, we train using loss for the full window (*i.e.*, $h = 6$ for the 30 minute analyses and $h = 12$ for the hour prediction), but report results using the last time step only (as dictated by the competition).

Pre-training. As mentioned above, we pre-train on the 2018 OhioT1DM and Tidepool data. Pooling these datasets together, we train a single model. The weights of this model are used to initialize the weights of a 'global' model trained on the 2020 OhioT1DM training set. This 'global' model is then fine-tuned to each participant from the 2020 OhioT1DM dataset, resulting in six participant-specific models. Baseline analyses do not use the large pooled dataset for pre-training, but start from a single global model (trained from random initialization) and then fine-tune.

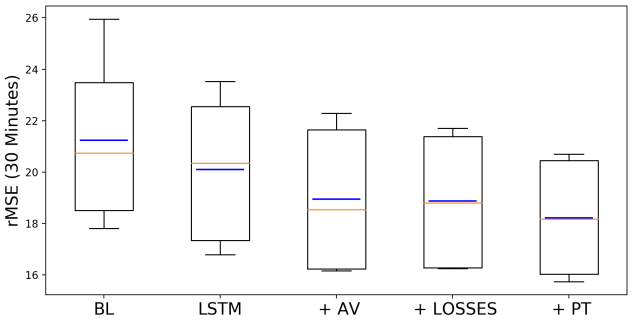


Figure 2. Cumulative effect of each proposed modification on 30-minute horizon forecast performance. Each modification includes the modifications to the left as well. BL = baseline, LSTM = updated block structure, +AV = added variables, +LOSSES = additional task losses added, +PT = used large pooled dataset for pre-training. Each box plot shows the distribution across the six subjects, with blue and orange lines indicating the mean and median across subjects, respectively.

Code. All of our experiments were implemented in Python/PyTorch [10]. The final initialization models and all code used in our analyses is publicly available [2].

3.1 Evaluation

We report results for models trained to predict 12 time steps (one hour horizon) and six time steps (30 minute horizon), reporting outcomes on only the final time step. We evaluate using the square root of the mean squared error (rMSE), and mean absolute error (MAE), defined as $MAE(\mathbf{y}, \mathbf{y'}) = \frac{|\mathbf{y}-\mathbf{y}'|}{n}$ for n samples. Metrics are applied directly to the raw CGM values with no preprocessing.

To examine the effect of each architecture modification, we perform an ablation study where we add each modification to the architecture incrementally (*i.e.*, first RNN blocks only, then RNN blocks with added variables, then RNN blocks with added variables and additional loss terms, then the full architecture with pre-training). We also experiment with each modification made in isolation.

Finally, beyond the evaluation dictated by the competition, we also calculate outcomes for the most crucial forecasting windows: those that represent the beginning of a hyper- or hypoglycemic event. These correspond to predictions for which the most recent time step is in the euglycemic range (blood glucose from 70-180 mg/dL), but the participant becomes hypoglycemic (< 70 mg/dL) or hyperglycemic (> 180 mg/dL) within the prediction horizon. We also examine the distribution of predictions vs. actual measurements in terms of the corresponding would-be treatment recommendations with a Clarke Error Grid Analysis [2].

4 Results and Discussion

Overall, the proposed approach leads to average rMSEs of 18.2 and 31.7 on the 30 and 60 minute horizons, respectively, and average

[2] https://gitlab.eecs.umich.edu/mld3/deep-residual-time-series-forecasting

MAEs of 12.8 and 23.6 for those horizons (**Table 1**). These values are comparable to the results achieved in the 2018 challenge (*i.e.*, 18.9 to 21.7 for 30 minute rMSE) [1, 13].

Based on our ablation study, each of our modifications improves performance over baseline, although to varying degrees **(Figure 2**). For the 30 minute horizon, mean rMSE across participants is 21.2 for the baseline model (BL). Adding the RNN block structure (bidirectional LSTM) and additional variables (+AV) improves performance to 20.1 and then 18.94, respectively. Adding the additional loss functions (+LOSSES) results in a slightly improved mean performance of 18.87, and pre-training on the Tidepool and 2018 OhioT1DM datasets (+PT) further improves performance to 18.2. We observed similar trends for the 60 minute horizon, and for MAE.

Table 1. Participant-level results for 30 and 60 minute horizons on the held-out test data from the 2020 competition.

	Prediction Horizon			
	30 Minutes		60 Minutes	
Participant ID	rMSE	MAE	rMSE	MAE
567	20.70	13.78	35.99	25.84
544	15.97	11.20	26.01	19.01
552	15.74	11.51	29.17	22.78
596	16.24	11.26	27.11	19.73
540	20.10	14.77	38.43	29.51
584	20.57	14.47	33.26	24.72
Mean	18.22	12.83	31.66	23.60

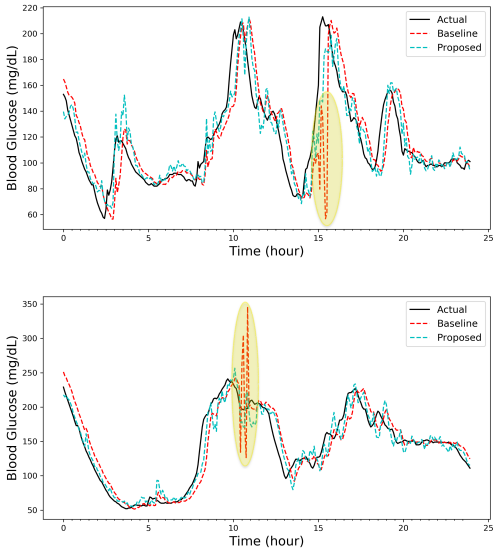


Figure 3. A full day of 30-minute predictions on two participants, comparing the baseline and our proposed approach. Qualitatively, our modifications result in fewer sudden extreme failures. In general, our predictions appear more accurate, compared to the baseline. (a) Best performance individual - 552 (rMSE = 15.7). (b) Worst performance individual - 567 (rMSE = 20.7).

While all of our modifications reduce rMSE, one of the largest improvements in performance comes from replacing the fully connected layers of N-BEATS with recurrent layers (improves performance from 21.2 to 20.1). These recurrent layers directly model each time step as a function of the previous, allowing for more accurate temporal representations. Compared to the baseline, the proposed approach leads to more faithful predictions and fewer extreme failures **(Figure 3**). Even without pre-training, the modified architecture eliminates the extreme failures observed in the baseline case (highlighted in **Figure 3**).

Additional variables are significantly more impactful when added to the RNN-based block structure (a 30 minute rMSE improvement from 20.1 to 18.9), when compared to adding them in isolation (an improvement from 21.2 to 21.0). Despite the poor performance when used in the original baseline architecture, we chose to include these variables in our updated RNN-based model. The RNN block structure has multiple input channels, which allows for the direct use of the variables. In contrast, the fully connected layers require a flat input, which removes much of the timing information. This illustrates the importance of the ordering of decision making in model architecture development. Although such decisions are easy when an exhaustive search of combinations is performed, this is not always feasible, so these decisions often must be made sequentially. If we had selected additional variables based on the performance of the baseline model alone, we may not have opted to use them, and would have lost the large benefit that they provide when used in tandem with the RNN block structure.

To test our hypothesis regarding the efficacy of backcasting on blood glucose values only (and not additional variables), we ran a post-hoc analysis with a model using residual backcasting for all input variables. As expected, we found that this decreased performance (30 minute rMSE = 19.2 vs. 18.9).

In contrast to the other modifications, the additional loss terms offer only slight improvements over the baseline. During model selection, this modification offered substantially more improvement on the OhioT1DM 2018 dataset (decreasing rMSE of the RNN + added variables model from 19.6 to 18.9). Yet even with additional tuning, this modification is only minimally helpful on the 2020 dataset, suggesting that the method may not be universally beneficial. In an ablation analysis, a model using only ($\mathcal{L}_f$) performs best (rMSE = 18.82) on the 2020 data. However, in contrast, the added variables seem to help more for the 2020 data than for the 2018 data. This suggests that perhaps the losses serve a regularizing effect when insufficient information is present in the additional variables, as we observed in the 2018 validation data.

Pre-training improves the performance of both the modified and baseline architectures, although it offers the most improvement for the baseline (improving performance from 21.2 to 19.8 when applied alone). This suggests that pre-training becomes less crucial once other improvements are made. Running an analysis using our proposed architecture and no pretraining over 4 random seeds results in similar performance across seeds (30-minute rMSEs were 18.87, 18.82, 18.95, and 18.89), indicating fairly robust performance over different random initializations. Further, pre-training improves performance significantly over all random initialization, demonstrating the benefit of pre-training beyond chance.

In our event-specific evaluation (includes only prediction windows where a hyperglycemic or hypoglycemic event occur), the mean rMSE is significantly higher compared to the holistic evaluation (for our proposed approach 30 minute horizon: 27.9 vs. 18.2). This is expected, since such events are difficult to anticipate. Specifically, the proposed approach achieves a higher rMSE for hyperglycemic events (30.5) than hypoglycemic (23.8). Interestingly, while our model performs better than the baseline for all events on average (our model: 27.9 vs baseline: 31.0) and for hyperglycemic events (30.5 vs 34.2), the two models perform comparably in the hypoglycemic range (23.8 vs 23.6). This could be due to the relative rarity of hypoglycemic events (for example, there were 322 in the test data, vs 868 hyper-

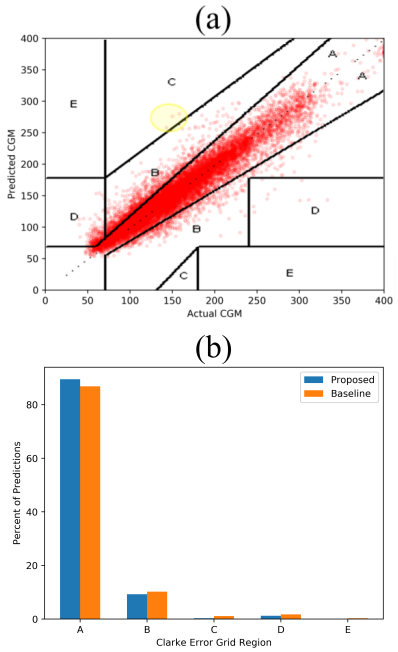


Figure 4. (a) A Clarke Error Grid depicting which treatment options would be recommended based on our proposed approach's predictions vs what treatments should be given based on the actual value. 99% of predictions fall within regions A and B, which indicate appropriate treatment recommendations. (b) Proportion of predictions in each region of the grid for our proposed approach and the baseline model.

glycemic events), or it could be a reflection of the MSE loss function over-emphasizing large values, which is unaccounted for in our approach.

In our Clarke Error Grid Analysis [2], we find that 99% of predictions fall in regions A and B (regions that would not lead to inappropriate treatment; *i.e.* an unnecessary bolus or rescue carbohydrates), with 90% in region A (predictions within 20% of the actual value) and 9% in region B (predictions that are more than 20% from the actual value but that would not lead to inappropriate treatment), indicating generally strong performance **(Figure 4)**. The proportion of points in regions A and B is slightly lower (97%) for our baseline model. Only 3 points (0.02%) fall into region C (points that would lead to unnecessary treatment, specifically predicting high CGM when it would actually be lower, which could lead to an inappropriate bolus and hypoglycemia), and 1% fall into region D (points that miss hyperglycemia or hypoglycemia). Reassuringly, no points fall into region E (regions that would treat hyperglycemia as hypoglycemia or vice versa).

5 Conclusion

We find deep residual forecasting to be effective when applied to the task of predicting blood glucose values. Augmenting a previously proposed architecture with RNNs in place of fully connected stacks, additional variables, and self-supervising loss functions all lead to improvements when applied to the task of blood glucose forecasting. Beyond blood glucose forecasting, we hypothesize that many of the proposed changes could be beneficial when applied to other forecasting tasks.

ACKNOWLEDGEMENTS

This work was supported by JDRF (award no. 1-SRA-2019-824-S-B). In particular, this award provided access to the Tidepool dataset that was used in pre-training the models. The views and conclusions in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied of JDRF.

REFERENCES

[1] J. Chen, K. Li, P. Herrero, T. Zhu, and P. Georgiou, 'Dilated recurrent neural network for short-time prediction of glucose concentration.', *KDH*, (2018).

[2] W.L. Clarke, D. Cox, L.A. Gonder-Frederick, W. Carter, and S.L. Pohl, 'Evaluating clinical accuracy of systems for self-monitoring of blood glucose.', *Diabetes Care*, **10**, (1987).

[3] F.A. Gers, F. Schmidhuber, and F. Cummins, 'Learning to forget: continual prediction with LSTM.', *9th International Conference on Artificial Neural Networks*, 850–855, (1999).

[4] D.P. Kingma and J. Ba, 'Adam: A method for stochastic optimization.', *International Conference for Learning Representations*, (2014).

[5] Z.C. Lipton, D.C. Kale, and R. Wetzel, 'Modeling missing data in clinical time series with rnns.', *Proceedings of Machine Learning for Healthcare*, (2016).

[6] C. Marling and R. Bunescu, 'The OhioT1DM dataset for blood glucose level prediction: Update 2020.', (2020).

[7] A. Neinstein, J. Wong, H. Look, B. Arbiter, K. Quirk, S. McCanne, Y. Sun, M. Blum, and S. Adi, 'A case study in open source innovation: developing the tidepool platform for interoperability in type 1 diabetes management.', *Journal of the American Medical Informatics Association*, **23**, (2016).

[8] B.N Oreshkin, D. Carpov, N. Chapados, and Y. Bengio, 'N-BEATS: Neural basis expansion analysis for interpretable time series forecasting', *ICLR*, (2020).

[9] S. Oviedo, J. Vehí, R. Calm, and J. Armengol, 'A review of personalized blood glucose prediction strategies for T1DM patients.', *Numerical Methods in Biomedical Engineering*, (2016).

[10] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, 'Automatic differentiation in PyTorch.', *NeurIPS*, (2017).

[11] J. Reifman, S. Rajaraman, A. Gribok, and W.K. Ward, 'Predictive monitoring for improved management of glucose levels.', *J Diabetes Sci Technol*, (2007).

[12] R.J. Williams and G.E. Hinton, 'Learning representations by back-propagating errors.', *Nature*, **323**, 533–536, (1986).

[13] T. Zhu, K. Li, P. Herrero, J. Chen, and P. Georgiou, 'Predictive monitoring for improved management of glucose levels.', *KDH*, (2018).

Personalised Glucose Prediction via Deep Multitask Networks

John Daniels and **Pau Herrero** and **Pantelis Georgiou**[1]

Abstract. Glucose control is an essential requirement in primary therapy for diabetes management. Digital approaches to maintaining tight glycaemic control, such as clinical decision support systems and artificial pancreas systems rely on continuous glucose monitoring devices and self-reported data, which is usually improved through glucose forecasting. In this work, we develop a multitask approach using convolutional recurrent neural networks (MTCRNN) to provide short-term forecasts using the OhioT1DM dataset which comprises 12 participants. We obtain the following results - 30 min: 19.79±0.06 mg/dL (RMSE); 13.62±0.05 mg/dL (MAE) and 60 min: 33.73±0.24 mg/dL (RMSE); 24.54±0.15 mg/dL (MAE). Multitask learning facilitates an approach that allows for learning with the data from all available subjects, thereby overcoming the common challenge of insufficient individual datasets while learning appropriate individual models for each participant.

1 INTRODUCTION

In recent years, the proliferation of biosensors and wearable devices has facilitated the ability to perform continuous monitoring of physiological signals. In diabetes management, this has come with the increasing use of continuous glucose monitoring (CGM) devices for helping with glucose control. The current literature on clinical impact of CGM devices shows that continuously monitoring blood glucose concentration levels has benefit in maintaining tight glycaemic control [5, 2]. As a next step, glucose prediction offers an opportunity to further improve glucose control by taking actions to avert adverse glycaemic events, such as suspension of insulin delivery in closed-loop systems to avert hypoglycaemia.

The general work in this area has typically involved collecting data covering physiological variables such as glucose concentration levels, heart rate, and self-reported data covering exercise,sleep, stress, illness, insulin, and meals. However, public datasets covering ambulatory monitoring of T1DM population are not widely available.

Deep learning [6] facilitates learning the optimal features and has been shown to perform better than other methods involving hand crafted features that have been employed in recent times for predicting glucose concentration levels. However, typically these models require relatively large amounts of data to converge on an appropriate model.

In this work, we employ a multitask learning [1] approach in order to improve the performance of the glucose forecasting in a neural network, where each individual is viewed as a task, using shared layers to enable learning form other individuals.

[1] Imperial College London, United Kingdom, email: jsd111@imperial.ac.uk, p.herrero-vinias@imperial.ac.uk, pantelis@imperial.ac.uk

2 RELATED WORK

Glucose prediction has been a long-standing area of focus in the diabetes community. As a result, many approaches have existed in order to provide near-time glucose concentration level forecasts.

Early work in this area have focused on physiological models and traditional machine learning methods in predicting glucose concentration levels [12, 3]. Recent work as seen in the 2018 Blood Glucose Predictive Challenge has seen a move towards deep learning methods with more impressive results [11, 9, 14, 8]. These have used convolutional architectures, recurrent architectures, or a combination of both to model the task of glucose prediction.

3 DATASET AND DATA PREPROCESSING

In this section, we detail the transformations that are performed on the data prior to training and testing the model for each T1DM participant.

3.1 OhioT1DM Dataset 2020

The OhioT1DM dataset 2020 [10] is a dataset comprising 12 unique participants that cover eight weeks of daily living. The participants are given IDs as the data is anonymised. This data comprises physiological data gathered using a continuous glucose monitor (blood glucose concentration levels) and wristband device (heart rate, skin conductance, skin temperature), activity data (acceleration, step count), and self-reported data (meal intake, insulin, exercise, work, sleep, and stressors).

3.2 Dealing with Missing Values

A non-trivial aspect of the datasets used for developing glucose prediction models is the aspect of missingness. This is evident in the Ohio T1DM dataset with missingness present in both physiological variables and self-reported data [4].

Linear Interpolation: The blood glucose values that are missing in this dataset are typically missing at random. This could be attributed to issues around replacing glucose sensors and/or transmitters, or dealing with faulty communication. As a result, we employ linear interpolation in the training set to handle imputation of missing blood glucose concentration levels in the dataset over a period of one hour. In the samples where more than an hour of CGM data is missing the sample is discarded from the training set. This is illustrated with an example sequence in (C) of Fig.1

On the other hand, features which comprise self-reported data the assumption is made that any missing values represent an absence of

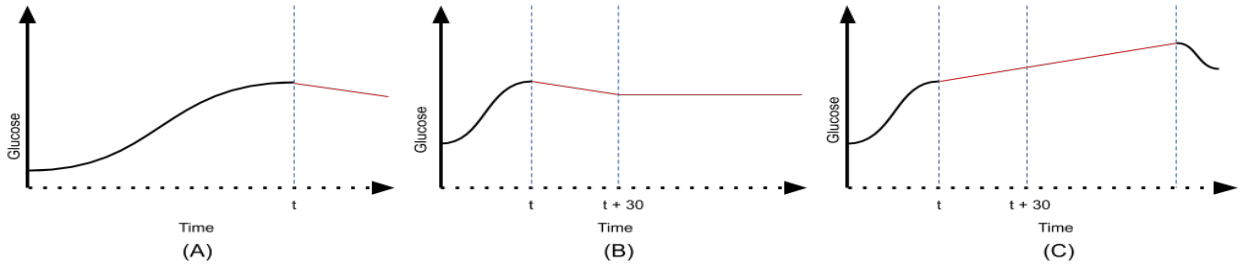


Figure 1. A visualisation of the imputation methods employed in this work. In (A) the input sequence has at least 30 minutes of recent values missing (eg. linear extrapolation). (B) shows the imputation scheme during testing for longer than 30 minutes of recent values missing (zero-order hold). Finally (C) shows the imputation scheme when the missing values of the input sequence are located between real values (linear interpolation).

said feature. Therefore all missing values in insulin, meal intake and reported exercise are imputed with zero.

The missingness in features from the self-reported data in the testing set is tackled similarly as in the training set. However, this is not the case for blood glucose concentration levels as interpolation when a current value at a given timestep is missing would lead to an inaccurate evaluation of model performance.

Extrapolation: In order to accurately evaluate the performance of the model we cannot always rely on interpolation at test time as this may require, in a real-time setting, an unknown future value to perform interpolation. Consequently, we need to rely on other methods of extrapolation to impute the missing glucose concentration levels. In this scenario (A), for gaps of data less than 30 minutes, we impute missing values with predicted values from the trained model. For missing recent values longer than 30 minutes as in (B), we pad the remaining values with the last computed value. In cases where, a gap larger than 30 minutes is evident in historical data and a current value is present at the given timestep, linear interpolation was then employed instead to provide a more accurate imputation.

3.3 Standardisation

To enable training the proposed model effectively, we perform transformation of the relevant input features (blood glucose concentration, insulin bolus, meal(carbohydrate) intake, and reported exercise). The blood glucose concentration levels are scaled down by a factor of 120. Similarly, the insulin bolus is scaled by 100 and meal intake values are scaled by 200 in the same range between features. The exercise values are transformed to a simple binary representation of the presence or absence of exercise, from the recorded exercise intensity on a range from 1-10.

4 METHODS

In this section we detail the machine learning technique that is used to provide the means of learning personalised models with the entire dataset. We detail the approach to develop the deep multitask network for personalisation. We provide a summary of the hyperparameters used in training as well and setting up the input for personalised multitask learning.

4.1 Multitask Learning

Multitask learning is an approach in machine learning that can be broadly described as a method of learning multiple tasks simultaneously with the aim of improving generalisation [1].

Multitask learning for personalisation has been used mainly in affective computing [13] with early work in diabetes management focusing on using multitask learning for developing prediction models for clustered groups of Type 1, Type 2, an non-diabetic participants [7] rather than leveraging similarities within groups such as gender, for personalised glucose predictions.

As seen Figure 2, the output from the shared layers are now fed into the individual(task)-specific fully connected layers of each user.

In a multitask setting of this kind, a multiplicative gating approach is used to ensure that the input corresponding to the particular user trains on just that user in the individual-specific layers. In that sense, at each iteration a batch that consists of data from a particular individual is used to train the shared layers and the layers specific to the individual.

4.2 CRNN Model

The deep learning model trained in the multitask learning setting is a convolutional recurrent neural network (CRNN) proposed by Li et. al [8] to perform short-term glucose prediction. This forms the basis of the single-task (STL) model. The convolutional recurrent model consists initially of a 3 temporal convolutional layers that perform a 1-D convolution with a Gaussian kernel over the sequence of input to extract features of various rates of appearance, followed by a max pooling layer after each convolution operation. The input is a 4-dimensional sequence that takes a 2-hour window of historical data.

The output from the convolutional layers performs feature extraction and feeds into a recurrent long short-term memory (LSTM) layer that is able to better model the temporal nature of the task.

The output from the shared layers feed into the fully connected layers of each user and to then provide the change in glucose value over the prediction horizon. This is then added to the current glucose value to provide the forecast glucose concentration level.

4.3 Loss Function

The loss function used for converging to the appropriate model for the glucose forecasting is the mean absolute error. This is expressed below as:

$$L(y, \hat{y}) = \frac{1}{N_{batch}} \sum_{k=1}^{N} |y - \hat{y}| \tag{1}$$

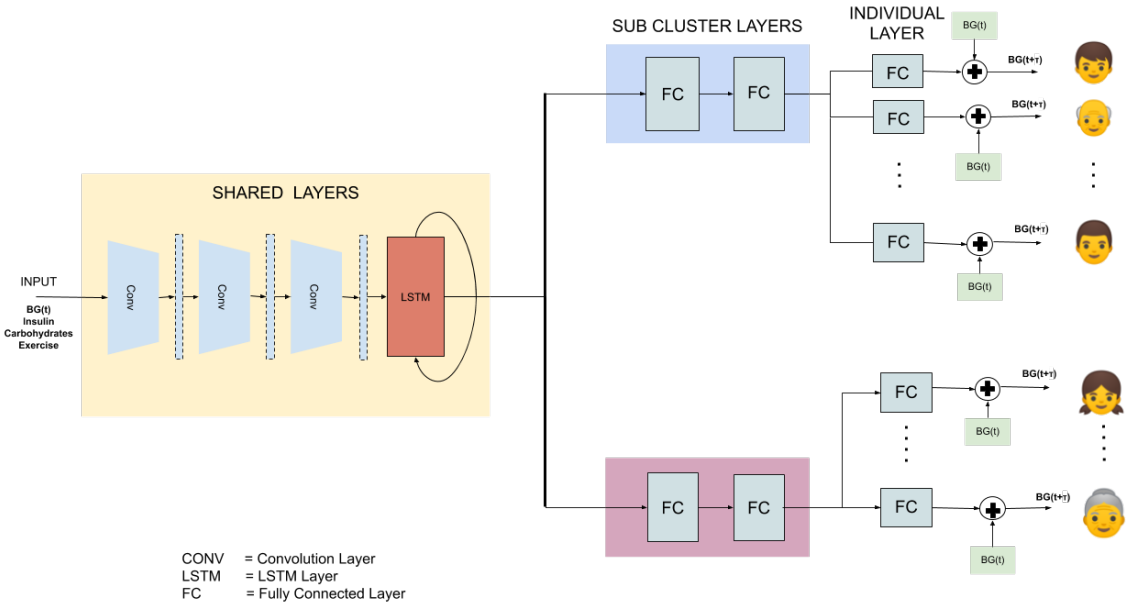


Figure 2. A detailed look at the formulation of convolutional recurrent networks in a multitask setting. In this setting, each user is represented as a task. In addition, the initial layers (convolutional and recurrent layers) are shared between each user, the next two (dense) layers are shared based on gender, and the last (dense) layer is specific to each user.

where $\hat{y}$ denotes the predicted results given the historical data and y denotes the reference change in glucose concentration over the relevant glucose prediction, and N_{batch} refers to the batch size.

4.4 Hyperparameters

The following table details provides the details of the hyperparameters used for the model architecture at each layer.

Table 1. A table detailing the size and dimensions of layers in the multitask CRNN model (MTCRNN)

Layer Description (layer)	Output Dimensions	No. of Parameters
Shared Convolutional Layers (Batch×Steps×Channels)		
(1) 1×4 conv	$128(1) \times 24 \times 8$	104
max_pooling, size 2	$128(1) \times 12 \times 8$	–
(2) 1×4 conv	$128(1) \times 12 \times 16$	528
max_pooling, size 2	$128(1) \times 6 \times 16$	–
(3) 1×4 conv	$128(1) \times 6 \times 32$	2080
max_pooling	$128(1) \times 3 \times 32$	–
Shared Recurrent Layer (Batch×Cells)		
(4) lstm	$128(1) \times 64$	24832
Sub-cluster Dense Layers (Batch×Units)		
(5) dense	$128(1) \times 256$	16640
(6) dense	$128(1) \times 32$	8224
Individual-Specific Dense Layers (Batch×Units)		
(7) dense	$128(1) \times 1$	33

The optimiser used for this work is Adam. The learning rate is 0.0053. The model is trained for 200 epochs. This value was obtained through grid search optimisation.

The model is developed on Keras 2.2.2, with a Tensorflow 1.5 backend. The training is performed on an NVIDIA GTX 1050 GPU.

The repository for the code accompanying the paper can be found at: https://github.com/jsmdaniels/ecai-bglp-challenge

5 RESULTS

5.1 Evaluation Metrics

The model is tested on data from six participant IDs: 540, 544, 552, 567, 584, 596.

The evaluation of the model is based on two metrics: root mean square error (RMSE) and the mean absolute error (MAE). The extrapolated points are not considered in calculating these metrics. The formulation of these metrics are provided below:

$$RMSE = \sqrt{\frac{1}{N}\sum_{k=1}^{N}(y-\hat{y})^2}, \tag{2}$$

$$MAE = \frac{1}{N}\sum_{k=1}^{N}|y-\hat{y}|. \tag{3}$$

where $\hat{y}$ denotes the predicted results given the historical data and y denotes the reference glucose measurement, and N refers to the data size.

In order to undertake a comprehensive evaluation of the model performance, the subsequent criteria for assessment are followed:

- **Performance evaluation over 30-minute and 60-minute prediction horizon (PH):** The RMSE and MAE for each participant is analysed for a the same length of values for both prediction horizons.
- **Comparison of training setting:** The performance of the multitask learning (MTL) approach is evaluated in the context of comparison with the performance of a single task learning (STL) approach which uses only patient specific data.

- **Multiple runs for each participant ID:** The multitask CRNN (MTCRNN) model uses randomly initialised weights at the start of training. Given the variable nature of this training procedure, the results reported are the average of 5 model runs.

The unit for results reported below is mg/dL. The best performance is in **bold**.

Table 2. A table showing prediction performance for 30 minutes the RMSE and MAE results of the six participants over 5 runs (CRNN)

	MTL		STL	
ID	RMSE	MAE	RMSE	MAE
540	**21.19±0.07**	**15.17±0.06**	22.45±0.39	16.21±0.34
544	**16.82±0.09**	**11.72±0.06**	18.63±1.59	12.57±0.23
552	**16.30±0.12**	**11.92±0.03**	17.11±0.24	12.68±0.49
567	**24.12±0.17**	**15.55±0.03**	24.73±0.45	16.01±0.71
584	**23.66±0.20**	**15.77±0.08**	24.30±0.48	16.20±0.23
596	**16.63±0.15**	**11.59±0.09**	16.78±0.20	12.00±1.77
Average	**19.79±0.06**	**13.62±0.05**	20.67±0.32	14.28±0.19

Table 3. A table showing the prediction performance for 60 minutes RMSE and MAE results of the six participants over 5 runs (CRNN)

	MTL		STL	
ID	RMSE	MAE	RMSE	MAE
540	**38.29±0.29**	**28.60±0.17**	41.06±0.24	30.33±0.69
544	**28.97±0.24**	20.77±0.20	29.60±0.37	**20.52±0.17**
552	**29.35±0.27**	**22.07±0.13**	30.32±0.10	22.53±0.13
567	40.19±0.79	28.77±0.13	**40.09±0.64**	**27.71±0.13**
584	37.82±0.78	26.88±0.37	**37.22±0.34**	**26.64±0.41**
596	**27.74±0.11**	**20.12±0.14**	28.13±0.48	20.30±0.41
Average	**33.73±0.24**	**24.54±0.08**	34.40±0.14	24.67±0.14

6 DISCUSSION

As seen in Table 3, the results shown provide a comprehensive evaluation of the model predictive performance.

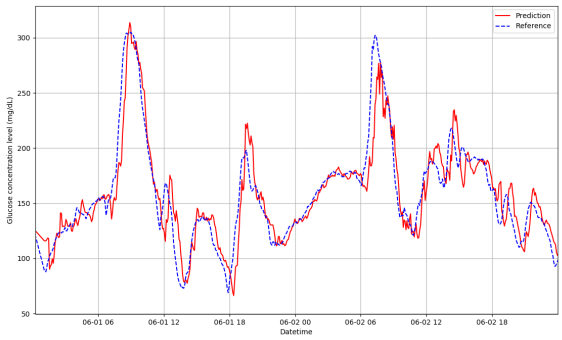


Figure 3. A graph showing the predictive performance of the model on participant ID:596 at a 30 minute predictive horizon.

Evidently, the model performance at PH = 30 minutes is better than the model performance at PH = 60 minutes, given that prediction at 60 minutes is a more complex task than prediction at 30 minutes.

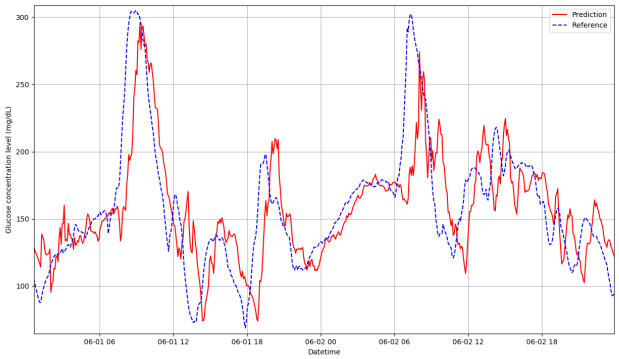


Figure 4. A graph showing the predictive performance of the model on participant ID:596 at a 60 minute predictive horizon.

Figures 3 and 4 exhibit the differences in performance as seen in the specific window for participant 596. The increased lag and reduced predictive performance can also be attributed to the higher chance of external activities (insulin, meals, exercise) that influence the blood glucose trajectory occurring over the prediction horizon.

The best predictive performances were achieved by the model with IDs 544, 552, 596 whereas, IDs 540, 567, and 584 exhibited worse performances over both 30 and 60 minute prediction horizons. An investigation of the glycaemic variability, using the coefficient of variation (CV) [2], of the training set of the former set of participants are stable (CV$\leq$36%) whereas the latter group are labile (CV$>$36%). The multitask learning approach definitively performs better over the single task approach over a 30-minute prediction horizon. However, the performance improvement of the MTL approach over a 60-minute prediction is not consistent across each participant and metric.

One potential issue with multitask learning is the issue of negative transfer. This can be described as a scenario in which one or more of the tasks (individuals) or sampled batches during training are not strongly correlated, degrading the learning in the shared layers, and subsequently the performance at test time.

7 CONCLUSION

In this work, we have presented a multitask convolutional recurrent neural network that is capable of performing short-term personalised predictions - 19.79±0.06mg/dL (RMSE) and 13.62±0.05mg/dL (MAE) at 30 minutes, as well as 33.73±0.24mg/dL (RMSE) and 24.54±0.15mg/dL (MAE) at 60 minutes. We work towards leveraging population data while still learning a personalised model. In the future, we hope to address further challenges such as negative transfer during learning that could improve the accuracy of individual models. This approach would enable more accurate models to be deployed in the face of limited personal data.

ACKNOWLEDGEMENTS

This work is supported by the ARISES project (EP/P00993X/1), funded by the Engineering and Physical Sciences Research Council.

REFERENCES

[1] Rich Caruana, 'Multitask Learning', *Machine Learning*, **28**(1), 41–75, (July 1997).

[2] Antonio Ceriello, Louis Monnier, and David Owens, 'Glycaemic variability in diabetes: clinical and therapeutic implications', *The Lancet Diabetes & Endocrinology*, (August 2018).

[3] E. I. Georga, V. C. Protopappas, D. Ardigò, M. Marina, I. Zavaroni, D. Polyzos, and D. I. Fotiadis, 'Multivariate Prediction of Subcutaneous Glucose Concentration in Type 1 Diabetes Patients Based on Support Vector Regression', *IEEE Journal of Biomedical and Health Informatics*, **17**(1), 71–81, (January 2013).

[4] Marzyeh Ghassemi, Tristan Naumann, Peter Schulam, Andrew L. Beam, and Rajesh Ranganath, 'Opportunities in Machine Learning for Healthcare', *arXiv:1806.00388 [cs, stat]*, (June 2018). arXiv: 1806.00388.

[5] Giacomo Cappon, Giada Acciaroli, Martina Vettoretti, Andrea Facchinetti, and Giovanni Sparacino, 'Wearable Continuous Glucose Monitoring Sensors: A Revolution in Diabetes Treatment', *Electronics*, **6**(3), 65, (September 2017).

[6] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, *Deep Learning*, MIT Press, 2016. http://www.deeplearningbook.org.

[7] Weixi Gu, Zimu Zhou, Yuxun Zhou, Miao He, Han Zou, and Lin Zhang, 'Predicting Blood Glucose Dynamics with Multi-time-series Deep Learning', in *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems - SenSys '17*, pp. 1–2, Delft, Netherlands, (2017). ACM Press.

[8] K. Li, J. Daniels, C. Liu, P. Herrero-Vinas, and P. Georgiou, 'Convolutional Recurrent Neural Networks for Glucose Prediction', *IEEE Journal of Biomedical and Health Informatics*, 1–1, (2019).

[9] Kezhi Li, Chengyuan Liu, Taiyu Zhu, Pau Herrero, and Pantelis Georgiou, 'GluNet: A Deep Learning Framework for Accurate Glucose Forecasting', *IEEE Journal of Biomedical and Health Informatics*, **24**(2), 414–423, (February 2020).

[10] Cindy Marling and Razvan Bunescu, 'The OhioT1DM Dataset for Blood Glucose Level Prediction', *In: The 5th International Workshop on Knowledge discovery in healthcare data.*, (2020). CEUR proceeding in press. Available at http://smarthealth.cs.ohio.edu/bglp/OhioT1DM-dataset-paper.pdf.

[11] John Martinsson, Alexander Schliep, Björn Eliasson, and Olof Mogren, 'Blood Glucose Prediction with Variance Estimation Using Recurrent Neural Networks', *Journal of Healthcare Informatics Research*, **4**(1), 1–18, (March 2020).

[12] C. Pérez-Gandía, A. Facchinetti, G. Sparacino, C. Cobelli, E.j. Gómez, M. Rigla, A. de Leiva, and M.e. Hernando, 'Artificial Neural Network Algorithm for Online Glucose Prediction from Continuous Glucose Monitoring', *Diabetes Technology & Therapeutics*, **12**(1), 81–88, (January 2010).

[13] Sara Ann Taylor, Natasha Jaques, Ehimwenma Nosakhare, Akane Sano, and Rosalind Picard, 'Personalized Multitask Learning for Predicting Tomorrow's Mood, Stress, and Health', *IEEE Transactions on Affective Computing*, 1–1, (2017).

[14] Taiyu Zhu, Kezhi Li, Jianwei Chen, Pau Herrero, and Pantelis Georgiou, 'Dilated Recurrent Neural Networks for Glucose Forecasting in Type 1 Diabetes', *Journal of Healthcare Informatics Research*, (April 2020).

Prediction of Blood Glucose Levels for People with Type 1 Diabetes using Latent-Variable-based Model

Xiaoyu Sun and **Mudassir Rashid** and **Mert Sevil** and **Nicole Hobbs** and **Rachel Brandt** and **Mohammad Reza Askari** and **Andrew Shahidehpour** and **Ali Cinar**[1]

Abstract. Regulation of blood glucose concentrations (BGCs) is a tough burden for people living with type 1 diabetes mellitus (T1DM). People with T1DM must administer exogenous insulin to maintain their BGC within an euglycemic range. Hyperglycemia (high BGC) and hypoglycemia (low BGC) can occur because of poor BGC management. Using recursively identified models to predict the future BGC values opens novel possibilities for improving the BGC regulation performance by adjusting the dose of insulin infusion, taking rescue carbohydrates, or both. The BGC prediction model can also benefit the development of artificial pancreas systems. In this paper, a latent variable (LV) based multivariable statistical modeling approach is applied to model BGC dynamics and forecast future BGCs. The LV-based model is a powerful linear method to build an empirical model by using the collected data. The model is evaluated with the Ohio T1DM dataset that contains BGCs from continuous glucose monitoring (CGM) sensors, basal and bolus insulin information from insulin pump, and additional information from wristbands or reported by the subject. The results indicate that the LV-based model can predict future BGC values with high accuracy for prediction horizons of 30 and 60 minutes.

1 Introduction

People with type 1 diabetes mellitus (T1DM), an immune disorder where the pancreas does not produce insulin, must administer exogenous insulin either by injections several times every day or infusion with an insulin pump to maintain their blood glucose concentrations (BGCs) within a safe range (70-180 mg/dL) [23]. Without effective regulation, people with T1DM may suffer from several long-term complications caused by hyperglycemia (high BGC), such as kidney failure, blindness, and the deterioration of cardiovascular health. They can also have low BGC (hypoglycemia), which may cause dizziness, diabetes coma, or even death because of the lack of energy for the brain [4].

Artificial pancreas (AP) systems are proposed to provide more reliable BGC management by automatically calculating the dose of insulin to infuse by using a closed-loop controller incorporating BGCs measured by continuous glucose monitoring (CGM) sensors, historical infused insulin data, and other available information, such as meal carbohydrates (CHOs) and exercise information [15, 9, 16, 18, 2, 10]. The closed-loop controller is the critical component in developing an AP system. Model-based controllers have become the preferred option in recent in silico and clinical studies where the future BGCs predicted by a model, historical CGM measurements, administered insulin, and constraint conditions are taken into account when computing the future insulin doses to be infused. The data-driven modeling techniques have been evaluated in many studies because of their computational tractability and identification efficiency.

The empirical modeling technologies for T1DM include linear and nonlinear methods. For nonlinear models, artificial neural networks (ANN) [1], convolutional neural networks (CNN) [26], recurrent neural networks (RNN) [13, 3], and other machine learning and deep learning techniques [17, 11] have been used to model the glucose dynamics. However, the nonlinear models can only be trained with a large data set, and it can be difficult to develop personalized models or to update the model parameters or structure. For the linear modeling methods, autoregressive model with exogenous inputs (ARX), and autoregressive moving average model with exogenous inputs (ARMAX) [7, 19, 25] are used to build personalized glucose prediction models with recursively updated model parameters where the future BGC values are modeled as a linear combination of historical measurements from sensors, administered insulin, and other information such as meal CHOs and exercise. Statistical methods based on latent variables (LVs) are demonstrated to have powerful abilities for various data analysis tasks, modeling, and process monitoring [20, 21], and has been proven as a good alternative linear model for type 1 diabetes [24].

In this paper, a novel multivariate statistical method proposed by Nelson and MacGregor [14, 8] where the score vector is estimated using missing data imputation technique is applied to model the glucose dynamics based on LVs derived from principal component analysis (PCA). There are three key steps in developing a BGC prediction model based on the LVs technique. First, a PCA is performed on the gathered data to decompose the data into a linear combination of scores and loadings. Then, the unobserved variables are estimated as conditional mean values computed from the gathered data and new measurements. Finally, the score of new observed data is estimated using incomplete observations and the future BGC values are predicted. The rest of this paper is organized as follows: Section 2 summarizes the pre-processing of the data set. The LV-based method is described and a glucose prediction model is developed in Section 3. The Ohio T1DM data set [12] is used to assess the performance of the model and the results are given and discussed in Section 4. Finally, some concluding remarks are provided in Section 5.

2 Data Pre-processing

2.1 Data

The Ohio T1DM dataset provided by the Blood Glucose Level Prediction (BGLP) challenge records eight weeks of data collected from

[1] Illinois Institute of Technology, USA, email: cinar@iit.edu

six T1DM patients with subject ID: 540, 544, 552, 567, 584, and 596. The data set contains BGC values measured by CGM sensors with a sampling time of 5 minutes, basal and bolus insulin information from the insulin pump, information collected from wristbands, and events (i.e., meal, exercise, work, illness, etc.) recorded by the subjects themselves (see [12] for details). However, there is no evidence on the accuracy of the subject-reported information, which may degrade the reliability of glucose prediction model and the wristband did not work continuously for a long period of time due to its limited battery power, which causes lots of missed data. Thus, only data collected from CGM sensors and insulin pumps is used in this study.

The insulin infused with an insulin pump includes basal insulin, given continuously since the defined start times, and bolus insulin, which is usually given to mitigate the effects of the rapid raise of BGC around meal times. Basal insulin is recorded as "basal" which is the basal insulin infusion rate and "temp basal" which defines the basal insulin infusion rate in specific time intervals to achieve better regulation of BGCs. There are three types of bolus insulin defined in the dataset: normal, normal dual and square dual. For "normal" bolus insulin, a certain dose of insulin is infused immediately to the patient, while for "normal dual" bolus insulin, a certain percentage of bolus is given to the subject up front and the remaining insulin amount is given gradually over a longer time duration. In this study, half of the dose is supposed to be given to the patient immediately and the remaining half is infused over the following 30 minutes. For "square dual" bolus insulin, the bolus insulin is administered continuously in the given time interval. The bolus and basal insulin values are converted to U/min and aligned to match the CGM sampling time.

The insulin on board (IOB), which represents the insulin that remains active within the body, is calculated from a physiological model [22] and is found useful in some previous works [1, 6]. The physiological model is represented as

$$\frac{dC_1(t)}{dt} = u(t) - K_{dia}C_1(t) \tag{1}$$

$$\frac{dC_2(t)}{dt} = K_{dia}\left(C_1(t) - C_2(t)\right) \tag{2}$$

$$IOB(t) = C_1(t) + C_2(t) \tag{3}$$

where C_1 and C_2 define two insulin compartments, $u(t)$ is the insulin infusion, and K_{dia} is a constant related to the duration of insulin action, set as 0.0182.

The dataset contains missing CGM measurements for in both the training and testing sets that cause discontinuous CGM curves. The gaps that are not greater than 3 samples in the training dataset are interpolated using first-order linear model. For the cases where more than 3 samples are missed, a single variable statistical model similar to the one used to modeling glucose dynamics is used. In the testing data set, with the intent of on-line implementation, only the historical CGM measurements are used to fill the missing gaps until new the CGM observations become available. Fig. 1 shows an example of the filled gap in the training dataset (Subject ID 540).

2.2 Batch Generation

The pre-processed CGM readings (y) and calculated IOB (IOB) are arranged as time series and a sliding window of 2 hours length is chosen to generate batch segments from the training data as

$$X_{train} = \begin{bmatrix} y(1) & \cdots & y(24) & IOB(1) & \cdots & IOB(24) \\ y(2) & \cdots & y(25) & IOB(2) & \cdots & IOB(25) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

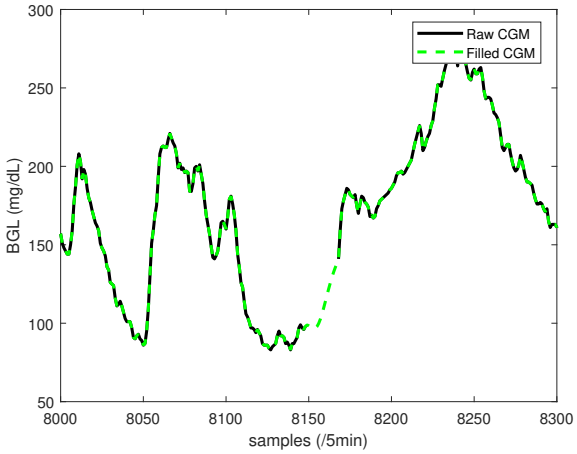


Figure 1. Illustration of the filled gap in training data (Subject ID: 540, 18 CGM measurements are missed)

where each row can be considered as an observation and each column is the values of a single variable.

3 Methods

A multivariable statistical technique based on LVs [14, 8] is developed to predict the future BGC values, where a LV model is developed using the PCA algorithm. Then, the conditional mean values are estimated from the same distribution, and future BGCs are predicted with LVs and incomplete observations.

3.1 Latent Variable Model

Consider a data matrix $X\,(N \times M)$ that contains N observations (rows) and M variables (columns), then a linear combination of an observation x in dataset X can be written as $t = p_1x_1 + \ldots + p_Mx_M$, where t is a new vector in the same space as x. The fundamental idea behind PCA is to find the loading vector p that maximizes the variation of t, thus the first LV of PCA model can be calculated by solving the following problem:

$$\underset{\|p\|=1}{\arg\max}\left(t^Tt\right) = \underset{\|p\|=1}{\arg\max}\left(p^TX^TXp\right) \tag{4}$$

where p is the vector of regression coefficients. Accordingly, the dataset X can be expressed as $X = tp^T + E$ with E denoting the residual matrix. We can get more components by solving the following problem:

$$\underset{\|p\|=1}{\arg\max}\left\|X - tp^T\right\|^2 \tag{5}$$

Traditionally, the successive progress is evaluated by

$$\frac{\|X\|^2 - \|E\|^2}{\|X\|^2}100\% \tag{6}$$

which is referred to as the percentage of explained variation of t and it is fixed as 95% in this study. If the first A (A is much smaller than the rank of X) significant components can summarize sufficiently well the dataset X, then a LV model developed using PCA algorithm can be expressed as

$$X = t_1p_1^T + t_2p_2^T + \ldots + t_Ap_A^T + E = T_{1:A}P_{1:A}^T + E \tag{7}$$

where $T_{1:A} = [t_1, \ldots, t_A]\,(N \times A)$ and $P_{1:A} = [p_1, \ldots, p_A]\,(M \times A)$ are now matrices containing A score vectors and A loading vectors, respectively.

3.2 Conditional Mean Replacement Method

For a new test observation z that was not used in the model development but is drawn from the same distribution as observations in X, the scores τ of the first A significant components of the new observation can then be calculated as

$$\tau_{1:A} = P_{1:A}^T z \tag{8}$$

Consider a situation where only part of the object z is observed, it is nature to assume the first R variables are measured and the remaining $(M - R)$ variables are unobserved, then without loss of generality, we have

$$z = \begin{bmatrix} z^{\#} \\ z^{*} \end{bmatrix}$$

where $z^{\#}$ is the observed variables and z^* represents the missing measurements. This induces the following partition in X:

$$X = \begin{bmatrix} X^{\#} & X^{*} \end{bmatrix}$$

where $X^{\#}$ contains the first R columns of X and X^* is made up of the remaining $(M - R)$ columns. Correspondingly, the loading matrix P can be partitioned as

$$P = \begin{bmatrix} P^{\#} \\ P^{*} \end{bmatrix}$$

where $P^{\#}$ is the submatrix comprising the first R rows of P and the remaining $(M - R)$ rows are defined as matrix P^*. Then, the score vector $\tau_{1:A}$ of the new observation z can be rewrite as

$$\hat{\tau}_{1:A} = P_{1:A}^{\#T} z^{\#} + P_{1:A}^{*T} z^{*} \tag{9}$$

For a given data matrix X and a new observation z with $z^{\#}$ denoting the measured variables that follow the same distribution as observations in X, the missing variables z^* of z can be estimated as the expected values from the conditional normal distribution:

$$\hat{z}^{*} = E\left[z^{*} \,\middle|\, z^{\#}, S\right] \tag{10}$$

Substituting the expression into the estimator of score vector of the new observation yields:

$$\hat{\tau}_{1:A} = \Theta_{1:A} P_{1:A}^{\#T} \left(S^{\#\#}\right)^{-} z^{\#} \tag{11}$$

where $S = \frac{X^T X}{N-1} = \begin{bmatrix} S^{\#\#} & S^{\#*} \\ S^{*\#} & S^{**} \end{bmatrix}$ is the covariance matrix of X and Θ is the square diagonal matrix consisted of the eigenvalues $\lambda = [\lambda_1, \ldots, \lambda_H]$ in descending order and H is the rank of X.

Finally, the unmeasured variables z^* in z can then be calculated from the estimated score vector along with the loading matrix:

$$\hat{z}^{*} = P_{1:A}^{*} \hat{\tau}_{1:A} \tag{12}$$

which yields the future predictions given the past observed glucose values.

3.3 Glucose prediction based on LV model

An accurate glucose prediction model could benefit diabetes management and significantly reduce the risk of hypoglycemia. To predict the future 60 minutes glucose values, the preceding one hour of data is assumed available and marked as $z^{\#}$. Then, the glucose prediction model can be developed and the subsequent future hour of BGC values can be predicted online as follows:

1. The batch data set X_{train} is generated for each subject using the training dataset.
2. While a new observation $z^{\#}$ is available:
   (a) Calculate the similarities between the new object $z^{\#}$ and observations in submatrix $X^{\#}$, select the most similar N observations from X_{train} to form a new data matrix X.
   (b) Develop PCA model using data matrix X as stated in (7).
   (c) Estimate the score vector τ of the new observation $z^{\#}$ as stated in (11).
   (d) Predict the next 1 hour's glucose values as stated in (12).

The above process was implemented in MATLAB 2019b and the future one hour's BGC values (12 samples) can be forecasted by feeding the testing data to the model. The codes are available: `https://github.com/xiaoyu1115/BGLP_2020.git`.

4 Results

In the clinical study, the training data from Ohio T1DM dataset with subject ID: 540, 544, 552, 567, 584, and 596 were first divided into two parts: the first 90% of data were used to develop PCA model and the model is tested with the remaining 10% of data to determine the number of observations that should be used in building the LV-based glucose prediction models. Using this approach, the computational burden in the modeling progress can be reduced significantly. Feeding both the processed training and testing data into the modeling process described in Section 3, the future glucose concentrations can be predicted with prediction horizons of up to 60 minutes. The root mean square errors (RMSEs) and mean absolute errors (MAEs) of the 6 subjects are summarized in Table 1. The prediction results are also analysed using Clark Error Grid (CEG) [5] and the percentages of data distributed in Zone A to Zone D are summarized in Table 1 as well.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N} \left(y\left(i\right) - \hat{y}\left(i\right)\right)^2} \tag{13}$$

$$MAE = \frac{1}{N}\sum_{i=1}^{N} \left|y\left(i\right) - \hat{y}\left(i\right)\right| \tag{14}$$

where y is the BGC values measured by CGM sensors, $\hat{y}$ is the predicted BGCs, and N is the total data points in the testing dataset for each subject.

In Table 1, the RMSE varies from 16.66 to 22.76 mg/dL for the prediction horizon of 30 minutes with an average value of 19.37 mg/dL. The RMSE varies from 26.81 to 38.99 mg/dL for the prediction horizon of 60 minutes with an average value of 32.59 mg/dL. It is reasonable to see this increase in RMSE because the unknown disturbances including meals and exercise that will influence the glucose dynamics significantly. The relationships between the prediction horizon and RMSE or MAE in Figure 2 also indicate that the prediction accuracy decreases with increasing prediction horizon. An

Table 1. RMSE (mg/dL), MAE (mg/dL) and CEG (%) results of the LV-based model (STD: standard deviation)

	PH=30 minutes						PH=60 minutes					
Subject	RMSE	MAE	Zone A	Zone B	Zone C	Zone D	RMSE	MAE	Zone A	Zone B	Zone C	Zone D
540	20.76	15.23	84.92	12.55	0.03	2.50	38.99	29.75	57.84	35.58	0.24	6.35
544	16.70	11.65	93.23	6.29	0	0.48	26.81	19.77	78.77	19.67	0.07	1.48
552	16.66	12.36	88.05	10.88	0	1.06	29.41	23.08	64.97	31.63	0.09	3.32
567	22.76	15.30	86.03	12.28	0.04	1.60	37.95	28.13	57.93	34.54	0.34	7.15
584	22.22	15.86	85.11	13.87	0	1.02	34.81	26.57	67.66	30.08	0.15	2.11
596	17.12	12.15	90.11	8.31	0	1.57	27.57	20.53	72.06	25.23	0.04	2.67
Mean	19.37	13.76	87.91	10.70	0.01	1.30	32.59	24.64	66.54	29.46	0.15	3.85
STD	2.87	1.89	3.27	2.87	0.02	0.68	5.35	4.12	8.17	6.03	0.12	2.34

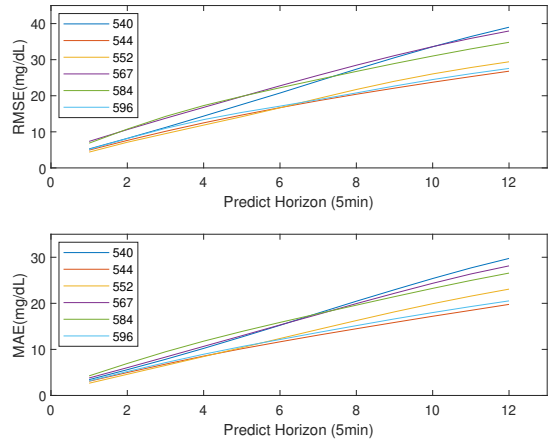


Figure 2. Relationship between prediction horizon and prediction accuracy

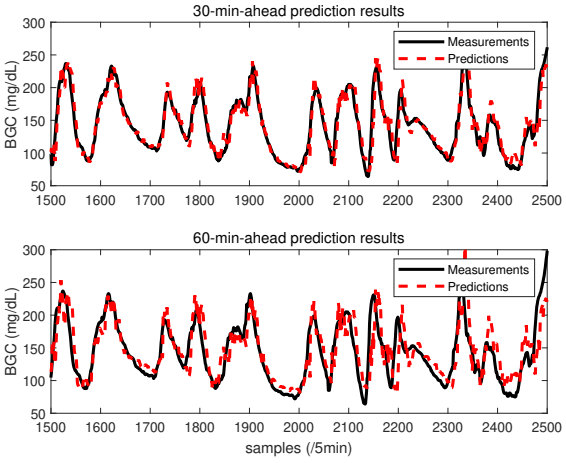


Figure 3. BGC prediction for Subject 544 using LV-based model

average of 87.91% data lie in zone A of CEG which indicates the high clinical accuracy of the model with prediction horizon of 30 minutes. The percentage of data in zone B increase dramatically as the prediction horizon increases to 60 minutes.

The forecasting performance of the model is better for subjects 544, 552, and 596 than subjects 540, 552, and 567. One of the reason is there are less noise and sensor failures contained in the dataset provided by subjects 544, 552, and 596. A filter might improve prediction performance for this case. And many gaps are observed in CGM measurements, which also deteriorate the prediction accuracy of the model in which the interpolated CGM data were used to predict the further BGC values.

The predicted BGC values for subject 544 with prediction horizon of 30 minutes and 60 minutes are shown in Figure 3. For the prediction horizon of 30 minutes, the LV-based model can predict the future BGC values with a comparable high accuracy, and most of the hyperglycemia and hypoglycemia events can be forecasted on time. When the prediction horizon is increased to 60 minutes, the prediction accuracy decreases, but is still acceptable. The prediction values can still provide an insight on the variation of BGC which can be used to tune the insulin infusion rate as a reference.

5 Conclusion

In this paper, an online recursively identified LV-based modeling approach is developed to predict the future BGC values with prediction horizons of 30 and 60 minutes. With the pre-processed dataset, the LV-based model is developed to calculate the LVs using a small submatrix of the training dataset when a new observation is available. The future blood glucose values are predicted as a linear combination of estimated scores and loadings. The proposed model is evaluated with the Ohio T1DM dataset and the results demonstrate the effectiveness of the model. Although the measurement noise is weight-averaged in the LV-based model, it still has a significant influence on modeling and prediction progress. Online denoising techniques would be one of the future study directions that might improve the prediction accuracy. Further, integrating other data fields with the personalized physiological models is a potential approach to improve the prediction performance in the future work.

ACKNOWLEDGEMENTS

The work of Xiaoyu Sun was supported by the China Scholarship Council under grant 201906080136. Funds provided by the Hyosung S. R. Cho Endowed Chair at Illinois Institute of Technology to Ali Cinar is gratefully acknowledged.

REFERENCES

[1] Arthur Bertachi, Lyvia Biagi, Iván Contreras, Ningsu Luo, and Josep Vehí, 'Prediction of blood glucose levels and nocturnal hypoglycemia using physiological models and artificial neural networks', in *3rd International Workshop on Knowledge Discovery in Healthcare Data, KDH@ IJCAI-ECAI 2018, 13 July 2018*, pp. 85–90.

[2] Marc Breton, Anne Farret, Daniela Bruttomesso, Stacey Anderson, Lalo Magni, Stephen Patek, Chiara Dalla Man, Jerome Place, Susan Demartini, and Simone Del Favero, 'Fully integrated artificial pancreas in type 1 diabetes: modular closed-loop glucose control maintains near normoglycemia', *Diabetes*, **61**(9), 2230–2237, (2012).

[3] Jianwei Chen, Kezhi Li, Pau Herrero, Taiyu Zhu, and Pantelis Georgiou, 'Dilated recurrent neural network for short-time prediction of glucose concentration', in *3rd International Workshop on Knowledge Discovery in Healthcare Data, KDH@ IJCAI-ECAI 2018, 13 July 2018*, pp. 69–73.

[4] Ali CinAr, 'Artificial pancreas systems: An introduction to the special issue', *IEEE Control Systems*, **38**(1), 26–29, (2018).

[5] William L Clarke, 'The original clarke error grid analysis (ega)', *Diabetes technology & therapeutics*, **7**(5), 776–779, (2005).

[6] Iván Contreras, Arthur Bertachi, Lyvia Biagi, Josep Vehí, and Silvia Oviedo, 'Using grammatical evolution to generate short-term blood glucose prediction models.', in *3rd International Workshop on Knowledge Discovery in Healthcare Data, KDH@ IJCAI-ECAI 2018, 13 July 2018*, pp. 91–96, (2018).

[7] Meriyan Eren-Oruklu, Ali Cinar, Lauretta Quinn, and Donald Smith, 'Estimation of future glucose concentrations with subject-specific recursive linear models', *Diabetes Technology and Therapeutics*, **11**(4), 243–253, (2009).

[8] Jesus Flores-Cerrillo and John F MacGregor, 'Latent variable mpc for trajectory tracking in batch processes', *Journal of process control*, **15**(6), 651–663, (2005).

[9] Iman Hajizadeh, Mudassir Rashid, Kamuran Turksoy, Sediqeh Samadi, Jianyuan Feng, Mert Sevil, Nicole Hobbs, Caterina Lazaro, Zacharie Maloney, Elizabeth Littlejohn, et al., 'Incorporating unannounced meals and exercise in adaptive learning of personalized models for multivariable artificial pancreas systems', *Journal of Diabetes Science and Technology*, **12**(5), 953–966, (2018).

[10] Roman Hovorka, Valentina Canonico, Ludovic J. Chassin, Ulrich Haueter, Massimo Massi-Benedetti, Marco Orsini Federici, Thomas R. Pieber, Helga C. Schaller, Lukas Schaupp, Thomas Vering, and Malgorzata E. Wilinska, 'Nonlinear model predictive control of glucose concentration in subjects with type 1 diabetes', *Physiological Measurement*, **25**(4), 905–920, (2004).

[11] Kezhi Li, John Daniels, Chengyuan Liu, Pau Herrero-Vinas, and Pantelis Georgiou, 'Convolutional recurrent neural networks for glucose prediction', *IEEE journal of biomedical and health informatics*, (2019).

[12] Cindy Marling and Razvan Bunescu, 'The ohiot1dm dataset for blood glucose level prediction: Update 2020'.

[13] John Martinsson, Alexander Schliep, Björn Eliasson, Christian Meijner, Simon Persson, and Olof Mogren, 'Automatic blood glucose prediction with confidence using recurrent neural networks', in *3rd International Workshop on Knowledge Discovery in Healthcare Data, KDH@ IJCAI-ECAI 2018, 13 July 2018*, pp. 64–68.

[14] Philip RC Nelson, Paul A Taylor, and John F MacGregor, 'Missing data methods in pca and pls: Score calculations with incomplete observations', *Chemometrics and Intelligent Laboratory Systems*, **35**(1), 45–65, (1996).

[15] Sediqeh Samadi, Kamuran Turksoy, Iman Hajizadeh, Jianyuan Feng, Mert Sevil, and Ali Cinar, 'Meal detection and carbohydrate estimation using continuous glucose sensor data', *IEEE Journal of Biomedical and Health Informatics*, **21**(3), 619–627, (2017).

[16] Mert Sevil, Mudassir Rashid, Zacharie Maloney, Iman Hajizadeh, Sediqeh Samadi, Mohammad Reza Askari, Nicole Hobbs, Rachel Brandt, Minsun Park, Laurie Quinn, et al., 'Determining physical activity characteristics from wristband data for use in automated insulin delivery systems', *IEEE Sensors Journal*, (2020).

[17] Qingnan Sun, Marko V Jankovic, Lia Bally, and Stavroula G Mougiakakou, 'Predicting blood glucose with an lstm and bi-lstm based deep neural network', in *2018 14th Symposium on Neural Networks and Applications (NEUREL)*, pp. 1–5. IEEE, (2018).

[18] Kamuran Turksoy, Elizabeth Littlejohn, and Ali Cinar, 'Multimodule, multivariable artificial pancreas for patients with type 1 diabetes: regulating glucose concentration under challenging conditions', *IEEE Control Systems Magazine*, **38**(1), 105–124, (2018).

[19] Kamuran Turksoy, Laurie Quinn, Elizabeth Littlejohn, and Ali Cinar, 'Multivariable adaptive identification and control for artificial pancreas systems', *IEEE Transactions on Biomedical Engineering*, **61**(3), 883–891, (2013).

[20] C. Undey and A. Cinar, 'Statistical monitoring of multistage, multiphase batch processes', *IEEE Control Systems Magazine*, **22**(5), 40–52, (2002).

[21] Cenk Ündey, Sinem Ertunç, Eric Tatara, Fouad Teymour, and Ali Çınar, 'Batch process monitoring and its application to polymerization systems', in *Macromolecular Symposia*, volume 206, pp. 121–134. Wiley Online Library, (2004).

[22] Malgorzata E Wilinska, Ludovic J Chassin, Helga C Schaller, Lukas Schaupp, Thomas R Pieber, and Roman Hovorka, 'Insulin kinetics in type-1 diabetes: continuous and bolus delivery of rapid acting insulin', *IEEE Transactions on Biomedical Engineering*, **52**(1), 3–12, (2005).

[23] Xia Yu, Mudassir Rashid, Jianyuan Feng, Nicole Hobbs, Iman Hajizadeh, Sediqeh Samadi, Mert Sevil, Caterina Lazaro, Zacharie Maloney, Elizabeth Littlejohn, Laurie Quinn, and Ali Cinar, 'Online glucose prediction using computationally efficient sparse kernel filtering algorithms in type-1 diabetes', *IEEE Transactions on Control Systems Technology*, 1–13, (2018).

[24] Chunhui Zhao, Eyal Dassau, Lois Jovanovič, Howard C Zisser, Francis J Doyle III, and Dale E Seborg, 'Predicting subcutaneous glucose concentration using a latent-variable-based statistical method for type 1 diabetes mellitus', *Journal of diabetes science and technology*, **6**(3), 617–633, (2012).

[25] Hong Zhao, Chunhui Zhao, Chengxia Yu, and Eyal Dassau, 'Multiple order model migration and optimal model selection for online glucose prediction in type 1 diabetes', *AIChE Journal*, **64**(3), 822–834, (2018).

[26] Taiyu Zhu, Kezhi Li, Pau Herrero, Jianwei Chen, and Pantelis Georgiou, 'A deep learning algorithm for personalized blood glucose prediction', in *3rd International Workshop on Knowledge Discovery in Healthcare Data, KDH@ IJCAI-ECAI 2018, 13 July 2018*, pp. 74–78.

Data Fusion of Activity and CGM for Predicting Blood Glucose Levels

Hoda Nemat[1] and **Heydar Khadem**[1] and **Jackie Elliott**[2] and **Mohammed Benaissa**[1]

Abstract. This work suggests two methods—both relying on stacked regression and data fusion of CGM and activity—to predict the blood glucose level of patients with type 1 diabetes. *Method 1* uses histories of CGM data appended with the average of activity data in the same histories to train three base regressions: a multilayer perceptron, a long short- term memory, and a partial least squares regression. In *Method 2*, histories of CGM and activity data are used separately to train the same base regressions. In both methods, the predictions from the base regressions are used as features to create a combined model. This model is then used to make the final predictions. The results obtained show the effectiveness of both methods. *Method 1* provides slightly better results.

1 INTRODUCTION

The literature emphasises the importance of the management of type 1 diabetes mellitus (T1DM) in reducing complications associated with the disease [1], [2]. The key role in T1DM management is to control blood glucose level (BGL) to remain in a normal range [3], [4].

The prediction of BGL from current and past information can be a useful contributor [5]. BGL prediction could provide early warnings concerning inadequate glycaemic control to prevent the occurrence of an adverse glycemic status [6], [7].

BGL prediction models could be classified into three main groups: physiological models, data-driven models, and hybrid models. Data-driven models explain the relationship between the present and past information to BGL prediction. In this regard, machine learning and time series approaches have been widely used [5].

Many studies have proposed data-driven BGL prediction methodologies. Mirshekarian et al. [8], Bertachi et al. [9], Martinsson et al. [10], Zhu et al. [11] and Xie et al. [12] in separate studies, developed prediction models to forecast BGL with a prediction horizon of up to 60 minutes.

Mirshekarian's model was based on a recursive neural network (RNN), which utilised long short- term memory (LSTM) units. CGM, insulin, meal, and activity information were inputs of their model. Bertachi used physiological models of insulin, carbohydrate, and activity on board to train an artificial neural network (ANN). Martinsson proposed an RNN model trained on historical blood glucose information to predict BGL in two horizons of 30 and 60 minutes. Zhu generated a dilated deep convolutional neural network fed by CGM, insulin, and carbohydrate intake as inputs. Xie applied an autoregression with exogenous inputs approach to predict BGL by exploiting current and past information of CGM data.

Physical activity is a critical factor in diabetes management. Therefore, investigation of the activity data in BGL prediction models is encouraged [13]. However, developing models with high accuracy using activity and CGM data is challenging, and limited studies have been done in this area. Data fusion of activity and CGM data normally result in models with a performance not comparable with those using CGM alone.

This paper proposes two novel CGM and activity data fusion methods to generate BGL prediction models with performance comparable with those using CGM data alone.

2 DATASET

To develop BGL prediction algorithms, we used the OhioT1DM dataset [14]. The dataset contains eight weeks' worth data of 12 people with T1DM. The data of six patients was released in 2018 for the first BGL prediction challenge [15] and data for additional six patients (referred by ID 540, 544, 552, 567, 584, and 596) was released for the second BGL prediction challenge in 2020 [14]. In this work, we used the data of the latter six patients.

The dataset includes data of CGM sensor, physical activity band, physiological sensor, and self-reported life-event. Among the different collected data, we explored CGM and activity data which were collected every 5 and 1 minutes, respectively. Detailed information about the sensors and devices as well as characteristics of the patients has been published [14], [15].

In the dataset, there are three types of activity data consisting of galvanic skin response, skin temperature, and magnitude of acceleration. In this work, we only used the data of the magnitude of acceleration. Hereafter, for simplicity, 'magnitude of acceleration' is referred to as 'activity'.

3 METHODOLOGY

This section presents the information about data preprocessing and the methodologies developed for the prediction of BGL.

This paper is submitted to the second Blood Glucose Level Prediction Challenge 2020, the 5th International Workshop on Knowledge Discovery in Healthcare Data.

[1] Department of Electronic and Electrical Engineering, University of Sheffield, UK, email addresses: hoda.nemat@sheffield.ac.uk, h.khadem@sheffield.ac.uk, m.benaissa@sheffiels.ac.uk

[2] Department of Oncology and Metabolism, University of Sheffield, UK, email address: j.elliott@sheffield.ac.uk

3.1 Preprocessing

Missing data in the training set is imputed using linear interpolation. For the testing set, on the other hand, linear extrapolation is used. This is to assure that future data is not seen by the model, and that the model can be used for a real-time application. Thus, we convert CGM and activity data to regular time series without any missing data in 5-minute and 1-minute intervals, respectively.

The next step was to unify the resolution of CGM and activity data. To do so, we downsampled the activity time series data to 5-minute intervals by capturing the nearest activity data to each CGM data and discarding the rest.

There were a considerable number of unavailable activity data at the beginning and/or end of training and/or test set. This was due to the difference in wear time of CGM and activity sensors. For these points average of activity data in the training set is used rather than linear interpolation or extrapolation. Table 1 shows the number of unavailable activity data for each patient ID.

Table 1. The number of non-existent activity data points in training and testing sets per data contributor.

Patient ID	testing set	training set
540	547	31
544	0	125
552	622	505
567	0	108
584	3	123
596	80	18

Another data preprocessing step was to reframe a time series problem to a supervised learning task. To this end, time series data were transformed into samples with lag observations as input and future observations as output. We use a rolling window with a history length of 6 or 12 data points for the input, which has the information of 30- or 60- minute history, respectively. Also, the output of each sample is a vector with 6 or 12 data points corresponding to prediction horizons of 30- and 60- minute, respectively.

3.2 Regression tools

Three base regressions and a stacked regression technique are used as tools to develop the final prediction models.

3.2.1 Base regressions

- *Multilayer perceptron (MLP)*

MLP [16] is an ANN that can be used for time series forecasting. In this work, a single-hidden-layer MLP model was used. The model comprised a dense layer of 100 nodes with an activation function of rectified linear unit (ReLU) followed by an output layer. Adam and mean absolute error were used as an optimiser and a loss function, respectively. The learning rate was 0.01, and the model was fitted with 100 epochs.

- *Long short-term memory (LSTM)*

RNN is also an artificial neural network suitable for working with sequential data. We used a vanilla LSTM recurrent network [17] with vector output which is used for multi-step ahead forecast. The model was composed of a hidden layer with 200 units followed by a fully-connected layer with 100 nodes and an output layer. Both hidden layers used ReLU as the activation function. Mean squared error was the loss function, Adam was the optimiser. The model trained with 100 epochs with a learning rate of 0.01.

- *Partial least squares regression (PLSR)*

PLSR carries considerable popularity in different applications, such as glucose sensing [18]. In this work, PLSR was applied as a regression tool. Different values were considered for the number of components—ranging from one to the length of the input window. Each time, the predicted residual sum of squares ($PRESS$) was calculated as follows.

$$PRESS = \sum_{i=1}^{N} (y - \hat{y}_i)^2 \quad (1)$$

Where, N is the size of the evaluation set, and y_i is the reference value, and $\hat{y}_i$ is the predicted value.

The number of components (A) resulting in the minimum value for $PRESS/(N - A - 1)$ is then selected [19].

3.2.2 Stacked regression

Stacked regression is applied to enhance the performance of BGL prediction [20]. This technique uses predictions from a number of models—first-level models—as features to train a new model—second-level model. In this work, a stacked regression structure was employed where the three base regressions mentioned in 2.3.1 were set as its first-level models and a PLSR as the second-level model (Figure 1).

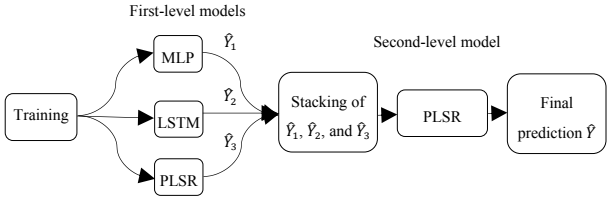


Figure 1. Diagram of the developed stacked regression.

3.3 Prediction methods

We developed two different methods using the stacked regression structure mentioned above to fuse CGM and activity data. Using these methods, models were then created to predict BGL of each patient for both horizons of 30 and 60 minutes. For each prediction horizon, two histories of 30 and 60 minutes were tried for training purposes.

3.3.1 Method 1

This method used the average value of activity data added to the window of CGM data to train the first-level models.

3.3.2 Method 2

In this method, the first-level models were trained twice. Once using a history of CGM data, and once using a history of activity data, thus producing six first-level models rather than three.

Table 3. Evaluation results of the first-level models of *Method 1* using a history of 30 minutes.

Patient ID	Model	PH: 30 min		PH: 60 min	
		RMSE	MAE	RMSE	MAE
540	PLSR	22.13	16.60	41.09	31.74
	MLP	21.96 ± 0.29	16.46 ± 0.21	40.53 ± 0.38	30.95 ± 0.33
	LSTM	**21.22 ± 0.12**	**15.82 ± 0.08**	**39.65 ± 0.28**	**30.38 ± 0.28**
544	PLSR	18.08	13.33	31.80	24.71
	MLP	17.95 ± 0.07	12.87 ± 0.13	31.61 ± 0.32	24.27 ± 0.71
	LSTM	**17.62 ± 0.20**	**12.60 ± 0.32**	**30.79 ± 0.29**	**23.02 ± 0.67**
552	PLSR	16.76	12.77	30.23	23.67
	MLP	16.96 ± 0.19	12.69 ± 0.21	30.38 ± 0.36	23.42 ± 0.61
	LSTM	**16.44 ± 0.17**	**12.18 ± 0.22**	**29.89 ± 0.47**	**22.53 ± 0.40**
567	PLSR	20.97	15.04	37.41	28.15
	MLP	21.44 ± 0.63	15.60 ± 0.76	37.96 ± 1.45	29.01 ± 1.35
	LSTM	**20.61 ± 0.20**	**14.64 ± 0.32**	**36.36 ± 0.31**	**27.08 ± 0.43**
584	PLSR	22.07	16.21	36.85	27.85
	MLP	21.60 ± 0.12	15.61 ± 0.14	**36.54 ± 0.74**	**27.27 ± 0.89**
	LSTM	**21.55 ± 0.26**	**15.58 ± 0.27**	36.75 ± 1.69	27.62 ± 2.08
596	PLSR	17.79	12.76	29.63	22.05
	MLP	18.01 ± 0.16	12.99 ± 0.17	29.75 ± 0.69	21.93 ± 0.38
	LSTM	**17.23 ± 0.17**	**12.25 ± 0.29**	**29.17 ± 0.22**	**21.29 ± 0.32**
Average	PLSR	19.63	14.45	34.50	26.36
	MLP	19.65 ± 0.24	14.37 ± 0.27	34.46 ± 0.66	26.14 ± 0.71
	LSTM	**19.11 ± 0.19**	**13.85 ± 0.25**	**33.77 ± 0.54**	**25.32 ± 0.70**

3.4 Evaluation

In the Ohio dataset, the last 10 days' worth of data for each contributor was allocated as the testing set and the rest as training [14]. To train and evaluation purposes, we used the training and testing sets, respectively. Extrapolated data and, the first 60 minutes of the test set was excluded when calculating the evaluation metrics. The latter is because the testing set starts immediately after the training set, and they are chronologically close to each other. Summarised statistics of the testing set for each patient is given in Table 2.

Table 2. The statistics of the patients' testing set.

Patient ID	Original data point	Imputed data point	Evaluation data point
540	2896	3066	2884
544	2716	3136	2704
552	2364	3950	2352
567	2389	2871	2377
584	2665	2995	2653
596	2743	3003	2731

Root mean square error (RMSE) and mean absolute error (MAE) were calculated as follows and considered as evaluation metrics.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{N}} \quad (2)$$

$$MAE = \frac{\sum_{i=1}^{N}|y_i - \hat{y}_i|}{N} \quad (3)$$

Where y_i, $\hat{y}_i$, and N have the same meaning as in (1).

4 RESULTS AND DISCUSSION

In this section, the results of RMSE and MAE for prediction models are provided for both prediction horizons of 30 and 60 minutes. Models with a performance dependent on random initialisation ran five times, and the mean and standard deviation of results are reported. We have used the acronym PH for the prediction horizon in the tables.

4.1 Method 1

Table 3 displays the evaluation results of the first-level models of *Method 1* when a history of 30 minutes is used for training. Based on the RMSE and MAE values, in both prediction horizons, LSTM had the best prediction performance for all patients except 584. For this patient, MLP had the best result. PLSR, as a simple linear regressor, produced results comparable to the non-linear neural network models.

Table 4. Evaluation results of the second-level model of *Method 1* using a history of 30 minutes.

Patient ID	PH: 30 min		PH: 60 min	
	RMSE	MAE	RMSE	MAE
540	21.19 ± 0.07	15.73 ± 0.09	39.41 ± 0.09	30.04 ± 0.15
544	17.40 ± 0.08	12.45 ± 0.08	30.48 ± 0.07	22.90 ± 0.08
552	16.25 ± 0.07	12.02 ± 0.05	29.32 ± 0.09	22.21 ± 0.02
567	20.40 ± 0.07	14.44 ± 0.07	36.12 ± 0.02	27.12 ± 0.07
584	21.54 ± 0.06	15.62 ± 0.06	36.27 ± 0.15	27.17 ± 0.16
596	17.17 ± 0.10	12.13 ± 0.09	28.77 ± 0.26	20.80 ± 0.17
Average	18.99 ± 0.08	13.73 ± 0.07	33.39 ± 0.12	25.04 ± 0.11

Table 5. Evaluation results of the first-level models of *Method 1* using a history of 60 minutes.

Patient ID	Model	PH: 30 min		PH: 60 min	
		RMSE	MAE	RMSE	MAE
540	PLSR	22.10	16.58	41.10	31.76
	MLP	21.58 ± 0.28	16.12 ± 0.22	40.53 ± 1.23	31.12 ± 0.91
	LSTM	**21.11 ± 0.18**	**15.56 ± 0.11**	**39.18 ± 0.37**	**30.00 ± 0.33**
544	PLSR	18.09	13.33	31.83	24.71
	MLP	18.09 ± 0.03	**13.05 ± 0.08**	32.34 ± 1.00	24.80 ± 1.76
	LSTM	**18.04 ± 0.35**	13.06 ± 0.48	**30.79 ± 0.39**	**23.15 ± 0.68**
552	PLSR	**16.79**	12.78	30.25	23.67
	MLP	17.58 ± 0.46	13.39 ± 0.70	**30.16 ± 0.43**	**22.89 ± 0.14**
	LSTM	16.97 ± 0.78	**12.59 ± 0.55**	30.69 ± 0.70	23.19 ± 0.55
567	PLSR	20.99	15.03	37.51	28.21
	MLP	21.71 ± 0.92	15.80 ± 1.06	37.34 ± 0.78	28.02 ± 0.76
	LSTM	**20.74 ± 0.50**	**14.75 ± 0.59**	**36.67 ± 0.98**	**27.52 ± 1.06**
584	PLSR	22.04	16.19	37.04	27.97
	MLP	22.10 ± 0.25	15.98 ± 0.23	37.13 ± 0.74	27.68 ± 0.89
	LSTM	**21.66 ± 0.10**	**15.63 ± 0.12**	**36.76 ± 0.46**	**27.18 ± 0.44**
596	PLSR	17.62	12.66	**29.48**	21.97
	MLP	18.05 ± 0.29	12.71 ± 0.27	29.71 ± 0.35	21.83 ± 0.21
	LSTM	**17.58 ± 0.19**	**12.55 ± 0.34**	29.55 ± 0.52	**21.63 ± 0.34**
Average	PLSR	19.60	14.43	34.53	26.38
	MLP	19.85 ± 0.37	14.51 ± 0.43	34.54 ± 0.75	26.06 ± 0.78
	LSTM	**19.35 ± 0.35**	**14.02 ± 0.36**	**33.94 ± 0.57**	**25.44 ± 0.57**

Table 4 shows the evaluation results of the second-level model of *Method 1* when a history of 30 minutes was used for training. Comparing these results with those in Table 3, the second-level model resulted in better prediction performance than all the first-level models for all patients and both prediction horizons. This means that the stacked regression technique helped improve prediction performance.

Table 5 displays the evaluation results of the first-level models of *Method 1*, when a history of 60 minutes was used for training. As results show, for both prediction horizons, LSTM had the best performance for a majority of the patients. In overall, PLSR provided the second-best results.

The evaluation results of the second-level model of *Method 1* using 60-minute history are shown in Table 6. In comparison with Table 5, it can be observed that the stacked regression technique advanced the prediction performance for all patients for this history, too. Also, in comparison with Table 4, *Method 1* had a better overall performance when it used a history of 30 minutes than a history of 60 minutes.

Table 6. Evaluation results of the second-level model of *Method 1* using a history of 60 minutes.

Patient ID	PH: 30 min		PH: 60 min	
	RMSE	MAE	RMSE	MAE
540	20.98 ± 0.13	15.50 ± 0.14	39.05 ± 0.17	29.68 ± 0.18
544	17.66 ± 0.09	12.66 ± 0.08	30.42 ± 0.36	22.82 ± 0.42
552	16.30 ± 0.09	12.04 ± 0.06	29.38 ± 0.24	22.26 ± 0.21
567	20.52 ± 0.17	14.54 ± 0.10	36.52 ± 0.10	27.31 ± 0.14
584	21.62 ± 0.17	15.63 ± 0.08	37.01 ± 0.28	27.64 ± 0.20
596	17.45 ± 0.08	12.27 ± 0.09	28.92 ± 0.27	20.92 ± 0.19
Average	19.09 ± 0.12	13.77 ± 0.09	33.55 ± 0.24	25.11 ± 0.23

4.2 Method 2

In this section, the evaluation result of *Method 2* is presented. To be concise, the results of the second-level model only are reported, which are the final predictions of the method.

Table 7 shows the evaluation results of *Method 2* using a 30-minute history. Comparing these results with those in Table 4, the prediction performance of *Method 2* was comparable with that of *Method 1* for all patients, except patient 552. This may be due to the existence of a large number of missing activity data points in this patient's data (as can be seen in Table 1).

Table 7. Evaluation results of *Method 2* using a history of 30 minutes.

Patient ID	PH: 30 min		PH: 60 min	
	RMSE	MAE	RMSE	MAE
540	21.26 ± 0.09	15.89 ± 0.07	39.48 ± 0.16	30.26 ± 0.19
544	17.59 ± 0.11	12.62 ± 0.12	30.68 ± 0.15	23.14 ± 0.20
552	19.85 ± 4.51	12.65 ± 0.46	35.70 ± 3.32	23.76 ± 0.40
567	20.52 ± 0.12	14.49 ± 0.12	36.39 ± 0.20	27.14 ± 0.19
584	21.72 ± 0.17	15.78 ± 0.10	36.53 ± 0.13	27.45 ± 0.08
596	17.24 ± 0.11	12.19 ± 0.07	28.83 ± 0.11	21.03 ± 0.13
Average	19.70 ± 0.85	13.94 ± 0.16	34.60 ± 0.68	25.46 ± 0.20

Table 8 lists the evaluation result of *Method 2* using a history of 60 minutes. Comparing these results with those in Table 6, the evaluation results for both methods were close to each other. Also, comparing these results with those in Table 7, *Method 2* made better predictions using a history of 60 minutes than a history of 30 minutes.

Table 8. Evaluation results of *Method 2* using a history of 60 minutes.

Patient ID	PH: 30 min		PH: 60 min	
	RMSE	MAE	RMSE	MAE
540	20.89 ± 0.05	15.49 ± 0.11	39.30 ± 0.35	29.80 ± 0.21
544	17.70 ± 0.14	12.68 ± 0.13	30.71 ± 0.22	23.25 ± 0.29
552	16.73 ± 0.51	12.33 ± 0.18	34.67 ± 3.51	23.47 ± 0.58
567	20.57 ± 0.14	14.63 ± 0.11	36.70 ± 0.30	27.48 ± 0.18
584	21.72 ± 0.06	15.71 ± 0.05	36.85 ± 0.09	27.69 ± 0.13
596	17.53 ± 0.21	12.26 ± 0.18	28.88 ± 0.21	21.02 ± 0.17
Average	19.19 ± 0.18	13.85 ± 0.13	34.52 ± 0.78	25.45 ± 0.26

5 SUMMARY AND CONCLUSION

This work contributes to the prediction of BGL by proposing two methodologies for data fusion of CGM and activity using stacked regression.

In the first method, the average value of activity data added to a window of CGM data was used as input to train prediction models. Initially, three base regression models consist of MLP, LSTM, and PLSR were trained. Subsequently, predictions from these base models were used as features to train a new PLSR model which then made final predictions.

In the second method, the same base regressions were trained once using windows of activity data and once using CGM data. The predictions of all trained base models were then fed as features to a new PLSR model for its training process. The new PLSR was used to make refined predictions.

The results obtained show that *Method 1* (average value of activity data added to the window of CGM data) had a slightly better performance than *Method 2* (first-level models trained twice, once with a history of CGM data, once using a history of activity data). In overall, *Method 1* using a history of 30 minutes had the best results by providing a RMSE of 18.99 and 33.39 for the prediction horizon of 30 minutes and 60 minutes, respectively.

6 SOFTWARE AND CODE

To implement the models, we used Python 3.6, TensorFlow 1.15.0 and Keras 2.2.5. Also, Pandas, NumPy and Sklearn packages of python were used. The codes were run on a commodity laptop. The codes of our implementation are available at: `https://gitlab.com/Hoda-Nemat/data-fusion-stacking.git`

REFERENCES

[1] G. S. Jeha *et al.*, "Continuous glucose monitoring and the reality of metabolic control in preschool children with type 1 diabetes," *Diabetes Care*, vol. 27, no. 12, pp. 2881–2886, 2004.

[2] L. S. Schilling *et al.*, "A new self-report measure of self-management of type 1 diabetes for adolescents," *Nurs. Res.*, vol. 58, no. 4, p. 228, 2009.

[3] E. R. Seaquist *et al.*, "Hypoglycemia and diabetes: A report of a workgroup of the american diabetes association and the endocrine society," *J. Clin. Endocrinol. Metab.*, vol. 98, no. 5, pp. 1845–1859, 2013.

[4] K. Makris and L. Spanou, "Is there a relationship between mean blood glucose and glycated hemoglobin?," *J. Diabetes Sci. Technol.*, vol. 5, no. 6, pp. 1572–1583, 2011.

[5] A. Z. Woldaregay, E. Årsand, T. Botsis, D. Albers, L. Mamykina, and G. Hartvigsen, "Data-driven blood glucose pattern classification and anomalies detection: machine-learning applications in type 1 diabetes," *J. Med. Internet Res.*, vol. 21, no. 5, p. e11030, 2019.

[6] J. Vehí, I. Contreras, S. Oviedo, L. Biagi, and A. Bertachi, "Prediction and prevention of hypoglycaemic events in type-1 diabetic patients using machine learning," *Health Informatics J.*, p. 1460458219850682, 2019.

[7] C. Berra *et al.*, "Hypoglycemia and hyperglycemia are risk factors for falls in the hospital population," *Acta Diabetol.*, vol. 56, no. 8, pp. 931–938, 2019.

[8] S. Mirshekarian, R. Bunescu, C. Marling, and F. Schwartz, "Using LSTMs to learn physiological models of blood glucose behavior," *Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. EMBS*, pp. 2887–2891, 2017.

[9] A. Bertachi, L. Biagi, I. Contreras, N. Luo, and J. Vehí, "Prediction of Blood Glucose Levels And Nocturnal Hypoglycemia Using Physiological Models and Artificial Neural Networks.," in *3rd International Workshop on Knowledge Discovery in Healthcare Data*, 2018, pp. 85–90.

[10] J. Martinsson, A. Schliep, B. Eliasson, C. Meijner, S. Persson, and O. Mogren, "Automatic blood glucose prediction with confidence using recurrent neural networks," *3rd Int. Work. Knowl. Discov. Healthc. Data*, vol. 2148, pp. 64–68, 2018.

[11] T. Zhu, K. Li, P. Herrero, J. Chen, and P. Georgiou, "A Deep Learning Algorithm for Personalized Blood Glucose Prediction.," in *3rd International Workshop on Knowledge Discovery in Healthcare Data*, 2018, pp. 64–78.

[12] J. Xie and Q. Wang, "Benchmark Machine Learning Approaches with Classical Time Series Approaches on the Blood Glucose Level Prediction Challenge.," in *3rd International Workshop on Knowledge Discovery in Healthcare Data*, 2018, pp. 97–102.

[13] M. H. Jensen, C. Dethlefsen, P. Vestergaard, and O. Hejlesen, "Prediction of Nocturnal Hypoglycemia From Continuous Glucose Monitoring Data in People With Type 1 Diabetes: A Proof-of-Concept Study," *J. Diabetes Sci. Technol.*, vol. 14, no. 2, pp. 250–256, 2020.

[14] C. Marling and R. Bunescu, "The OhioT1DM Dataset for Blood Glucose Level Prediction: Update 2020," in *5th International Workshop on Knowledge Discovery in Healthcare Data*, 2020.

[15] C. Marling and R. C. Bunescu, "The OhioT1DM Dataset For Blood Glucose Level Prediction.," in *3rd International Workshop on Knowledge Discovery in Healthcare Data*, 2018, pp. 60–63.

[16] F. Murtagh, "Multilayer perceptrons for classification and regression," *Neurocomputing*, vol. 2, no. 5–6, pp. 183–197, 1991.

[17] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[18] H. Khadem, M. R. Eissa, H. Nemat, O. Alrezj, and M. Benaissa, "Classification before regression for improving the accuracy of glucose quantification using absorption spectroscopy," *Talanta*, vol. 211, 2020.

[19] S. Wold, M. Sjöström, and L. Eriksson, "PLS-regression: a basic tool of chemometrics," *Chemom. Intell. Lab. Syst.*, vol. 58, no. 2, pp. 109–130, 2001.

[20] L. Breiman, "Stacked regressions," *Mach. Learn.*, vol. 24, no. 1, pp. 49–64, 1996.

Blood Glucose Level Prediction as Time-Series Modeling using Sequence-to-Sequence Neural Networks

Ananth Bhimireddy [1] and **Priyanshu Sinha** [2] and **Bolu Oluwalade** [3] and **Judy Wawira Gichoya** [4] and **Saptarshi Purkayastha** [5]

Abstract. The management of blood glucose levels is critical in the care of Type 1 diabetes subjects. In extremes, high or low levels of blood glucose are fatal. To avoid such adverse events, there is the development and adoption of wearable technologies that continuously monitor blood glucose and administer insulin. This technology allows subjects to easily track their blood glucose levels with early intervention, preventing the need for hospital visits. The data collected from these sensors is an excellent candidate for the application of machine learning algorithms to learn patterns and predict future values of blood glucose levels. In this study, we developed artificial neural network algorithms based on the OhioT1DM training dataset that contains data on 12 subjects. The dataset contains features such as subject identifiers, continuous glucose monitoring data obtained in 5 minutes intervals, insulin infusion rate, etc. We developed individual models, including LSTM, BiLSTM, Convolutional LSTMs, TCN, and sequence-to-sequence models. We also developed transfer learning models based on the most important features of the data, as identified by a gradient boosting algorithm. These models were evaluated on the OhioT1DM test dataset that contains 6 unique subject's data. The model with the lowest RMSE values for the 30- and 60-minutes was selected as the best performing model. Our result shows that sequence-to-sequence BiLSTM performed better than the other models. This work demonstrates the potential of artificial neural networks algorithms in the management of Type 1 diabetes.

Keywords. Blood glucose prediction, Time-series model, Wearable devices, Transfer learning

1 INTRODUCTION

Diabetes Mellitus is a chronic disease characterized by high blood glucose levels. According to the 2020 CDC National Diabetes Statistics Report, about 34.2 million people or 10.2 percent of Americans have diabetes [3]. Diabetes is classified as Type 1, Type 2 and Gestational Diabetes. Insulin is an enzyme produced in the pancreas that helps in blood glucose absorption into cells. In Type 1 diabetes (T1DM), the pancreas produces little or no insulin. In contrast, in Type 2 diabetes (T2DM), the pancreas produces a small amount of insulin, or the body is resistant to the effect of insulin. This absence or resistance to insulin leads to an increase in blood glucose levels, known as hyperglycemia. Symptoms of hyperglycemia include excessive thirst, excessive urination, sweating, etc. Diabetic ketoacidosis is a serious complication of uncontrolled hyperglycemia that can lead to death. On the other hand, elevated insulin levels in the body can cause low levels of blood glucose, a condition known as hypoglycemia. Dizziness, weakness, coma, or eventually, death can occur in uncontrolled hypoglycemia. Insulin and glucose control are critical to the management of diabetes, and hence titration of the administered insulin doses is critical in management of diabetes patients.

Glucose levels vary according to the patient's diet, and activities throughout the day. Sensors have been developed to estimate blood glucose levels at various time intervals. These sensors are useful in diabetes management because they provide longitudinal data about subjects' blood glucose and show the distinctive patterns throughout the day. These sensors are frequently coupled with the use of insulin pumps to deliver short-acting insulin continuously (basal rate) and specific insulin quantity after a meal for appropriate glycemic control. Although the sensors and insulin pumps have helped to improve patient care, patients are typically unaware of an impending adverse event of severe hyperglycemia or hypoglycemia. These adverse effects commonly occur when patients are asleep. There is an opportunity for the development of accurate prediction models using previously collected sensor data to estimate future values of blood glucose levels to prevent the occurrence of adverse events.

In this study, we utilize the OhioT1DM dataset, which contains blood glucose values of twelve T1DM subjects collected at intervals over a total time span of eight weeks [13]. These individuals had an insulin pump with continuous glucose monitoring (CGM), wearing a physical activity band and self-reporting life events using a smartphone application. CGM blood glucose data were obtained at 5-minute intervals [12]. We developed multiple models for predicting glucose values at 30 and 60 minutes in the future, using the CGM values, and mean Root Mean Square Error (RMSE) as an evaluation metric. The code for this study can be found at `https://github.com/iupui-soic/bglp2`

2 RELATED WORK

Models like LSTM and RNN have been used for forecasting in [10] which has been improved since then. The paper explores short-term load forecasting for individual electric customers and proposes an LSTM based framework to tackle the issue. Machine Learning models such as XGBoost have been used to predict glycemia in type-1 diabetic patients in [14]. This paper experiments primarily with

[1] Indiana University Purdue University Indianapolis, USA, email: anbhimi@iupui.edu
[2] Mentor Graphics India Pvt. Ltd., India, email: priyanshusinha@outlook.com
[3] Indiana University Purdue University Indianapolis, USA, email: boluwala@iupui.edu
[4] Emory University School of Medicine, USA, email: judywawira@emory.edu
[5] Indiana University Purdue University Indianapolis, USA, email: saptpurk@iupui.edu

XGBoost algorithm to predict the blood glucose levels at a 30-minute horizon in the OhioT1DM dataset. Features from pre-trained TimeNets have been used for clinical predictions in [7]. This paper uses pre-training a network for some supervised or unsupervised tasks on datasets, and then fine-tuning via transfer learning for a related end-task to leverage the resources of labeled data in making predictions. This paper points out that training deep learning models such as RNNs and LSTMs requires large labeled data and is computationally expensive.

Deep learning models like Recurrent Neural Network (RNN) have been used on the OhioTIDM dataset to predict future blood glucose values [16], including the BGLP Challenge at KDH@IJCAI-ECAI 2018 (`http://ceur-ws.org/Vol-2148/`). In some cases, these data-driven models use only the CGM values or use physiological data such as the insulin concentration, amount of carbohydrate in meals and physical activities. Chen et al. created a data-driven 3-layer dilated recurrent neural network model with a mean RMSE of 18.9, with a range of 15.2995 and 22.7104 [4]. They concluded that the missing data and the continuous fluctuations in the data influenced the model's performance. Their model bettered the Convolutional Neural Network (CNN) that gave an average RMSE of 21.726 for six subjects [21]. Bertachi et al. predicted blood glucose levels using Artificial Neural Networks with the inclusion of physiological data [1]. Their results were not significantly dissimilar to those obtained in the data-driven models. All the previous studies demonstrate that a lower RMSE is obtained at the 30-minute prediction when compared to the 60-minutes values. We postulate that a hybrid approach of combining both the data-driven and physiological models could improve on the performance of the individual models and incorporate this in our approach.

3 METHODS

3.1 Dataset Description

A detailed description of the dataset has been previously published in the OhioT1DM dataset paper [12]. We used the data provided on 12 subjects for training, and 6 test subjects. Furthermore, the parameters basal and temp basal were merged into a single parameter.

We converted both the training and testing datasets from XML to CSV, preserving the time intervals. We did not use interpolation on the datasets as rules of the competition have prohibited interpolation. We tried using the forward and backward filling to fill the null values in the datasets, but they are creating additional time intervals which becomes a problem in testing datasets. So, no re-sampling technique was used in this paper to preserve the time intervals of the samples. The data pre-processing is performed on only 6 patients (test and train datasets) whose results are to be predicted. Subject-548 has the highest number of training records (12150) and Subject-552 has the lowest no. of records(9080). The no.of features varied from subject to subject which causes unevenness while training time series models. So, we have added features to subjects as required to ease the process of training and predicting. Missing data is handled in all columns by inputting the null values with zeros(0). We did not disturb the ***glucose value*** column as required by the competition.

3.2 Description of ML models

We used the following deep learning models to predict the blood glucose levels of each subject. The data pre-processing and model development are summarized in figure 1.

3.2.1 Long Short-Term Memory (LSTM) Networks

LSTMs were originally introduced by Hochreiter and Schmidhuber [8] and later refined and popularized [17] [20]. LSTMs are a special kind of RNNs, capable of learning long-term dependencies. This quality of LSTMs helps memorize useful parts of the sequence and the model learns parameters more efficiently, making it useful for time series models.

We trained two models using LSTMs, one with 5-min interval data and another with 30-min interval data. In each model, we used all the available features at time *t* to predict the glucose value at time *t+1*. Before fitting the dataset, we scaled the dataset using MinMaxScaler from scikit-learn [15].

The LSTM model was built using the Keras [6] platform. We used 128 LSTM units, followed by a dense layer (150 units), dropout layer (0.20), dense layer (100 units), dropout layer (0.15), dense layer (50 units), dense layer (20 units) and a final layer with one unit (for prediction). We used *ReLU* as the activation function with *Adam* optimizer. The loss was calculated in *mean squared error (MSE)* and later converted into *Root Mean Squared Error (RMSE)*. The model was trained for 200 epochs with a batch size of 32. The results of the model for each subject are provided in the results table.

3.2.2 Bi-directional Long Short-Term Memory (BiLSTM) Networks

As our input text data is static, and the entire sequence is available at the same time, we implemented a BiLSTM model to observe how processing the sequence from either direction affected the accuracy. The architecture of the BiLSTM model is similar to the LSTM model and is thus useful for time series prediction.

The data processing and model parameters for BiLSTM and LSTM model were similar with an exception in the model's first layer, where the scaled data was inputted into the Bidirectional LSTM with 128 units. The model was trained for 200 epochs with a batch size of 32. The results of the model for each subject are provided in the results table.

3.2.3 Temporal Convolutional Networks (TCN)

TCNs were originally introduced by Lea and Videl [11] in 2016. TCNs are extremely useful in capturing the high-level temporal relationships in sequential networks. The TCN architecture allows capturing long-range spatio-temporal relationships. TCN's help capture the blood glucose level of subjects who usually have routine lifestyles, as TCNs can capture hierarchical relationships at low, intermediate, and high time scales.

The data processing steps are similar to the LSTM model. The TCN model was built using the Keras platform, but the depth of the model is relatively simpler compared to the LSTM and BiLSTM model. The scaled data was inputted into a TCN layer and then connected to a dense layer with one unit for the output. The model used *Adam* optimizer and *MSE* for calculating loss which was later converted to *RMSE*. The model was trained for 10 epochs and the obtained results are provided in the results table.

3.2.4 Convolutional LSTM

Convolutional LSTMs (ConvLSTM) were introduced by Xingjian Shi et al. [18] in 2015. ConvLSTMs are created by extending the fully connected LSTM to have convolutional structure in both the input-to-state and state-to-state transitions. ConvLSTM network captures

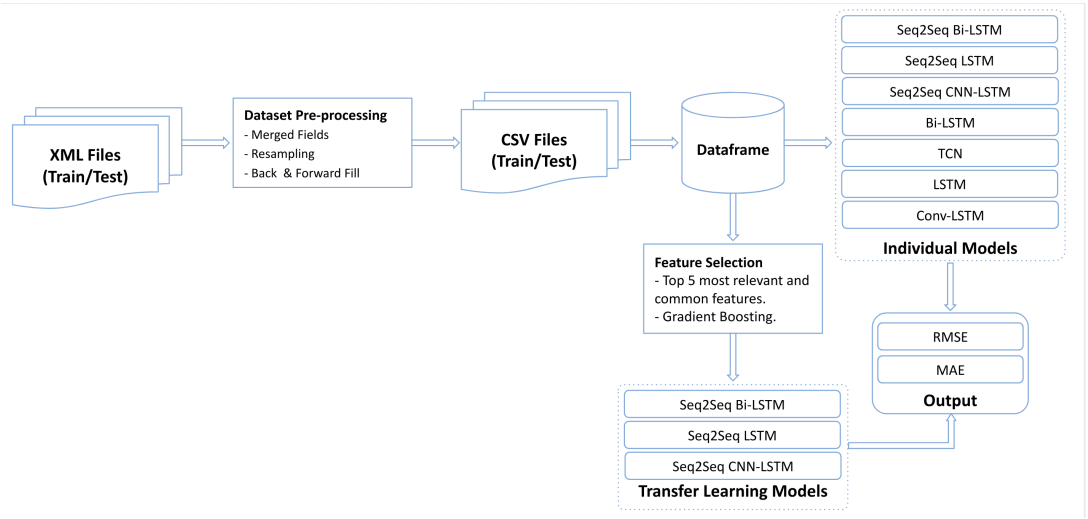


Figure 1. Process Architecture

spatio-temporal correlations better and usually outperform Fully Connected LSTM networks.

The scaled data was reshaped and inputted to a convolution layer with 32 filters of kernel size 1, followed by LSTM layer with 128 units, Dense layer with 150 units, Dropout layer with 0.2 dropout rate, Dense layer with 100 units, Dropout layer with 0.15 rate, Dense layer with 50 units, Dense layer with 16 units and finally a Dense layer with 1 unit for prediction. *ReLU* activation function was used in all layers. The model used *Adam* optimizer and the *MSE* loss was further converted to RMSE for model comparison. The model was trained for 200 epochs with a batch size of 32. The obtained results are provided in the results table.

3.2.5 Description of Sequence-to-Sequence Models

Sequence to sequence models were first introduced by Google in 2014 [19]. These models map fixed-length input with fixed-length output where length of input and output differ. Sequence-to-sequence models consist of three parts:

- **Encoder:** Encoder consists of stacks of several recurrent (LSTM) units where each unit takes a single element of the input sequence, extracts information from it and propagates it to the next unit.
- **Encoded Vector:** This is the intermediate step and the final hidden layer of the encoder. It also acts as the first hidden layer for the decoder to make predictions. This vector encapsulates all information from all input samples and provides this information to the decoder.
- **Decoder:** This consists of stacks of recurrent unit where each unit predicts output at time step t. Each unit accepts a hidden layer as input and produces output as well as its own hidden state.

The main advantage of this architecture is that it can map sequences of different lengths to each other. We applied 3 variants of sequence to sequence models viz., sequence-to-sequence LSTM, sequence-to-sequence Bi-LSTM, and sequence-to-sequence CNN-LSTM.

For training of sequence-to-sequence models, we split the data into windows of 60 minutes. This approach is intuitive and helpful as BGLP values can be predicted 1-hour ahead. It is also helpful while modelling as the model can be used to predict blood glucose values at specific time periods, let's say (after 10 minutes) or a whole sequence of blood glucose values.

To evaluate these sequence-to-sequence models we used walk forward validation. Here the model predicts the next one hour and then the actual data for one hour is given to make prediction for next one hour. See below table 1 for more illustration:

Table 1. Description of Sequence-to-Sequence input and prediction values

Input	Prediction
1st 60 minutes data	2nd 60 minutes data
[1st + 2nd] 60 minutes data	3rd 60 minutes data
[1st + 2nd + 3rd] 60 minutes data	4th 60 minutes data

For our training, we kept our input size (number of prior observations required to make next predictions) as 30 minutes data to predict next 60 minutes data. Each sequence-to-sequence model used in our work is described below:

1. **Sequence-to-Sequence LSTM**
   In this model, we used 200 LSTM cells for the encoder and decoder. This layer was followed with 2 dense layers containing 150 and 1 units wrapped in a *TimeDistributed* layer. The model was trained for 80 epochs with batch size of 40. We used *Adam* optimizer [9] with learning rate of 0.01 and loss function as *MSE*.

2. **Sequence-to-Sequence Bi-LSTM**
   In this model, we used 100 Bi-LSTM cells (LSTM cells wrapped in Bidirectional wrapper) for the encoder and decoder. This layer was followed with 2 dense layers containing 150 and 1 units wrapped in a TimeDistributed layer. The model was trained for 80 epochs with batch size of 40. Similar to sequence-to-sequence LSTM, Adam optimizer was used with 0.01 learning rate and "mean squared error" as the loss function.

3. **Sequence-to-Sequence CNN-LSTM** In this model, we used 2 1D Convolutional layer with filter size of 128 and 64 respectively. Convolutional layers were followed with MaxPooling 1D layer and flatten layer for the encoder. 200 LSTM cells are used for the decoder. This layer was followed with 2 dense layers containing 100 and 1 units wrapped in a TimeDistributed layer. The model was trained for 80 epochs with batch size of 40. Similar to

above models, Adam optimizer was used with 0.01 learning rate and "mean squared error" as the loss function.

3.2.6 Transfer Learning

For transfer learning, we first found the most relevant and common features among all 12 subjects. The importance of features was found using a ***Gradient Boosting*** algorithm. We set the cumulative frequency to 0.99 for feature selection and the 5 most important and common features are as follows: *'finger stick value', 'basal rate value', 'galvanic skin response value', 'skin temperature value', 'bolus dose value'*

Using the above-identified important features only, we trained our model on a randomly selected subject (567) and subsequently, fine-tuned each sequence-to-sequence model on each subject. The final model was used for prediction on the test data. The configurations of each model was similar to the sequence-to-sequence model as described above.

4 RESULTS

Figures 2 and 3 shows the comparison between the actual and the predicted values obtained from our best performing model i.e. sequence-to-sequence BiLSTM model. From the figures, it is clear that our model closely predicts the values of the test data by following similar peaks and troughs.

Table 2. RMSE and MAE results of Sequence-to-Sequence BiLSTM model

	RMSE		MAE	
Subjects	30 Mins	60 Mins	30 Mins	60 Mins
584	29.7	42.6	18.1	30.0
567	20.7	35.1	14.4	24.9
596	18.6	28.3	12.7	19.3
552	18.2	30.0	13.3	22.2
544	19.8	32.9	13.7	23.1
540	24.3	41.4	17.8	31.0
Mean	21.8	35.0	15.0	25.0
SD	4.0	5.4	2.1	4.2

The RMSE of the 30 minutes horizon predictions of the models are presented in tables 3, 4 and 5. From the tables, the sequence-to-sequence (Seq2Seq) models performed better than all the other models with an average RMSE of 23.5 for Seq-2-Seq LSTM, **21.8 for Seq-2-Seq BiLSTM**, and 23.0 for Seq-2-Seq CNN-LSTM. Table 2 describes the RMSE and MAE values for the sequence-to-sequence BiLSTM model which performed better than the individual and transfer learning models.

From Table 2, the value for the RMSE varies from 18.2 in subject 552 to 29.7 in subject 584 at 30 minutes, and between 28.3 in subject 596 to 42.6 in subject 584 at 60 minutes. The MAE values lies between 12.7 to 18.1 at 30 minutes and between 19.3 and 31.0 at 60 minutes.

Subject 584 in figure 3 shows more fluctuations, has a higher peak and lower troughs than subject 552 in figure 2. The effect of these variations are reflected in our results as patient 584 has the highest RMSE value while subject 552 has the lowest RMSE values. It is evident that the levels of variations in the individual subjects contributes significantly to the differences in RMSE values of the individual subjects in our model.

Table 3. RMSE values for individual models at 30 minutes Horizon

Subjects	LSTM	BiLSTM	TCN	ConvLSTM
584	27.97	29.06	26.55	26.57
567	25.65	27.04	25.71	26.04
596	19.47	20.30	18.95	21.08
552	20.62	20.30	17.14	17.73
544	21.34	22.06	80.67	20.94
540	30.21	31.72	25.94	27.35
Mean	25.0	24.4	34.7	23.2
SD	3.99	4.98	21.13	3.43

Table 4. RMSE values for sequence-to-sequence models at 30 Minutes horizon

Subjects	Seq-2-Seq LSTM	Seq-2-Seq BiLSTM	Seq-2-Seq CNN-LSTM
567	29.2	20.7	29.0
540	25.0	24.3	23.1
544	18.5	19.8	19.2
596	18.6	18.6	19.0
584	30.7	29.7	31.0
552	19.2	18.1	17.2
Mean	23.5	21.8	23.0
SD	5.0	4.0	5.2

Table 5. RMSE values for sequence to Sequence transfer learning model at 30 Minutes horizon

Subjects	Seq-2-Seq LSTM	Seq-2-Seq BiLSTM	Seq-2-Seq CNN-LSTM
567	33.5	30.8	32.6
540	39.7	32.7	44.2
544	23.1	31.5	21.8
596	19.2	18.0	17.9
584	36.8	37.3	38.1
552	21.8	14.3	17.7
Mean	29.0	27.4	28.7
SD	7.9	8.3	10.2

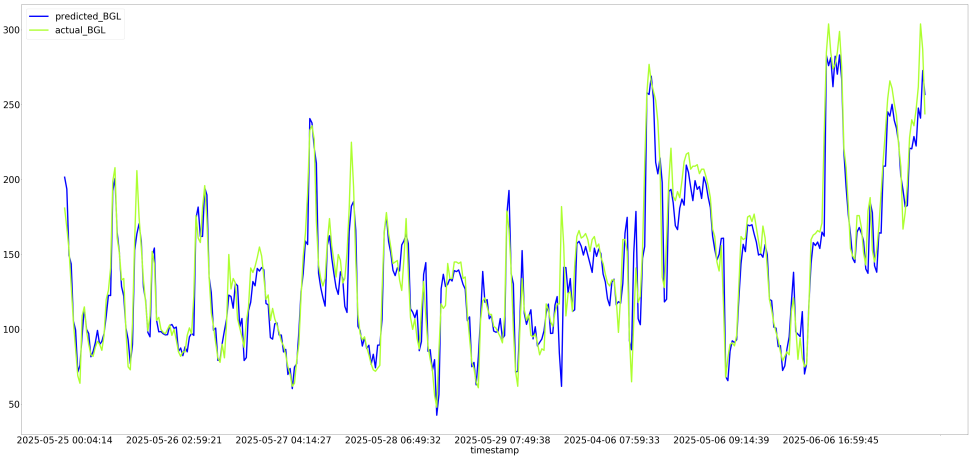


Figure 2. Actual and Predicted values of subject 552 at 30 minutes horizons

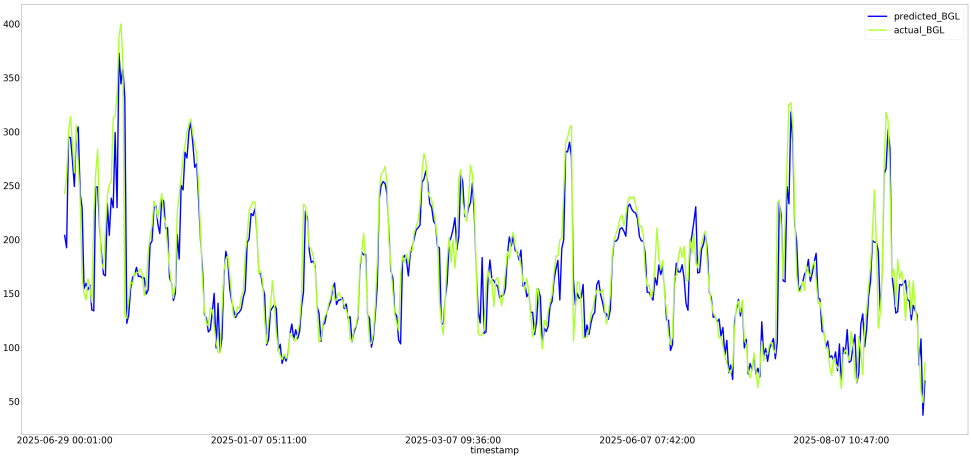


Figure 3. Actual and Predicted values of subject 584 at 30 minutes horizons

5 CONCLUSION AND FUTURE SCOPE

In this paper, we present results of application of deep learning models to make predictions of blood glucose values. Potential benefits such as the prevention of adverse events associated extreme glucose values serve as a source of motivation for these efforts. Overall, sequence-to-sequence models especially Bi-LSTM have the best performance as these models are best at mapping sequences irrespective of their lengths. Our performance is affected by fluctuations in glucose values and also with missing data as described in previous experiments. Given the overall success of transfer learning, we also evaluated the potential of single model prediction via transfer learning approach. The transfer learning approach was inferior to the sequence to sequence models.

Compared to the previous paper for BGLP Challenege, we observed that two papers [2] and [5] have better results than our proposed results. But [5] have used interpolation as a part of data processing which is against the rules of the competition and [2] did not mention details about data processing. Our future work will be to improve the transfer learning model as we are provided with more common features among all subjects, so that we can create a generic model for predicting blood glucose levels. However, the development of a generic model can be challenging because of confounding factors such as variations in sensor types, lifestyles, physiology and genetics. It is therefore pertinent that these factors are considered in future endeavors.

REFERENCES

[1] Arthur Bertachi, Lyvia Biagi, Iván Contreras, Ningsu Luo, and Josep Vehí, 'Prediction of blood glucose levels and nocturnal hypoglycemia using physiological models and artificial neural networks.', in *KHD@ IJCAI*, pp. 85–90, (2018).

[2] Arthur Bertachi, Lyvia Biagi, Iván Contreras, Ningsu Luo, and Josep Vehí, 'Prediction of blood glucose levels and nocturnal hypoglycemia using physiological models and artificial neural networks', in *KHD@IJCAI*, (2018).

[3] CDC, *National Diabetes Statistics Report 2020. Estimates of diabetes and its burden in the United States.*, 2020.

[4] Jianwei Chen, Kezhi Li, Pau Herrero, Taiyu Zhu, and Pantelis Georgiou, 'Dilated recurrent neural network for short-time prediction of glucose concentration.', in *KHD@ IJCAI*, pp. 69–73, (2018).

[5] Jianwei Chen, Kezhi Li, Pau Herrero, Taiyu Zhu, and Pantelis Georgiou, 'Dilated recurrent neural network for short-time prediction of glucose concentration', in *KHD@IJCAI*, (2018).

[6] François Chollet et al. Keras. `https://keras.io`, 2015.

[7] Priyanka Gupta, Pankaj Malhotra, Lovekesh Vig, and Gautam Shroff, 'Using features from pre-trained timenet for clinical predictions', (07 2018).

[8] Sepp Hochreiter and Jürgen Schmidhuber, 'Long short-term memory', *Neural Comput.*, **9**(8), 1735–1780, (November 1997).

[9] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014.

[10] W. Kong, Z. Y. Dong, Y. Jia, D. J. Hill, Y. Xu, and Y. Zhang, 'Short-term residential load forecasting based on lstm recurrent neural network', *IEEE Transactions on Smart Grid*, **10**(1), 841–851, (Jan 2019).

[11] Colin Lea, René Vidal, Austin Reiter, and Gregory D. Hager, 'Temporal convolutional networks: A unified approach to action segmentation', *CoRR*, **abs/1608.08242**, (2016).

[12] Cindy Marling and Razvan Bunescu, 'The ohiot1dm dataset for blood glucose level prediction: Update 2020'.

[13] Cindy Marling and Razvan C Bunescu, 'The ohiot1dm dataset for blood glucose level prediction.', in *KHD@ IJCAI*, pp. 60–63, (2018).

[14] Cooper Midroni, Peter leimbigler, Gaurav baruah, maheeder kolla, alfred whitehead, and Yan Fossat, 'Predicting glycemia in type 1 diabetes patients: Experiments with xgboost', (07 2018).

[15] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, 'Scikit-learn: Machine learning in Python', *Journal of Machine Learning Research*, **12**, 2825–2830, (2011).

[16] M Sangeetha and M Senthil Kumaran, 'Deep learning-based data imputation on time-variant data using recurrent neural network', *Soft Computing*, 1–12, (2020).

[17] Alex Sherstinsky, 'Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network', *CoRR*, **abs/1808.03314**, (2018).

[18] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo, 'Convolutional LSTM network: A machine learning approach for precipitation nowcasting', *CoRR*, **abs/1506.04214**, (2015).

[19] Ilya Sutskever, Oriol Vinyals, and Quoc V Le, 'Sequence to sequence learning with neural networks', in *Advances in neural information processing systems*, pp. 3104–3112, (2014).

[20] Peng Zhou, Zhenyu Qi, Suncong Zheng, Jiaming Xu, Hongyun Bao, and Bo Xu, 'Text classification improved by integrating bidirectional LSTM with two-dimensional max pooling', in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pp. 3485–3495, Osaka, Japan, (December 2016). The COLING 2016 Organizing Committee.

[21] Taiyu Zhu, Kezhi Li, Pau Herrero, Jianwei Chen, and Pantelis Georgiou, 'A deep learning algorithm for personalized blood glucose prediction.', in *KHD@ IJCAI*, pp. 64–78, (2018).

A Deep Learning Approach for Blood Glucose Prediction of Type 1 Diabetes

Jonas Freiburghaus and **Aïcha Rizzotti-Kaddouri** and **Fabrizio Albertetti** [1]

Abstract. An essential part of this work is to provide a data-driven model for predicting blood glucose levels that will help to warn the person with type 1 diabetes about a potential hypo- or hyperglycemic event in an easy-to-manage and discreet way. In this work, we apply a convolutional recurrent neural network on a real dataset of 6 contributors, provided by the University of Ohio [5]. Our model is capable of predicting glucose levels with high precision with a 30-minute horizon (RMSE = 17.45 [mg/dL] and MAE = 11.22 [mg/dL]), and RMSE = 33.67 [mg/dL] and MAE = 23.25 [mg/dL] for the 60-minute horizon. We believe this precision can greatly impact the long-term health condition as well as the daily management of people with type 1 diabetes.

1 INTRODUCTION

Type 1 diabetes is a disease in which the cells responsible for insulin production are destroyed. Because insulin is the hormone that triggers absorption of glucose within the cells, people with diabetes need to monitor their glucose concentration in the blood and readjust it by frequent insulin injections, following a well-defined medical protocol (e.g., once during the day and once before each meal, to keep blood sugar levels within the normal range). The main challenge in handling diabetes is the optimization of insulin injections in order to avoid hypoglycemia and hyperglycemia. This is complicated by the fact that besides insulin intake and diet, glucose levels are also affected by several other factors such as physical activity, lifestyle, mental state, stress, etc. Despite the various accomplishments made in continuous diabetes monitoring (a.k.a. continuous glucose monitoring, CGM), such methods remain invasive. Furthermore, they are only able to provide the glycemic state at a given time, when the insulin level may already be unacceptable (too high or too low). A proactive detection could therefore dramatically improve the daily handling of diabetes by the patients themselves.

This work presents an approach based on deep learning algorithms for predicting glucose levels in the future (30-minute and 60-minute horizons). Our work is based on the architecture of a recurrent neural network (CRNN) from [3] and proposes certain variants, such as multi-step predictions, regression model using blood glucose level data for each person every 5 minutes, and the inclusion of other data such as basal insulin, bolus insulin, and meal values.

The goal of using a CRNN architecture is twofold. (1) Convolutional layers act as filters and automatically learn to detect the features of interest for prediction. They are also particularly convenient for analyzing time series with little signal processing required. And (2), recurrent neural network are well-known for the capacity to learn long-term relationships between the different values. For instance, it is necessary for the network to be able to capture a correlation between the ingestion of carbohydrates now and a change in the blood glucose level in a near future.

This paper is organized as follows: Section 2 presents related work for glucose prediction, Section 3 formulates the problem of glucose prediction and our contribution, Section 4 details our methodology and discuss experimental results. Finally, in Section 5 we summarize the importance of our contribution and suggest some future work.

2 RELATED WORK

Predicting blood glucose levels for diabetes (type 1 or type 2) using machine learning has gained a lot of attention and has resulted in several methods and applications being proposed recently. They are however either based on solely measuring the glucose levels or the resulting prediction accuracy is not yet high enough to be considered as a reliable predictor of a potential critical glycemic condition. Several types of regression algorithms can be used, including SVR, classic statistical methods such as ARIMA, deep learning neural networks, or even a naive persistence algorithm, to name a few.

Gu et al. [2] propose a personalized smartphone-based non-invasive blood glucose monitoring system that detects abnormal blood glucose levels events by jointly tracking meal, drug and insulin intake, as well as physical activity and sleep quality. It automatically collects daily exercise and sleep quality, and predicts the current blood glucose level of users, together with manual records of food, drug and insulin intake. It needs re-calibration using CGM devices once every three weeks and is based on multi-division deep dynamic recurrent neural network framework. Plis et al. [6] propose a solution that uses a generic physiological model of blood glucose dynamics to generate features for a SVR model that is trained on contributor specific data. It is shown that, the model could be used to anticipate almost a quarter of hypoglycemic events 30 minutes in advance, however the demonstrated corresponding precision is 42%. Contreras et al. [1] present an alternative approach to glucose levels prediction, based on previous studies that incorporated medical knowledge into a grammar aimed to build expression for glucose that considered previous glucose values, carbohydrate intake, and insulin administration. They extend the previous research to investigate a novel and complementary approach that uses symbolic regression through grammar evolution to determine an approximate glucose levels and fluctuations using personalized blood glucose predictive models. In the same order of topic, we note the work in [4]. It also contains a comparison with a prediction using latent variable with exogenous input (LVX [9]) model, in this model bolus insulin and meal are included in the predictor matrix X but in this work only meal information is

[1] HES-SO University of Applied Sciences and Arts of Western Switzerland, email: aicha.rizzotti@he-arc.ch

added . We can see that the RMSE for a prediction horizon of 60 min for clinical data, varies from 37.02 to 35.96 [mg/dL]. In [3], they use a deep learning architecture for predicting 30-minute and 60-minute horizons on both real and simulated patients. For real patients and a 30-minute horizon, they report an RMSE of 21.07 [mg/dL]. However, only a few approaches have used deep learning algorithms for CGM on clinical data and more specifically for this dataset provided by the University of Ohio [5].

3 Glucose Prediction

The aim of this work is to predict glucose levels in advance in order to avoid situations of hyperglycemia or hypoglycemia, as well as others negative effects on the health. For instance, chronic hyperglycemia may induce fatigue and vision problems among others. For that purpose, we created a model capable of predicting the glycemia of type 1 diabetes, where values must be as accurate as possible. In this context, the metrics are the RMSE and the MAE. The smaller these values, the more reliable the model. The real gain for a patient is to be able to make decisions at any given time considering the prediction of future values, and possibly avoid glycemia-related discomforts while minimizing intrusive methods.

4 METHODOLOGY

4.1 Approach

Our approach can be summarized by the following steps:

1. Data importation
2. Data preprocessing
3. Implementation of the CRNN prediction model with multi-step forecasting
4. Training, testing and tuning on selected features
5. Delivery of the forecasted blood glucose levels

Data importation involves loading, merging, and aligning values from multiple sources under the same time scale. The selected features are basal/bolus insulin, carbohydrates, and blood glucose levels.

In the preprocessing step, all variables must have measurements carried over at the same time. This requires the use of subsampling or oversampling methods and, corollary, defining imputing methods. Linear interpolation is used for the glucose level on the training set to resample the time series at a frequency of 5 minutes. The others features may be imputed with null values when required, as their nature is sparse. We can also use domain-specific functions, such as an equation describing the absorption rate over time for carbohydrates. The preprocessing steps are summarized by :

1. Save all the blood glucose timestamps
2. Resample the features to a time delta of 1 second
3. Forward fill the missing values by using the last available values
4. Fill the left missing values with 0
5. Resample the features to a time delta of 5 minutes
6. Smooth each feature with a 1D Gaussian filter over a window containing the past 2 hours of data

We use the saved timestamps at preprocessing step 1 to generate the results at the same timestamps as the measured values.

Linear interpolation is used during the training process to impute the missing values. This allows to have more data points for the model to be trained on. However, we should note that linear interpolation is not ideal for big gaps of missing values. The interpolation is not used at test time as it could lead to a data peek. Meaning the predictions would be contaminated by future values.

The CRNN model target values are based on the following equation:

$$y_{t+L} = bg_{t+L} - bg_t, \text{for } L = 1, 2, \ldots, 12 \tag{1}$$

where bg_t is the blood glucose value at time t, L is the lag value in timesteps for the horizon, and y the label to predict, that is the differentiated value of the blood glucose level.

For instance, if the blood glucose level is 80 mg/dL at the current time and 60 mg/dL 30 minutes later, the label for a prediction horizon of 30 minutes at the current time would be -20 mg/dL.

Respectively, as the model does not predict directly the blood glucose level but only the difference from the last known value. The predicted blood glucose level is obtained with the following equation:

$$\widehat{bg}_{t+L} = bg_t + \hat{y}_{t+L}, \text{for } L = 1, 2, \ldots, 12 \tag{2}$$

where bg_t is the blood glucose value at time t, $\widehat{bg}_{t+L}$ is the predicted blood glucose level at time $t + L$, $\hat{y}_{t+L}$ is the predicted blood glucose level difference at time t with lag L, L is the lag value in timesteps for the horizon, and y the label to predict.

It is important to note that the CRNN only outputs the values $\hat{y}_{t+L}$.

The model is capable of giving a prediction for each 5 minutes prediction horizon up to 60 minutes that is 5, 10, ..., 60 minutes. This feature may give valuable information to a user and thereby improve their blood glucose level control. An example of such a prediction is given in Fig. 5.

The overall architecture of the CRNN is based on [3] and described in Fig. 1. The input signals time series are fed into a CNN for extracting relevant features. The purpose of the pooling layers is to gradually reduce the spatial dimension while keeping only the highest values included in the pooling window. Then, these features are fed into an RNN layer to model the relationships over time. Finally, a dense neural network is used as a last layer for regressing the desired target.

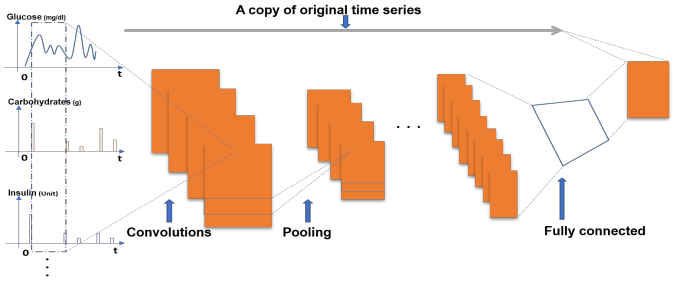


Figure 1. A multi-layer CRNN composed of convolutional layers, pooling layers, an RNN network, and a dense neural network

The training of a model is done on a contributor-basis, that is one model is trained per contributor. The reason is that each glucose response is individual, and a one-population model does not seem reasonable for CGM.

4.2 Experimental Results

4.2.1 Dataset

Our results reported in this work are based on the OhioT1DM dataset [5]. For each contributor a train set as well as test set are provided. One model only is pretrained on the data from the 6 contributors of 2018. Then for each 6 data contributor of 2020 transfer learning is applied, resulting in one trained model per contributor.

The reported results are based on the following signals: glucose level, basal insulin, bolus insulin, and meal for 6 data contributors (540, 544, 552, 567, 584, and 596). While more signals were available, we decided to use only these signals. Indeed, we performed several tests with the complete dataset and the preliminary results indicated better results with a limited set of features.

The use of a sliding window consisting of the last 2 hour data, resulting in 24 data points is based on [3]. Cross- and/or auto-correlation may provide a good starting point to find a reasonable sliding window size. Of course, the computing complexity must be taken into consideration depending on the targeted deployment hardware.

4.2.2 Architecture and learning process

The detailed architecture of the CRNN is presented in Table1.

Layer description	Output dimension
Convolution 1D	(Batch size, 24, 8)
Max pooling 1D	(Batch size, 12, 8)
Convolution 1D	(Batch size, 12, 16)
Max pooling 1D	(Batch size, 6, 16)
Convolution 1D	(Batch size, 6, 32)
Max pooling 1D	(Batch size, 3, 32)
LSTM	(Batch size, 64)
Dense	(Batch size, 256)
Dense	(Batch size, 32)
Dense	(Batch size, 12)

Table 1. Neural network layers and output shapes

The model is pretrained on batches of size 1024 over 1000 epochs, with an RMSProp optimizer. The learning rate is initially set to 0.001 and is reduced with a factor of 0.1 when the model does not progress after 3 epochs. Early stopping is used similarly with a patience of 50 epochs in order to regularize the model. The last model's weights with the lowest validation loss are then restored.

For each data contributor of 2020 the pretrained model is loaded and trained similarly as the pretraining stage. With the only difference that the learning rate is reduced with a patience of 15 epochs and that one model is saved for each contributor.

4.2.3 Results

The RMSE and the MAE are calculated for the six contributors using the following equations.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2} \tag{3}$$

where $\hat{y}_i$ is the predicted value, y_i the truth value and n the number of observation.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|\hat{y}_i - y_i| \tag{4}$$

The first observation is the errors systematically increase for each contributor over time. It is not surprising that the larger the prediction window, the larger the error in general grown up.

A comparison between contributors was also performed (Fig. 2 and Fig. 3). As we can see, for example with the contributor 596, the prediction curve for 30 minutes and 60 minutes follows the real curve with an RMSE = 13.34 and MAE = 9.08 [mg/dL], and RMSE = 27.74 and MAE = 19.13 [mg/dL]. These specific curves are also detailed in Fig. 4.

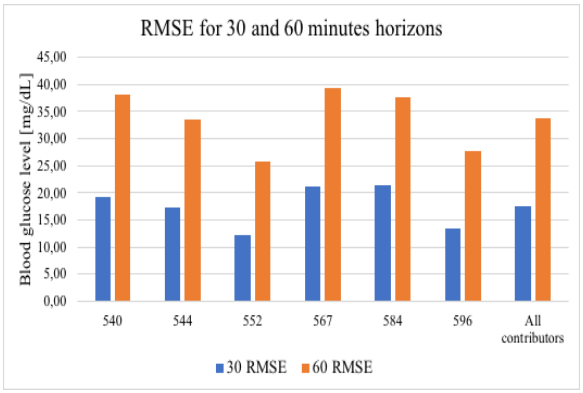


Figure 2. RMSE-30 vs RMSE-60 horizons for the 6 contributors of 2020

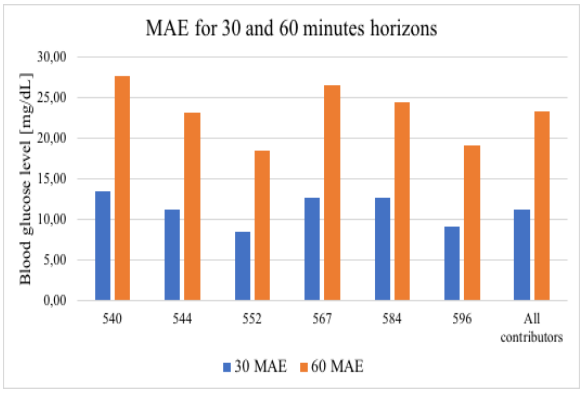


Figure 3. MAE-30 vs MAE-60 horizons for the 6 contributors of 2020

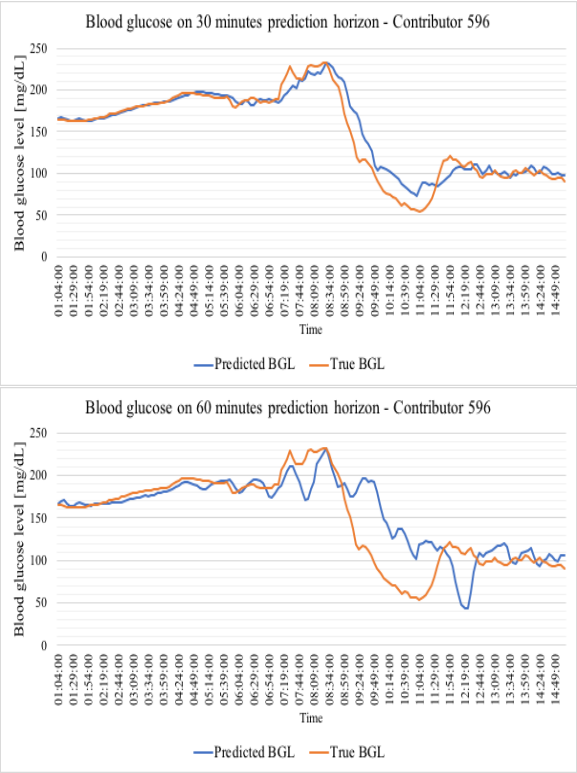


Figure 4. Example of prediction results of contributor 540, for 30 and 60 minutes

The evaluation of the prediction from the contributor-side can be performed by a multi-step prediction (as illustrated in Fig. 5). Prediction is made through one forward pass of a horizon of the last 2 hours data and outputs the horizon for the next hour represented by the orange curve. In the given example, the model seems to have predicted well the tendencies.

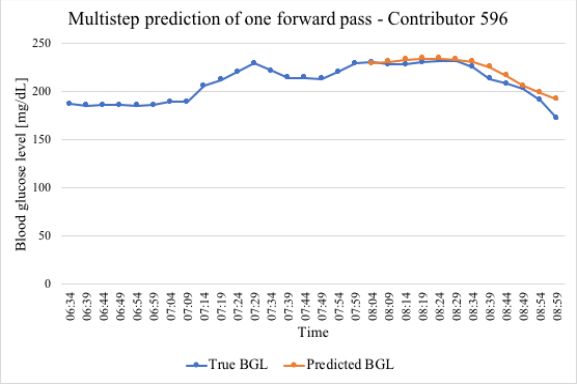


Figure 5. Multi-step prediction of one forward pass

A comparison of our approach with other algorithms is summarized in Table 2. We notice that the RMSE of our work at horizon 30 is smallest and at 60 minutes similar to [3]. Let us highlight that in (1) the database is different. The persistent algorithm is the baseline model. It forecasts the blood glucose level by using the last know value.

$$\hat{y}_{t+L} = bg_{t-1}, \text{for } L = 1, 2, \ldots, 12 \quad (5)$$

Prediction horizon	Metrics (mg/dL)	(1) Li's CRNN	(2) BASELINE	(3) CRNN
		Overall		
30	**MAE**	NA	18.13 ± 0.00	11.22
	RMSE	$21.0.7 \pm 2.35$	25.76 ± 0.00	17.45
60	**MAE**	NA	30.70 ± 0.00	23.25
	RMSE	33.27 ± 4.79	42.00 ± 0.00	33.67

Table 2. Comparison of different algorithms: (1) Li's CRNN from [3], (2) baseline with persistence forecast (where the previous value is predicted), and (3) the proposed CRNN. Let us note that (1) uses different data (not publicly available) than from (2) and (3).

With the hypothesis of a hypo-glycemia starting below 70 mg/dL and a hyper-glycemia starting above 150 mg/dL, we believe that our model delivers relevant and actionable results for real patients. In our opinion, a RMSE of 17.45 mg/dL for a 30-minute horizon indicates that data-driven decisions could be made in regard to avoiding hypo or hyper-glycemic related events.

5 CONCLUSION

In this paper we described a model for predicting future blood sugar levels of people with type 1 diabetes. A CRNN approach was proposed with the advantages of using only 4 different signals and very little signal processing. The evaluation was performed using RMSE and MAE metrics, with different horizons and on multiple contributors. The results were compared with different algorithms.The results report low error rates given the problematic of glucose prediction, and in our opinion could be considered for real-world implementation. Yet, several research tracks remain to be explored. For instance, testing additional features that can influence blood sugar levels such as stress or illness. We can also think of extracting manual features from the given signals with signal processing methods, and defining domain-specific imputation methods, such as for the absorption of carbohydrates over time. It would be also interesting to further personalize predictions as suggested in [7]. Another direction could be to use reinforcement learning approaches for the insulin recommendation, such as in [8].Those self-learning approaches are adaptable and personalize the daily insulin values to ensure glucose control, despite inter and intra-patient variability.

ERRATUM

During the making of the camera ready version, we found an error in the preprocessing stage thanks to the great reviews. The missing values were treated using a linear interpolation during the testing of the model. Thereby the predictions were contaminated by future values. This error was corrected by removing the interpolation and dealing with missing values as explained in this paper version.

Code source

It is available at https://github.com/JonasFreibur/BLGP-HES-SO

ACKNOWLEDGEMENTS

The authors would like to thank the HES-SO for funding this research.

REFERENCES

[1] Iván Contreras, Silvia Oviedo, Martina Vettoretti, Roberto Visentin, and Josep Vehí, 'Personalized blood glucose prediction: A hybrid approach using grammatical evolution and physiological models.', *PloS one 12.11*, (2017).

[2] Weixi Gu, Yuxun Zhou, Zimu Zhou, Xi Liu, Han Zou, Pei Zhang, Costas J Spanos, and Lin Zhang, 'Sugarmate: Non-intrusive blood glucose monitoring with smartphones.', *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol*, (2017).

[3] Kezhi Li, John Daniels, Chengyuan Liu, Pau Herrero-Vinas, and Panteli Georgiou, 'Convolutional recurrent neural networks for glucose prediction.', *EEE journal of biomedical and health informatics*, (2019).

[4] Chengyuan Liu, Josep Vehi, Nick Oliver, Pantelis Georgiou, and Pau Herrero, 'Enhancing blood glucose prediction with meal absorption and physical exercise information.', *arXiv preprint arXiv:1901.07467*, (2018).

[5] Razvan C. MARLING, Cindy et BUNESCU, 'The ohiot1dm dataset for blood glucose level prediction.', *KHD@ IJCAI*, (update 2020).

[6] Kevin Plis, Razvan Bunescu, Cindy Marling, Jay Shubrook, and Frank Schwartz, 'A machine learning approach to predicting blood glucose levels for diabetes management. in proceedings of aaai workshop on modern artificial intelligence for health analytics.', *AAAI Press*, 35–39, (2014).

[7] Qingnan Sun, Marko V Jankovic, João Budzinski, Brett Moore, Peter Diem, Christoph Stettler, and Stavroula G Mougiakakou, 'A dual mode adaptive basal-bolus advisor based on reinforcement learning.', *IEEE journal of biomedical and health informatics*, **23**(6), 2633–2641, (2018).

[8] Qingnan Sun, Marko V Jankovic, and Stavroula G Mougiakakou, 'Reinforcement learning-based adaptive insulin advisor for individuals with type 1 diabetes patients under multiple daily injections therapy', in *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 3609–3612. IEEE, (2019).

[9] Chunhui Zhao, Eyal Dassau, Lois Jovanovič, Howard C Zisser, Francis J Doyle III, and Dale E Seborg, 'Predicting subcutaneous glucose concentration using a latent-variable-based statistical method for type 1 diabetes mellitus', *Journal of diabetes science and technology*, **6**(3), 617–633, (2012).

Multi-Scale Long Short-Term Memory Network with Multi-Lag Structure for Blood Glucose Prediction

Tao Yang and **Ruikun Wu** and **Rui Tao** and **Shuang Wen** and **Ning Ma** and **Yuhang Zhao** and **Xia Yu** and **Hongru Li**[1]

Abstract. Accurate blood glucose (BG) prediction is necessary for daily glucose management of diabetes therapy. As glucose dynamics are often affected by various factors, such as diet, physical exercise, and insulin injection, it is difficult to consider all the relevant information and make a balance between the high-dimensional inputs and learning efficiency for a deep learning network. In this work, a novel multivariate predictor with a multi-scale long short-term memory (MS-LSTM) network was developed to automatically characterize the high-dimensional temporal dynamics and extract the features of blood glucose fluctuation and temporal trends sufficiently. Meanwhile, a multi-lag structure is designed for multiple variables, which can extract the dependence between different variables and blood glucose fluctuations more effectively. Furthermore, long-term sparse information was encoded and compressed to improve the learning efficiency of this deep learning network. The predictive capability of the proposed method was illustrated through 30-min and 60-min ahead glucose prediction in the OhioT1DM-2 Dataset. The root means square error (RMSE) values of 30-min and 60-min ahead predictions were 19.048 and 32.029, respectively, and the mean absolute error (MAE) values of 30-min and 60-min ahead predictions were 13.503 and 23.833. The results demonstrate the efficiency and prediction accuracy of the offline deep learning network, especially in the case of high-dimensional variables availability.

1 INTRODUCTION

Diabetes is a chronic disease characterized by the inability to maintain glucose homeostasis. Healthy pancreas controls the release of glucagon and insulin through α-cells and β-cells, respectively, to maintain normal blood glucose levels [7]. Type 1 diabetics cannot produce insulin normally because the β-cells are compromised, which leads to hyperglycemia and hypoglycemia [5], [17]. In recent years, advances in continuous glucose monitoring (CGM) and continuous subcutaneous insulin infusion (CSII) technologies have contributed to the closed-loop treatment of diabetes [1], [2], and [4]. The subcutaneous glucose concentration prediction algorithm has the potential to improve further the closed-loop treatment system for diabetes [8], [14], [15], and [18]. However, it is difficult to establish a multivariate physiological model to predict blood glucose precisely due to the influence of daily behaviors such as diet, physical exercise, and insulin injection [6]. Recently, some multivariate data-driven models are used to predict blood glucose levels and achieve satisfactory results. A successful case is the multivariable LSTM network proposed in paper [12], which has obtained better prediction results than the support vector regression model and diabetes experts.

[1] Northeastern University, China, email: lihongru@ise.neu.edu.cn

Nevertheless, different behaviors have different temporal effects on glucose fluctuation [3]. Using a unified lag for all variables may not be able to extract information about different characteristics sufficiently. Therefore, using multiple lags for each variable has positive implications for blood glucose prediction. An end-to-end recurrent neural network framework is proposed in paper [13], which is equipped with an adaptive input selection mechanism to improve the prediction performance of the multivariate time series. Based on this work, we develop a multi-scale LSTM (MS-LSTM) network that can capture the high-dimensional temporal dynamics and extract the features of blood glucose fluctuation and temporal trends sufficiently. Meanwhile, the multi-lag structure in the network can more effectively extract the dependence between different variables and blood glucose fluctuations. Compared with the traditional single-lag structure, using the multi-lag structure can extract more comprehensive features. Furthermore, long-term sparse information is encoded and compressed to accelerate the learning of deep networks. The MS-LSTM model was tested independently several times on the testing dataset, and the prediction results show that the model is excellent and robust.

This paper is organized as follows: section 2 explains the data preprocessing used; section 3 describes the architecture of the MS-LSTM network; section 4 illustrates model-free prediction methods in case of missing data; section 5 analyses the experimental results; section 6 summarizes the main contents from this study.

2 DATA PREPROCESSING

The variables selected for prediction included BG value, basal insulin dosage, bolus insulin dosage, carbohydrate intake, and timestamp [11]. Other variables provided were not selected for prediction, such as galvanic skin response, skin temperature, and acceleration. We used some data preprocessing methods, including aligning the original data, filling in the missing data, detecting and reconciling BG outliers, and normalizing the data. These data processing techniques will be illustrated in detail in the following sections.

2.1 Data alignment

The data in OhioT1DM-2 Dataset was collected by multiple devices, and some of the data was manually recorded by the patient, which caused the raw data to be asynchronous [16]. Therefore, the data needs to be aligned before feeding to the prediction model. Firstly, a time grid with a 5-minute sample period was derived based on the continuous glucose monitoring (CGM) data, and the missing data

was filled with zeros. Secondly, the timestamps of some insulin injections and carbohydrate intakes information cannot precisely match the timestamps of CGM data. They were reset to the timestamps of CGM data with the smallest time difference to keep the temporal correlation between the variables as much as possible [3].

2.2 CGM outlier detection and reconciliation

CGM measurements contain noise because of physical interference. Therefore, outlier detection and reconciliation are necessary to remove potential noise. Firstly, a gaussian process regression (GPR) model was trained to detect outliers of CGM measurements. The training dataset of the GPR model was the first 288 points of the training dataset. The input of the GPR model was CGM measurements from time $t-30$ to $t-5$, 6 points in total, and its output was mean ($\mu(t)$) and variance ($\sigma^2(t)$) of the CGM prediction at the time t. Then $\mu(t)$ and $\sigma^2(t)$ was used to reconcile CGM outlier at the time t as equation(1).

$$g(t) = \begin{cases} \mu(t) - 4.5\sigma^2(t) & , g(t) < \mu(t) - 4.5\sigma^2(t) \\ \mu(t) + 4.5\sigma^2(t) & , g(t) > \mu(t) + 4.5\sigma^2(t) \\ g(t) & ,others \end{cases} \quad (1)$$

where $g(t)$ is the BG level at time t.

2.3 Missing data filling

In the OhioT1DM-2 Dataset, basal insulin dosage and CGM measurements have missing data in some situations. As the basal insulin dosage has daily periodicity, it can be filled by the previous day's data. Although many methods are applied for missing CGM value filling, the accumulative error will inevitably increase as the number of filling increasing. Therefore, to degrade the accumulative error caused by data filling, the first-order Taylor series extrapolation and historical averages were weighted and summed to fill in the missing CGM values as the number of continuous missing items was less than 12. The respective methods for the missing numbers greater than or equal to 12 will be explained in detail later. It should be noted that the missing CGM values in the training dataset will not be filled to avoid additional noise.

2.4 Data normalization

Data normalization can accelerate deep network training and improve the accuracy of the model to a certain extent. We used three methods to normalize the data, and the results show that the model with coefficient normalization had the best performance. Coefficient normalization refers to only scale the amplitude of data to maintain the distribution of the raw data as much as possible [10]. The scaling of different variables was shown in Table 1.

Table 1. Scaling of different variables.

Variable	Glucose level	Timestamp	Basal	Bolus	Meal
Scaling	1/100	1/100	1/12	1	1/10

3 MS-LSTM MODEL

In this section, we will introduce the architecture of the MS-LSTM model and explain how the model is trained and tested.

3.1 Model architecture

As shown in Figure 1, the MS-LSTM model has a multi-scale hierarchy structure, which can learn the short-term and long-term dependence of blood glucose sequence. Meanwhile, the multi-lag structure can extract features on time-windows of different sizes, the features extracted on a large time-window are more abundant, and the features extracted on a small time-window are more time-sensitive. Therefore, compared with single-lag, the multi-lag structure can extract more comprehensive features and more effectively extract the dependence between different variables and blood glucose fluctuations. Theoretically, the more lags used, the more comprehensive features extracted, but correspondingly, the training time of the model will increase. Therefore, three lags were used for all variables to balance the training time and adaptability, as shown in Table 2, where PH represents the prediction horizon.

Table 2. Scale levels or lags of different variables.

	PH=30	PH=60
Variable	Scale level or lag	Scale level or lag
Blood glucose	1×7,2×7,3×7	1×9,2×9,3×9
Basal and timestamp	8,16,24	12,24,36
Bolus and timestamp	8,16,24	12,24,36
Meal and timestamp	8,16,24	10,20,30

Specifically, for predicting blood glucose after 30 minutes, the three scales adopted for the blood glucose variable were 1×7, 2×7, and 3×7, which means that all scale levels are 7, and the dilated sampling rate is 1, 2 and 3, respectively. Three lags of the basal variable were 8, 16, and 24, respectively. To ensure the unity of the output dimensions, in the multi-scale hierarchical and multi-lag structure, the number of LSTM states was equal to the minimum scale of blood glucose variable. As shown in Table 3, to sufficiently extract the useful information of various variables, the number of LSTM states in the feature fusion layer was 256. The number of nodes in the fully connected layer later was 256, 64, and 1, respectively, and some dropout layers are added between the fully connected layers to avoid the network overfitting problem.

Table 3. Detailed information of the MS-LSTM network.

		PH=30	PH=60
Structure	Layer name	Parameter	Parameter
Multi-scale hierarchical	LSTM	7 Unit	9 Unit
	LSTM	7 Unit	9 Unit
Multi-lag	LSTM	7 Unit	9 Unit
	LSTM	7 Unit	9 Unit
LSTM and Fully Connected (FC) layer	LSTM	256 Unit	256 Unit
	FC	256 Unit	256 Unit
	Dropout	0.2	0.2
	FC	64 Unit	64 Unit
	Dropout	0.1	0.1
	FC	1 Unit	1 Unit

3.2 Training and testing

The training data was divided into a training set and a verification set at a ratio of 9:1. The last 10% of the training dataset is closest to the

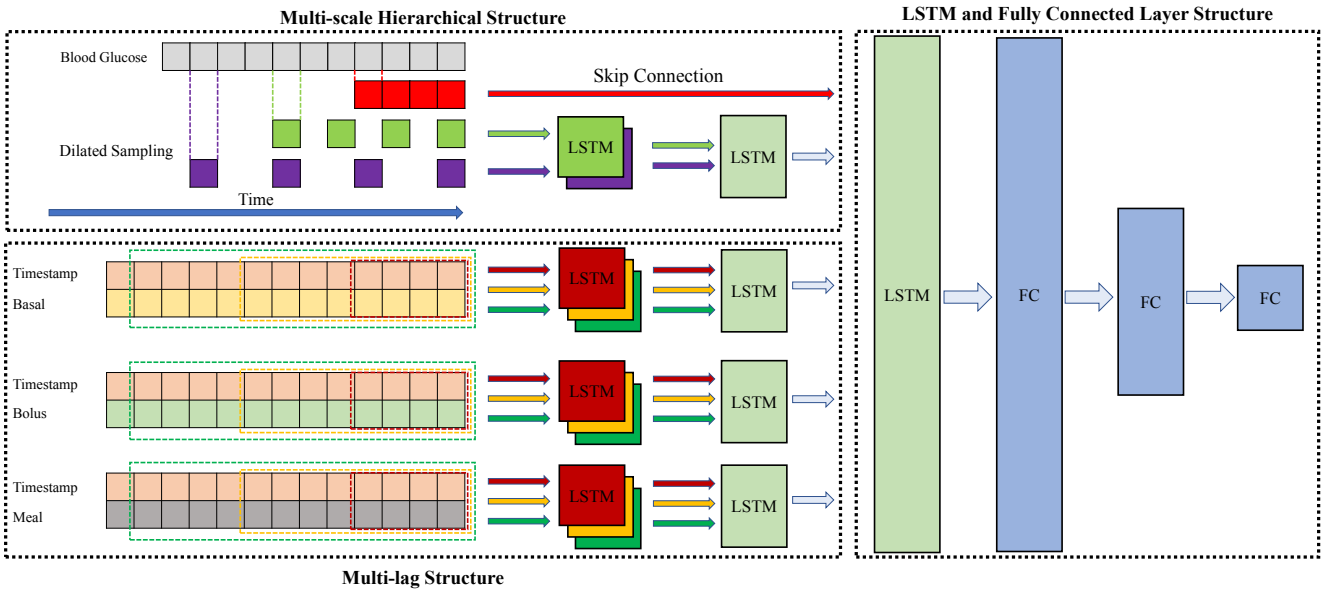


Figure 1. Block diagram representation of the MS-LSTM network

testing dataset in time, and its distribution is most similar to the testing dataset, so it was set apart as the verification set. When training the model, each iteration was evaluated on the verification set. When the model had not obtained better results after 300 consecutive evaluations, the training would be stopped, and the model which performs best on the verification set before would be saved. The training stop strategy that can effectively avoid the problem of overfitting the network is called early stopping. Because the 13th point on the test set needs to be predicted, some training data was added at the beginning of the test set to ensure that the number of prediction points meets the requirements. Besides, for several CGM data after a noticeable amount of continuously missing data, the model was not used for prediction. Instead, two model-free prediction algorithms with adaptive weight prediction and remain prediction were used to predict, respectively. Finally, the predictions were limited in the range of 40 to 400. The flow diagram is shown in Figure 2.

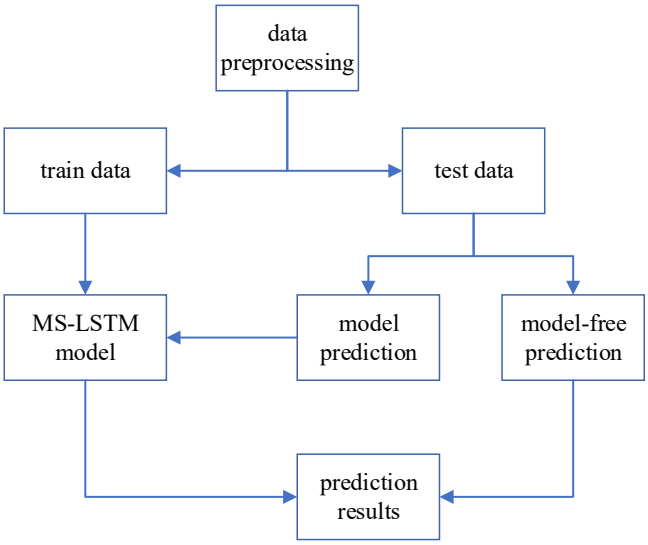


Figure 2. Flow diagram of blood glucose prediction

Training batch size: The experiment used mini-batch for weight adjustment, and the batch size of each update weight will affect the accuracy of the model. In the experiment, it was found that the larger batch size could improve the accuracy and accelerate the training process of the model, so the batch size was set to 1024.

Loss function: The experiment compared the negative log-likelihood (NLL) loss function, ε-insensitive loss function, mean absolute error (MAE) loss function, and root mean square error (RMSE) loss function. The results displayed that the model trained with the RMSE loss function had the best performance.

Optimizer: This experiment tested the root mean square prop (RMSProp) optimizer and adaptive moment estimation (Adam) optimizer [9]. The results showed that the performances of RMSProp and Adam were similar, but Adam had a significant advantage in the convergence speed. Therefore, Adam optimizer was used to update model weights, and the learning rate was set to 0.0001. In summary, the hyperparameters are shown in Table 4.

Table 4. Summary of the hyperparameters.

Hyperparameter	Value
Training batch size	1024
Optimizer	Adam optimizer
Learning rate	0.0001
Training stop strategy	early stopping
Loss function	RMSE

The experimental environment is Win10 Professional 64-bit operating system, the hardware platform is Intel Core i7 9750H processor, NVIDIA GeForce GTX 1660 Ti graphics processing unit, 16G memory notebook computer, and the development tool is Python 3.6, Keras 2.2.4, TensorFlow-GPU 1.12.0. The code used in the experiment is available on Github. In this hardware and software environment, the average training time for the MS-LSTM model was about 10 minutes.

4 MODEL-FREE PREDICTION

When the number of the missing CGM data is more than 11, the predictions of the MS-LSTM model for the following several values will cause a significant deviation. Therefore, for these CGM data, adaptive weight prediction and remain prediction are used instead of the model. The adaptive weight prediction algorithm uses short-term maintainability and long-term periodicity of blood glucose levels to make predictions. Specifically, when fewer CGM data are missing, the prediction is close to the last CGM value before the missing data, that is, depending on the short-term maintainability of blood glucose levels. On the contrary, when there are more missing data, the prediction is close to the CGM value at the same time of the previous day, that is, depending on the long-term periodicity of blood glucose levels. The process of adaptive weight prediction can be described by equation (2)-(4).

$$f = n_{miss}/(n_{miss}+c) \tag{2}$$

$$g_{av}(t) = \frac{1}{2n+1}\sum_{288-n}^{288+n} g(t - T \times n_{back}) \tag{3}$$

$$P_{aw} = (1-f)g_{last}(t) + f \times g_{av}(t)) \tag{4}$$

where n_{miss} is the number of missing data between the current prediction and the last CGM measurement before the missing data. c is a constant not less than 0, and the value in this experiment is set to 68. f is the adaptive weight factor, depend on n_{miss} and c. T is the blood glucose measurement period, the value in the OhioT1DM-2 Dataset is 5 minutes. n is a positive integer constant not less than 0, and the value in this experiment is set to 1. $n_{back} \in \{288-n, 288-n+1, ..., 288+n\}$. $g(t)$ is the BG level at time t. $g_{av}(t)$ is the average value of the CGM data of $2n+1$ points at the same time on the previous day, which represents the long-term periodicity of BG levels. $g_{last}(t)$ is the last CGM value before the missing data, which represents the short-term maintainability of BG levels. Finally, P_{aw} is the adaptive weight prediction value. As shown in Figure 3, the black points in the period from time D to F are the predictions produced by adaptive weight prediction algorithm.

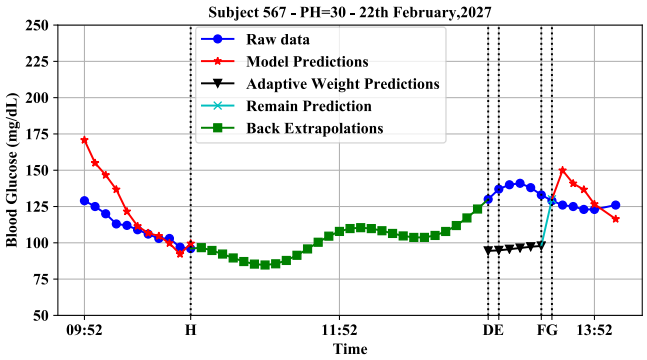


Figure 3. Prediction results of the three algorithms

When the first CGM data appears after the missing data, the value would be directly used as the predicted value of the required prediction horizon. So this algorithm is called remain prediction. As shown in the sky blue point in Figure 3, the blood glucose value at time D was the prediction value at time G.

Then, when two CGM values appeared after the missing data, as shown about the BG values at time D and E in Figure 3. Based on these two points, the reverse first-order Taylor series extrapolation was performed. Then the extrapolated data and the average historical data before the missing data were weighted and summed to ensure the smoothness of the filled data. The green points in Figure 3 were the extrapolated backward data, which were used by the MS-LSTM model to predict BG level after time G.

5 RESULTS AND ANALYSIS

The performance of the model was evaluated by the root mean square error (RMSE) and mean absolute error (MAE) between the predictions and the original test data.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2} \tag{5}$$

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|\hat{y}_i - y_i| \tag{6}$$

where $\hat{y}_i$ is the predicted BG value, y_i is the target value and N represents the size of the testing dataset. To be noted that, the extrapolated values of BG were removed when evaluating the performance of the model, which guarantees the predictions had the same number as the test data.

According to the preceding steps, the results of four independent experiments are summarized in Table 5, where SD represents the standard deviation. All subjects used the same experimental parameters, but the RMSE of each patient varied from 15 to 22. Among them, the smallest RMSE is 15.871 for patient 596, and the largest RMSE is 21.934 for patient 567. The prediction results are shown in Figure 4-5. It is worth noting that the average RMSE variance of the MS-LSTM model is only 0.061 in 30 minutes prediction horizon, which reflects the excellent robustness of the model.

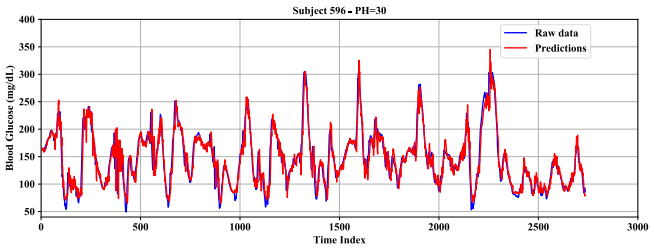


Figure 4. Blood glucose prediction results of subject 596 produced by the MS-LSTM model

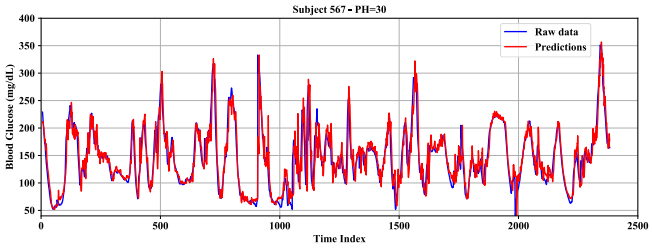


Figure 5. Blood glucose prediction results of subject 567 produced by the MS-LSTM model

The subject 567 has many consecutive spikes, which is the primary source for the prediction error. Besides, another source of prediction

Table 5. RMSE and MAE values of the MS-LSTM model for 6 subjects.

Subject	Test point	PH=30		PH=60	
		Average RMSE ± SD	Average MAE ± SD	Average RMSE ± SD	Average MAE ± SD
540	2884	20.996 ± 0.062	15.244 ± 0.051	38.219 ± 0.029	28.675 ± 0.017
544	2704	16.687 ± 0.025	11.679 ± 0.014	27.424 ± 0.100	19.522 ± 0.030
552	2352	16.918 ± 0.064	12.726 ± 0.058	30.109 ± 0.185	23.340 ± 0.320
567	2377	21.934 ± 0.039	14.698 ± 0.027	37.155 ± 0.369	27.324 ± 0.377
584	2653	21.881 ± 0.142	15.417 ± 0.127	33.913 ± 0.026	25.362 ± 0.091
596	2731	15.871 ± 0.035	11.258 ± 0.041	25.358 ± 0.227	18.777 ± 0.137
Mean		19.048 ± 0.061	13.503 ± 0.053	32.029 ± 0.156	23.833 ± 0.162

error is the missing data, as shown in the predictions after the missing data in Figure 6. Finally, a slight time delay is observed in the prediction curve, and it is also a problem for most prediction methods.

The CGM measurements contain noise because of physical interference. We used the GPR model to detect and reconcile CGM outliers to the greatest extent. However, only some severe outliers were detected and reconciled because there was no judgment standard for outliers. There are still many outliers in the raw CGM data, which is very unfavorable for the prediction model learning. Therefore, denoising CGM and obtaining high-quality data is very important to improve the performance of the prediction model.

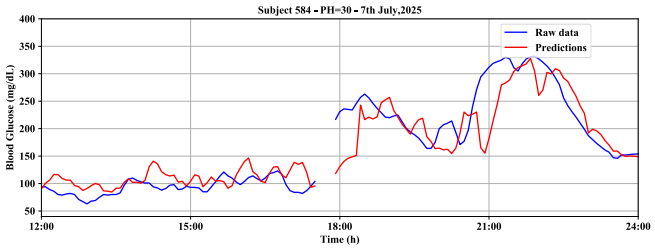


Figure 6. Prediction performance in case of missing data

6 CONCLUSION

In this paper, the MS-LSTM network is developed to adaptively characterize high-dimensional temporal dynamics and extract the long-term and short-term features of glucose fluctuation. Meanwhile, a multi-lag structure is designed for multiple variables, which can extract the dependence between different variables and blood glucose fluctuations more effectively. The long-term sparse temporal data is encoded and compressed to suitable for efficient learning with the model. The mean value of the RMSE for 6 subjects is 19.048, with standard deviation equals to 0.061 in 30-minute PH. Missing data and rapid fluctuations in blood glucose levels are the two main factors that affect the prediction performances of the model.

7 FUNDING

This research was supported by National Natural Science Foundation of China (No.61973067 and No.61903071).

REFERENCES

[1] R.M. Bergenstal, D.C. Klonoff, S.K. Garg, B.W. Bode, M. Meredith, R.H. Slover, A.J. Ahmann, S.W. Welsh, J.B. Lee, F.R. Kaufman, and AI-HS Group, 'Threshold-based insulin-pump interruption for reduction of hypoglycemia', *New England Journal of Medicine*, **369**, 224–232, (2013).

[2] B. Buckingham, F. Cameron, P. Calhoun, D.M. Maahs, D.M Wilson, H.P. Chase, B.W. Bequette, J. Lum, J. Sibayan, and R.W. Beck, 'Outpatient safety assessment of an in-home predictive low-glucose suspend system with type 1 diabetes subjects at elevated risk of nocturnal hypoglycemia', *Diabetes Technology & Therapeutics*, **15**, 622–627, (2013).

[3] J.W. Chen, K.Z. Li, P. Herrero, T.Y. Zhu, and P. Georgiou, 'Dilated recurrent neural network for short-time prediction of glucose concentration', in *CEUR Workshop Proceedings*, volume 2148, pp. 69–73, Stockholm, Sweden, (2018).

[4] P. Dua, F.J. Doyle, and E.N. Pistikopoulos, 'Multi-objective blood glucose control for type 1 diabetes', *Medical & Biological Engineering & Computing*, **47**, 343–352, (2009).

[5] A. Facchinetti, S. Favero, G. Sparacino, and C. Cobelli, 'An online failure detection method of the glucose sensor-insulin pump system: Improved overnight safety of type-1 diabetic subjects', *IEEE Transactions on Biomedical Engineering*, **60**, 406–416, (2013).

[6] E.I. Georga, V.C. Protopappas, D. Ardigo, M. Marina, I. Zavaroni, D. Polyzos, and D.I. Fotiadis, 'Multivariate prediction of subcutaneous glucose concentration in type 1 diabetes patients based on support vector regression', *IEEE Journal of Biomedical and Health Informatics*, **17**, 71–81, (2013).

[7] N.D.D. Group, 'Classification and diagnosis of diabetes mellitus and other categories of glucose intolerance', *Diabetes*, **28**, 1039–1057, (1979).

[8] C.S. Hughes, S.D. Patek, M.D. Breton, and B.P. Kovatchev, 'Hypoglycemia prevention via pump attenuation and red–yellow–green "traffic" lights using continuous glucose monitoring and insulin pump data', *Journal of Diabetes Science & Technology*, **4**, 1146–1155, (2010).

[9] D.P. Kingma and J.L. Ba, 'Adam: A method for stochastic optimization', in *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, San Diego, CA, United states, (2015).

[10] J. Martinsson, A. Schliep, B. Eliasson, C. Meijner, S. Persson, and O. Mogren, 'Automatic blood glucose prediction with confidence using recurrent neural networks', in *CEUR Workshop Proceedings*, volume 2148, pp. 64–68, Stockholm, Sweden, (2018).

[11] C. Midroni, P. J. Leimbigler, G. Baruah, M. Kolla, A.J. Whitehead, and Y. Fossat, 'Predicting glycemia in type 1 diabetes patients: Experiments with xgboost', in *CEUR Workshop Proceedings*, volume 2148, pp. 79–84, Stockholm, Sweden, (2018).

[12] S. Mirshekarian, R. Bunescu, C. Marling, and F. Schwartz, 'Using lstms to learn physiological models of blood glucose behavior', in *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 2887–2891, Jeju Island, Korea, Republic of, (2017).

[13] L. Munkhdalai, T. Munkhdalai, H.P. Kwang, T. Amarbayasgalan, E. Erdenebaatar, Hyun W.P., and Keun H.R., 'An end-to-end adaptive input selection with dynamic weights for forecasting multivariate time series', *IEEE Access*, **7**, 99099–99114, (2019).

[14] M. Phillip, T. Battelino, E. Atlas, O. Kordonouri, N. Bratina, S. Miller, T. Biester, S.M. Avbelj, I. Muller, R. Nimri, and T. Danne, 'Nocturnal glucose control with an artificial pancreas at a diabetes camp', *New England Journal of Medicine*, **368**, 824–833, (2013).

[15] J.C. Pickup, 'Insulin-pump therapy for type 1 diabetes mellitus', *New England Journal of Medicine*, **366**, 1616–1624, (2012).

[16] J.Y. Xie and Q. Wang, 'Benchmark machine learning approaches with classical time series approaches on the blood glucose level prediction challenge', in *CEUR Workshop Proceedings*, volume 2148, pp. 97–102, Stockholm, Sweden, (2018).

[17] S. Zavitsanou, A. Mantalaris, M.C. Georgiadis, and E.N. Pistikopoulos, 'In silico closed-loop control validation studies for optimal insulin delivery in type 1 diabetes', *IEEE Transactions on Biomedical Engineering*, **62**, 2369–2378, (2015).

[18] C. Zecchin, A. Facchinetti, G. Sparacino, and C. Cobelli, 'Reduction of number and duration of hypoglycemic events by glucose prediction methods: a proof-of-concept in silico study', *Diabetes Technology & Therapeutics*, **15**, 66–77, (2013).

Analysis of the performance of Genetic Programming on the Blood Glucose Level Prediction Challenge 2020

David Joedicke[1,4], Oscar Garnica [2], Gabriel Kronberger[1], J. Manuel Colmenar[3]

Stephan Winkler[4,1], J. Manuel Velasco[2], Sergio Contador[2,3], J. Ignacio Hidalgo[2]

Abstract. In this paper we present results for the Blood Glucose Level Prediction Challenge for the Ohio2020 dataset. We have used four variants of genetic programming to build white-box models for predicting 30 minutes and 60 minutes ahead. The results are compared to classical methods including multi-variate linear regression, random forests, as well as two types of ARIMA models. Notably, we have included future values of bolus and basal into some of the models because we assume that these values can be controlled. Additionally, we have used a convolution filter to smooth the information in the bolus volume feature. We find that overall tree-based GP performs well and better than multi-variate linear regression and random forest, while ARIMA models performed worst on the here analyzed data.

1 INTRODUCTION

This paper describes our contribution to the Blood Glucose Level Prediction Challenge (BGLPC) for the Ohio2020 dataset described in [15]. We present a comparison among different algorithmic techniques related to linear regression applied to this glucose prediction problem, where we highlight four of them, based on tree-based Genetic Programming (GP) [14]: GP, GP with offspring selection [1] (GP-OS); and a single-objective as well as a multi-objective variant of Grammatical Evolution[16] denoted as GE and MOGE. In addition, we present three approaches based on classical methods. In particular, we consider Random Forest [2], denoted as RF, a multivariate linear regression, denoted as LR, and two ARIMA models [18], denoted as A-0 and A-1. All the methods will be briefly described in the following section, as well as the pre-processing of data we have performed. In data pre-processing several features where derived from exising data and added to the dataset. The experimental results will be analyzed in Section 3. We use root mean squared error (RMSE) and mean absolute error (MAE) as metrics to measure the accuracy of our results. Finally, the conclusions will be drawn in Section 4.

[1] Josef Ressel Center for Symbolic Regression, Upper Austria University of Applied Sciences. Emails: david.joedicke@fh-ooe.at, Gabriel.Kronberger@fh-hagenberg.at, stephan.winkler@fh-ooe.at.

[2] Universidad Complutense de Madrid. Emails: ogarnica@ucm.es, mvelascc@ucm.es, hidalgo@ucm.es.

[3] Rey Juan Carlos University. Email: josemanuel.colmenar@urjc.es.

[4] Johannes Kepler University Linz

2 ALGORITHMIC PROPOSAL

2.1 Data pre-processing

Data pre-processing proved to be challenging in this competition as the exact rules of the competition were rather opaque especially regarding usage of future information and the difference between the online and the offline case. The main pitfalls were: (i) the set of features is different for the six data contributors, (ii) different sampling rates for features, (iii) variance in the duration between sampling values (e.g. blood glucose values are usually sampled every five minutes but not always), (iv) some missing values are encoded as zeros (e.g. zero values for skin temperature).

In the ARIMA model we only used the glucose level. For all the other models we used the following data pre-processing steps. We prepared a Python script that we used for pre-processing training as well as testing data. We used only the set features which are available for all data contributors even though we built six individual models. Correspondingly, we only used the following features: glucose level, basal, bolus type, bolus dose, galvanic skin response (gsr), and skin temperature. We used numerical encoding to encode the categorical variable bolus type. For the skin temperature we removed all zeros values. For the basal value we replaced all missing values with zeros.

We incorporated lagged variables for our models (e.g. the glucose level five minutes ago). For this, we extended our dataset with lagged features, whereby we used a maximum lag of two hours. So, for each feature we produced 24 (120 min / 5 min) additional features. Hence, we require values at multiples of five minutes. This is not the case in the provided datasets. Therefore, we first prepared a intermediate larger dataset with one row for every minute (equidistant sampling). In this dataset, we had to fill missing values for glucose level, galvanic skin response, and skin temperature. For the training data we used linear interpolation to fill these gaps, for the test data we used the last known value, since future values should not be used to predict the glucose value. Using the sub-sampled and interpolated dataset we prepared the lagged features and finally we reduced the number of rows again by keeping only rows where we have a target glucose value (either 30 or 60 minutes ahead).

In our modelling efforts for GP and GP-OS, we assume that the basal value as well as the bolus type and dose can be controlled externally. This assumes an application of the model as part of a model-predictive controller for an insulin pump, whereby the goal is to optimize the automatic administration of insulin. Therefore, we have included "future information" for the blood glucose prediction. The variables that we assume to be controlled and known are: basal, bolus type, and bolus dose. For these variables we included forward look-

ing features up to the prediction horizon (6 features for 30 minutes and 12 features for 60 minutes).

Finally, we added features for smoothed bolus dose values using a convolution process. Even though the bolus dose is administered almost instantaneously, the effect is not immediate. Instead, the underlying dynamic uptake process has a longer-lasting diminishing effect. We used a convolution function (Bateman function) to produce smoothed features for the bolus dose. For this smoothed bolus dose we also prepared lagged features (backwards and forwards) using the same scheme as described above.

2.2 Algorithms

After pre-processing the data as described above, we used machine learning methods to find models that describe future values of glucose after 30 minutes, $\hat{g}_{t+30}$ and after 60 minutes, $\hat{g}_{t+60}$, as a function of basal value (bv), bolus dose (bd), basis GSR value (gsr), basis skin temperature (sk), bolus type (bt) and glucose level (gl):

$$\hat{g}_{t+30/t+60} = f(bv(t-60...t), bd(t-60...t), ...) \quad (1)$$

We used seven different algorithms to model the function described in Equation (1). Linear Regression (LR) and Random Forest (RF) are well known methods that are used as benchmarks for our models. Additionally, we used two GP, two GE algorithms, and two ARIMA models to predict the glucose value. Next, we detail our proposals[5].

2.2.1 Genetic Programming

Symbolic regression (SR) is a specific method of regression analysis, where the model is represented as a closed-form mathematical expression [14]. A unique characteristic of SR is that the model structure does not have to be pre-specified. Instead, a SR solver (i.e. GP) automatically constructs mathematical expressions from the set of input variables (with their respective allowed time offsets) as well as mathematical operators and functions.

We use genetic programming (GP), an evolutionary technique that iteratively produces solutions for a given optimization problem. GP is specifically designed to find programs that solve given tasks; when applied to SR, these programs are formulas that are based on of mathematical operators, variables, and constants. Being an evolutionary algorithm, GP initially creates a randomly set of formulas and then, over many generations, produces new formulas by means of crossover and mutation operators. The improvement of these formulas is reached by selection operators: in each generation the parents for the new solution candidates are selected, and new individuals can be inserted into the next generation either automatically or only if they are selected by some kind of offspring selection. We used the GP implementation in HeuristicLab [6] and created models with a maximum size of 100 nodes and ten levels. We used GP in two different variants:

- Standard GP (GP): 1000 individuals, tournament selection as parent selection mechanism, elitism, termination criterion: 1000 generations.
- Offspring selection GP (OSGP): 1000 individuals, random parents selection, strict offspring selection (i.e., individuals are sent to the next generation if they are better than their parents [1]), elitism, termination criterion: maximum selection pressure 200 (i.e., as soon as the number of individuals that have to be created so that 1000 successful ones are found in one generation has reached 200000).

2.2.2 Grammatical Evolution

Grammatical Evolution (GE) [16] is a variant of GP which uses chromosomes to encode the information of the individuals (trees). In GE, a grammar is applied to perform the decoding process that generates the trees which, in this case, will be the mathematical expressions that represent prediction models of glucose values. Given that this method uses chromosomes, it allows the application of classical genetic operators such as crossover or mutation directly at the chromosome level, instead of the tree level, as happens in GP. We evaluate two GE proposals:

- Standard GE: we follow the same implementation and grammars of [12].The GE approach only considers one objective function, which will be either RSME or MAE. We present here only the results with RMSE, since they are significantly better with the parameters used.
- Multi-Objective GE (MOGE): we propose a multi-objective implementation of GE where the underlying algorithm is the well-known NSGA-II [9]. The MOGE approach considers RSME as one of the objective functions and a custom objective function called F_{CLARKE} as the second objective. F_{CLARKE} is based on the Clarke Error Grid (CEG) metric, and was defined as shown in Equation (2). In the expression, $|E|$ represents the number of points that belong to zone E of CEG, which is the most dangerous one for the patient, $|D|$ corresponds to the second most dangerous zone, D, and $|C|$ corresponds to zone C. Zone B was not included in the formula because it represents a not very dangerous zone, and A corresponds to the safe zone. A more detailed explanation of F_{CLARKE} can be found in [7].

$$F_{CLARKE} = 100 \cdot |E| + 10 \cdot |D| + |C| \quad (2)$$

Prediction models with GE use information of the previous 60 minutes while MOGE models can use data from the previous two hours. Additional configurations will be explored and presented at the workshop. In all the experiments, both GE and MOGE, we perform 10 runs with 400 individuals over 1000 generations, random initialization of the population (half-ramped) allowing a maximum number of 5 wrappings using a crossover probability of 0.7 and a mutation probability of 0.1. Executions were run on our Pancreas Model Tool described in [13]. Unlike the GP description above, with the two GE variants we only use information of the past and present. We did not use all the generated features, but only those of every 15 minutes before. So, we use historical data from 120, 105, 90, 75, 60, 45, 30 and 15 minutes ago for MOGE and 60, 45, 30 and 15 for GE. We only consider the glucose level, basal, bolus type, bolus dose, galvanic skin response, and skin temperature variables. We would like to highlight that recent papers that combines GE with other techniques, such as, data augmentation [17], random GE and bagging [11] or clustering [8] achieved better results than the GE configurations studied in this paper. We limit GE in order to follow the instructions of the Challenge.

[5] Source files are available under request at absys@ucm.es https://drive.google.com/drive/folders/1TOGvl55iR10aqRFO8GoD2v6TQD4djiCE?usp=sharing

[6] https://dev.heuristiclab.com

2.2.3 *ARIMA model*

In addition to the GP and GE models, we have also fitted two auto-regressive integrated moving average, ARIMA(p, d, q) models to estimate glucose values. Equation (3) presents the expression of an ARIMA(p, d, q) model where g_s is the actual value of the glucose and ϵ_s is the random error at sample s, respectively, while p, q, and d integers called the orders of the model. All our models only include glucose values and do not use exogenous variables such as insulin doses or carbohydrates.

$$\hat{g}_s = \sum_{i=1}^{p} \alpha_i g_{s-i} + \left(\epsilon_s + \sum_{i=1}^{q} \theta_i \epsilon_{s-i} \right) + \sum_{i=0}^{d} \phi_i s^i \qquad (3)$$

We evaluate both off-line and on-line models. The off-line models are created using the training data for each patient. We define 192 models by sweeping the three ARIMA parameters as follows. The auto-regressive order ranges in $p \in [2, 10]$, the moving average $q \in [2, 10]$, and the integrative part uses the values $d \in [0, 1]$, so that $9 \times 9 \times 2 = 192$. The basics behind the election of these ranges is that the model takes into account glucose values up to 10 samples (50 minutes) previous to the current time. The model's coefficients –up to $p + q + d$ coefficients per model– are estimated using maximum likelihood given the univariate glucose time series, g_s, on the complete training dataset for each patient. Once the 192 models have been estimated, we select two models per patient: the model with the lowest RMSE at 30-minutes horizon and the one with the lowest RMSE at 60 minutes.

Regarding on-line models, with each new glucose value in the testing dataset, the procedure defines a 4-hour time window using the last 48 samples –including the last one–, and it estimates the 192 ARIMA models over the time window using maximum likelihood. Again, the 192 models are created by sweeping the three ARIMA parameters, as stated above. Next, we select the best model. Unlike off-line models, now we cannot use future glucose values to select the model that will provide the lowest RMSE in the future. Hence, we select the current best model based on the history of the best models up to the current sample. We have evaluated four different criteria to choose the best model for 30-minutes predictions and six criteria for 60-minute predictions.

- We select the values of (p, q, d) of the model with the lowest absolute error 30 minutes ago to create the current model for 30-minutes and 60-minutes predictions. Note that given the current glucose value, we know the model with the lowest error 30 minutes ago.
- We select the values of (p, q, d) of the model with the lowest absolute error 60 minutes ago in the prediction of the current glucose to create the current model for a 60-minutes prediction.
- We select the values of (p, q, d) of the off-line model for 30-minutes and 60-minutes predictions.
- We define an "ensemble" ARIMA averaging the value of p and q for the six best models 30, 35, 40, ..., and 55 minutes ago. We use the rounded averaged values of p and q to create the current model for 30-minutes and 60-minutes predictions.
- Similar approach than the previous item, but we average the parameters of the six best models 60, 65, ..., and 85 minutes ago. We use the rounded averaged values of p and q to create the current model for a 60-minutes prediction.
- We select the model with the lowest Akaike Information Criterion (AIC) value to estimate both, 30-minutes and 60-minutes predictions. AIC is a criteria to compare models with different number of parameters and select the models with better trade-off between goodness-of-fit and the number of parameters of the model, a.k.a parsimony.

In some cases, the procedure cannot bring the best model because the parameters that provided the best estimation either 30 minutes or 60 minutes ago cannot produce a stable ARIMA model in the current time. Due to this fact, the overall best-performing criteria is to choose the current ARIMA model using the Akaike Information Criterion.

3 EXPERIMENTAL RESULTS

Table 1 presents the experimental results in terms of RMSE and MAE for all the algorithms and for both 30 and 60 minutes prediction horizons. For GP, GP-OS, and GE, predictions are obtained with the models that obtained the lowest RMSE value in the training phase after 10 runs. The remaining 9 models were not evaluated nor reported. For the MOGE, also 10 run were made in the training phase, and from all the solutions of the 10 Pareto fronts, we also selected the model with RMSE value, independently of the value of the F_{CLARKE}. Results represent the values for the predictions of this selection. GE and MOGE were run just with the configuration explained on section 2.2.2 and no parameters optimization was performed.

Regarding GP and GP-OS results on table 1 may differ from those reported in the submitted files. After analyzing the results we noticed that at the beginning and the end of the data our results are fluctuating. A few high and low predicted glucose values influence the quality of the results a lot. We decided to remove those unnatural values by more likely results (lower boundary: 40, upper boundary: 400). This procedure is only included in the results of this paper, not in the submitted files.

#P	RF	GP	GP-OS	LR	GE	MOGE	A-0	A-1
60 minutes - RMSE								
540	44,06	37,13	39,97	38,87	41.16	40.94	47,26	57,40
544	28,08	28,45	28,77	28,40	33.46	29.64	35,61	45,63
552	27,24	26,08	25,91	28,90	31.04	29.85	27,18	34,39
567	37,76	35,99	35,82	36,19	39.68	37.82	47,53	51,16
584	38,11	37,84	34,63	37,12	38.17	37.84	41,05	48,03
596	29,58	27,56	27,12	27,77	30.31	28.65	33,33	42,36
Avg.	34,14	32,18	32,04	32,88	35.64	34.12	38,66	46,49
60 minutes - MAE								
#P	**RF**	**GP**	**GP-OS**	**LR**	**GE**	**MOGE**	**A-0**	**A-1**
540	31,62	27,83	30,33	29,65	32.01	31.76	31,71	39,35
544	20,37	20,13	20,35	21,17	26.77	22.50	23,38	29,96
552	20,47	19,78	19,51	22,42	23.56	23.08	15,28	19,56
567	27,65	26,06	25,87	27,14	30.26	28.50	30,60	35,11
584	29,18	27,45	26,09	27,74	29.08	28.82	26,24	32,83
596	21,70	20,26	20,07	20,89	22.82	21.27	21,44	21,44
Avg.	25,17	23,58	23,70	24,83	27.42	25.99	24,77	29,71
30 minutes - RMSE								
#P	**RF**	**GP**	**GP-OS**	**LR**	**GE**	**MOGE**	**A-0**	**A-1**
540	27,00	21,67	22,26	22,00	23.10	22.04	31,09	41,39
544	17,96	17,83	17,46	17,54	19.20	17.62	21,49	31,82
552	17,45	17,50	20,84	19,42	17.29	16.61	16,59	22,66
567	25,61	22,16	23,03	23,56	23.31	22.17	29,66	35,59
584	25,69	24,83	25,81	27,08	22.87	22.21	27,01	36,96
596	19,90	16,76	16,85	17,68	18.58	16.96	21,23	21,23
Avg.	22,27	20,13	21,04	21,22	20.73	19.60	24,51	31,61
30 minutes - MAE								
#P	**RF**	**GP**	**GP-OS**	**LR**	**GE**	**MOGE**	**A-0**	**A-1**
540	19,19	15,82	15,89	16,22	16.26	16.36	20,17	26,88
544	12,60	12,10	12,06	12,44	13.80	12.97	13,92	19,51
552	13,30	13,13	15,38	14,55	12.33	12.44	9,41	12,30
567	17,12	14,69	15,80	16,18	16.41	14.97	18,87	23,17
584	17,71	16,63	17,72	17,54	17.00	16.64	17,06	23,28
596	14,23	11,91	12,03	12,84	13.36	12.10	13,57	13,57
Avg.	15,69	14,05	14,81	14,96	14.86	14.25	15,50	19,79

Table 1. Quality of the models created for 30 / 60 minutes predictions. For each modeling method we give error metrics (RSME, MAE) for 30 / 60 minutes predictions.

Table 2 shows the percentage of predictions on zones of the Clarke Error Grid [6] for both time horizons. Results are ordered by higher %A, then higher %B, lower %E, lower %D and lower %C. The first thing that can be said is that, in terms of CEG, 30 minutes is not very hard to predict. Most of the algorithms achieved excellent results with less than 3% of the predictions in the dangerous zones. For a prediction horizon of 60 minutes, all the machine learning techniques obtained less than 5% of dangerous predictions, and GP approaches seems to be the best option. However, a deeper analysis for statistical significance in required. First, we depict in figure 1 a

Algorithm	%A	%B	%C	%D	%E
30 minutes					
GP	88.03	10.43	0.25	1.45	0.02
GP-OS	86.40	11.97	0.17	1.54	0.02
LR	86.03	12.23	0.25	1.65	0.02
RF	85.25	12.57	0.45	2.10	0.00
MOGE	87.52	11.29	0.00	1.19	0.00
GE	86.46	12.69	0.03	0.82	0.00
A-0	84,93	13,90	0,33	0,83	0.07
A-1	77.91	19.66	1.60	0.64	0.20
60 minutes					
GP	69.90	26.56	0.26	3.32	0.03
GP-OS	69.90	26.73	0.21	3.13	0.05
LR	67.35	28.76	0.35	3.60	0.03
RF	67.57	28.73	0.30	3.61	0.00
MOGE	64.27	31.15	0.29	4.31	0.00
GE	60.82	34.66	0.29	4.24	0.00
A-0	62.25	36.08	1.66	1.66	0.36
A-1	54.37	39.53	4.50	0.97	0.63

Table 2. Average percentage of predictions on zones of the Clarke Error Grid [6] for both time horizons. Results are ordered by higher %A, then higher %B, lower %E, lower %D and lower %C.

graphical ranking (in terms of RMSE) of all the algorithms for each patient and for 30 a 60 minutes prediction horizons. Each algorithm is represented by its acronym and a different color, the closer the position to the name of id of the patient, the better, i.e the lower RMSE on test files. GP is the best for all the patients in 30 minutes and for 4 out of 6 in 60 minutes. Looking for statistical significance, the first

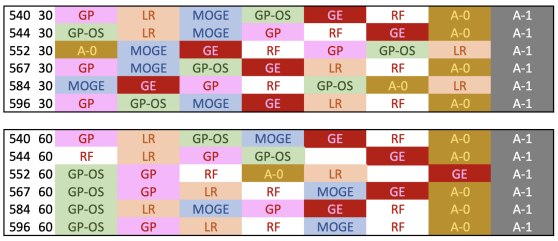


Figure 1. A graphical view of the ranking of each algorithm for each patient dataset. Clearly GP approach is the best as a general rule in terms of RMSE.

plots we created are density plots, using a kernel density estimation (KDE) of the distribution of the samples to visualize it. The objective is to visualize if the data meets the conditions for a parametric test, which is not the case. Figure 2 shows that the data is not distributed according to a Gaussian distribution and, nor the variance is the same for all the algorithms. Data distribution is multi-modal and a non-parametric test is necessary. All the plots were obtained with [4]. We use the graphical representation of the Nemenyi test [10], that compares all the algorithms pairwise. This non parametric test is based on the absolute difference of the average rankings of the predictors. For a significance level $\alpha = 0.05$ the test determines the critical difference (CD) and if the difference between the average ranking of two algorithms is grater than CD, then the null hypothe-

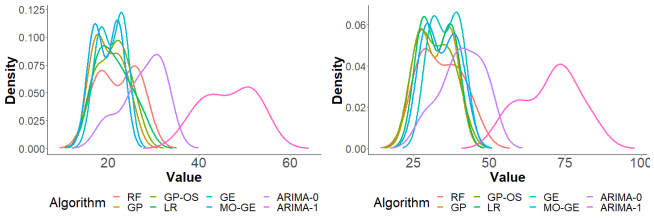


Figure 2. Density plots of the distribution of the RMSE results for all the algorithms for 30 minutes (left). The distribution are clearly multi-modal and a non parametric test is recommended. Similar plots were obtained for 60 minutes (right) and for MAE.

sis that the algorithms have the same performance is rejected. Figure 3 shows the graphical comparison where statistical differences are demonstrated to be significant. Finally we follow the Bayesian

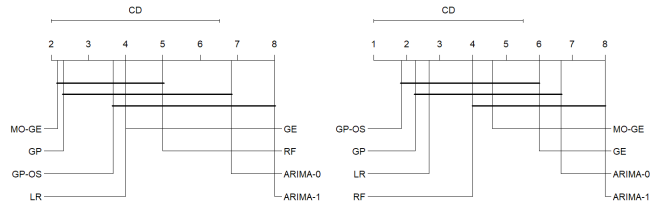


Figure 3. Nemenyi test for all the algorithms and RMSE (30 min, left, 60 min right) using the graphical representation of [10].

model of [3, 5] based on the Plackett-Luce distribution over rankings to analyse multiple algorithms in multiple problems. Figure 4 shows that GP and MOGE have the highest probability of being the best for 30 minutes, however there is not clear evidence for 60 minutes.

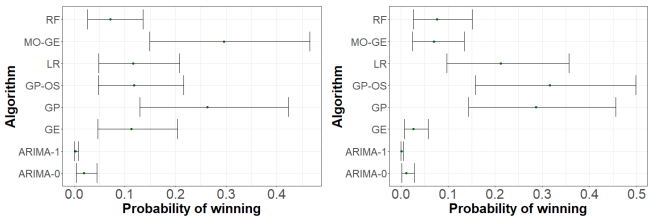


Figure 4. Bayesian model of [5] to analyse the algorithms in the set of patients and RMSE. Figure represents the probability of being the best and its standard deviation.(30 min, left, 60 min right)

4 CONCLUSION

The competition proved to be a very good test-bed for the modelling approaches as it is concerned with real-world data. The large amount of data for training proved to be challenging. For instance the ARIMA training process took several days to complete.

The decision made by the organizers of the competition to disallow usage of all future data is in our point of view not ideal. If we want to use prediction models for optimal blood glucose control it is necessary to assume that we can control the bolus and basal for

the forecasting horizon. Of course, a large amount of uncertainty remains because of unknown events in the forecasting horizon such as meals and higher activity or stress levels.

It would be interesting to try to improve the models by using all the available data for each data contributor. We only used the intersection of features available in all data sets which however limits the potential for specialization of models to individuals.

ACKNOWLEDGMENTS

This work has been also partially funded with the support of the Christian Doppler Research Association within the Josef Ressel Centre for Symbolic Regression. This work has been also partially supported by the Spanish Ministerio de Ciencia, Innovación y Universidades (MCIU/AEI/FEDER, UE) under grant ref. PGC2018-095322-B-C22; and Comunidad de Madrid y Fondos Estructurales de la Unión Europea with grant ref. P2018/TCS-4566. UCM group is supported by Spanish Ministerio de Economía y Competitividad grant RTI2018-095180-B-I00, Fundación Eugenio Rodríguez Pascual, Comunidad de Madrid grants B2017/BMD3773 (GenObIA-CM) and Y2018/NMT-4668 (Micro-Stress - MAP-CM), and structural Funds of European Union.

REFERENCES

[1] Michael Affenzeller and Stefan Wagner, 'Offspring selection: A new self-adaptive selection scheme for genetic algorithms', in *Adaptive and Natural Computing Algorithms*, 218–221, Springer, (2005).
[2] Leo Breiman, 'Random forests', *Machine Learning*, **45**, 5–32, (2001).
[3] Borja Calvo, Josu Ceberio, and Jose A Lozano, 'Bayesian inference for algorithm ranking analysis', in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pp. 324–325, (2018).
[4] Borja Calvo and Guzmán Santafé Rodrigo, 'scmamp: Statistical comparison of multiple algorithms in multiple problems', *The R Journal, Vol. 8/1, Aug. 2016*, (2016).
[5] Borja Calvo, Ofer M Shir, Josu Ceberio, Carola Doerr, Hao Wang, Thomas Bäck, and Jose A Lozano, 'Bayesian performance analysis for black-box optimization benchmarking', in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pp. 1789–1797, (2019).
[6] W.L. Clarke, D. Cox, L.A. Gonder Frederick, W. Carter, and S.L. Pohl, 'Evaluating clinical accuracy of systems for self-monitoring of blood glucose', *Diabetes Care*, **10**(5), 622–628, (September 1987).
[7] Sergio Contador, J Manuel Colmenar, Oscar Garnica, and J Ignacio Hidalgo, 'Short and medium term blood glucose prediction using multi-objective grammatical evolution', in *International Conference on the Applications of Evolutionary Computation (Part of EvoStar)*, pp. 494–509. Springer, (2020).
[8] Sergio Contador, J Ignacio Hidalgo, Oscar Garnica, J Manuel Velasco, and Juan Lanchares, 'Can clustering improve glucose forecasting with genetic programming models?', in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pp. 1829–1836, (2019).
[9] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan, 'A fast and elitist multiobjective genetic algorithm: Nsga-ii', *IEEE transactions on evolutionary computation*, **6**(2), 182–197, (2002).
[10] Janez Demšar, 'Statistical comparisons of classifiers over multiple data sets', *Journal of Machine learning research*, **7**(Jan), 1–30, (2006).
[11] J Ignacio Hidalgo, Marta Botella, J Manuel Velasco, Oscar Garnica, Carlos Cervigón, Remedios Martínez, Aranzazu Aramendi, Esther Maqueda, and Juan Lanchares, 'Glucose forecasting combining markov chain based enrichment of data, random grammatical evolution and bagging', *Applied Soft Computing*, **88**, 105923, (2020).
[12] J Ignacio Hidalgo, J Manuel Colmenar, Gabriel Kronberger, Stephan M Winkler, Oscar Garnica, and Juan Lanchares, 'Data based prediction of blood glucose concentrations using evolutionary methods', *Journal of medical systems*, **41**(9), 142, (2017).
[13] J Ignacio Hidalgo, J Manuel Colmenar, J Manuel Velasco, Gabriel Kronberger, Stephan M Winkler, Oscar Garnica, and Juan Lanchares, 'Identification of models for glucose blood values in diabetics by grammatical evolution', in *Handbook of Grammatical Evolution*, 367–393, Springer, (2018).
[14] John R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press, Cambridge, MA, USA, 1992.
[15] Cindy Marling and Razvan Bunescu, 'The ohiot1dm dataset for blood glucose level prediction: Update 2020', (2020).
[16] Michael O'Neill and Conor Ryan, 'Grammatical evolution', *IEEE Trans. Evolutionary Computation*, **5**(4), 349–358, (2001).
[17] Jose Manuel Velasco, Oscar Garnica, Juan Lanchares, Marta Botella, and J Ignacio Hidalgo, 'Combining data augmentation, edas and grammatical evolution for blood glucose forecasting', *Memetic Computing*, **10**(3), 267–277, (2018).
[18] G.Peter Zhang, 'Time series forecasting using a hybrid arima and neural network model', *Neurocomputing*, **50**, 159 – 175, (2003).

Multi-lag Stacking for Blood Glucose Level Prediction

Heydar Khadem[1] and **Hoda Nemat**[1] and **Jackie Elliott**[2] and **Mohammed Benaissa**[1]

Abstract. This work investigates blood glucose level prediction for type 1 diabetes in two horizons of 30 and 60 minutes. Initially, three conventional regression tools—partial least square regression (PLSR), multilayer perceptron, and long short-term memory—are deployed to create predictive models. They are trained once on 30 minutes and once on 60 minutes of historical data resulting in six basic models for each prediction horizon. A collection of these models are then set as base-learners to develop three stacking systems; two uni-lag and one multi-lag. One of the uni-lag systems uses the three basic models trained on 30 minutes of lag data; the other uses those trained on 60 minutes. The multi-lag system, on the other hand, leverages the basic models trained on both lags. All three stacking systems deploy a PLSR as meta-learner. The results obtained show: i) the stacking systems outperform the basic models, ii) among the stacking systems, the multi-lag shows the best predictive performance with a root mean square error of 19.01 mg/dl and 33.37 mg/dl for the prediction horizon of 30 and 60 minutes, respectively.

1 INTRODUCTION

Diabetes mellitus is a metabolic disorder and a significant cause of morbidity and mortality worldwide [1]. As yet, there is no cure developed for diabetes; and management of the corresponding life-impeding conditions is recommended as the most successful way to control the disease [6]. In fact, the occurrence of the associated complications can be suspended or even prevented by effective management of the disease [11].

Among different types of diabetes, the importance of the self-management for type 1 diabetes mellitus (T1DM) is accentuated [8, 19]. The key factor in T1DM management is to control the blood glucose level (BGL) within the normal range [2]. BGL predictive models could contribute to achieving this goal. They can help avert adverse glycaemic events by forecasting them and giving patients the chance to take corrective actions ahead of time [2].

The importance of the development of BGL predictive models in T1DM management has spurred research into this field [16, 22]. According to the knowledge requirement, predictive models can be classified as; physiological, data-driven, and hybrid models [21]. Data-driven models interpret trends in sequences of data to make estimations of future BGLs. Machine learning approaches are broadly adopted in this area [21].

Mirshekarian et al. [17] developed a model to predict blood glucose in 30-minute and 60-minute horizons using a recursive neural network (RNN) with long short- term memory (LSTM) units. The model explored BGL, insulin, food, and activity information as inputs. For the same prediction horizons, Bertachi et al. [4] and Georga et al. [9], in separate studies, proposed predictive models. Bertachi et al. applied an artificial neural network contemplating glucose, insulin, carbohydrate and physical activity as inputs for their system. BGL profile, insulin, carbohydrate intake and physical activity were inputs for a support vector regression (SVR) in the model developed by Georga et al. Investigating continuous glucose monitoring (CGM) data by recursive and direct deep learning approaches, Xie et al. [22] recommended a model for BGL prediction. Martinsson et al. [15] proposed an automatic forecast model for a prediction horizon of up to 60 minutes using RNN. The model used only the information from past BGLs as input. Bunescu et al. [7] created descriptive features to train a SVR using a physiological model of blood glucose dynamics. Carbohydrate intake, insulin administration, and the current and past BGLs were inputs of their model. Despite extensive research devoted to the development of predictive models, the performance of the proposed models remains a challenge [3].

In this work, we contributed to the improvement of BGL prediction for T1DM by applying a multi-lag stacking methodology. Initially, three conventional regression tools—partial least squares, multilayer perceptron, and long-short term memory—were applied to forecast BGLs in horizons of 30 and 60 minutes. Each tool was trained twice; once on a lag of 30 minutes and once on a lag of 60 minutes of CGM data. Therefore, six basic models were created for each prediction horizon. For each horizon, three stacking systems were then developed where predictions from a selection of the basic models were used as features to train a new regression. The first two stacking systems followed a uni-lag approach. They used predictions from the three base models trained on a history of 30 minutes and 60 minutes, respectively. The third system was multi-lag and used predictions from all six base models. The stacking systems resulted in appreciable improvements in predictive accuracy as compared to the basic predictive models. The third stacking system showed a predictive performance better than the other systems.

This is the first paper, to our knowledge, that has combined models with different time-lags to generate a multi-lag BGL prediction system.

2 DATASET

The Ohio T1DM dataset comprises several features collected from 12 individuals with type 1 diabetes in 8 weeks [14, 13]. The last ten days' worth of data for each contributor was considered as the test set. Data for a cohort of six subjects was released in 2018 for the first BGL prediction challenge [14]; data for another six subjects was released in 2020 for the second challenge [13].

In this work, the 2020's data was investigated for developing and evaluating predictive models. Among the collected features were

[1] Department of Electronic and Electrical Engineering, University of Sheffield, UK, email addresses: h.khadem@sheffield.ac.uk, hoda.nemat@sheffield.ac.uk, m.benaissa@sheffiels.ac.uk

[2] Department of Oncology and Metabolism, University of Sheffield, UK, email address: j.elliott@sheffield.ac.uk

CGM data every 5 minutes, which was the only feature explored in this work. A brief description of the CGM data in the Ohio T1DM dataset released for 2020 BGL prediction challenge is displayed in Table 1.

Table 1. Number of test and training examples of each participant in Ohio T1DM dataset released in 2020 [13].

Patient ID	Number of Training Examples	Number of Test Examples
540	11947	2896
544	10623	2716
552	9080	2364
567	10858	2389
584	12150	2665
596	10877	2743

3 METHODS

As mentioned earlier, this work proposes methodologies to predict BGL in horizons of 30 and 60 minutes. The detail of the pursued methodologies is presented in this section

3.1 Pre-processing

The first pre-processing task was taking care of missing data. Missing data in the training set was imputed applying a simple linear interpolation. Alternatively, for the test set, a linear extrapolation was employed. This was to ensure the model is not contaminated by observing future data in its pre-processing stage.

The next pre-processing step was transferring the time series forecasting problem to a supervised learning task. To this end, a rolling window consisting of a lag and future data was used as explanatory and dependent variables respectively. To give an illustration, for forecasting BGL of 30 minutes later using a history of 60 minutes, for example, we used a window with the length of 18. As a consequence of the 5-minute interval between data points, it therefore follows that the first 12 data points in the window were explanatory variables, and the rest were dependent variables.

3.2 Prediction methods

First, six basic predictive *models* were created by means of three conventional regression tools. Subsequently, employing stacking learning, three more advanced predictive *systems* were developed where a collection of the basic models were considered as base-learners and a partial least squares regression as meta-learner. All proposed models/systems were personalised to individuals.

3.2.1 Basic models

Initially, for each prediction horizon of 30 and 60 minutes, the following three conventional regressions tools were employed to generate six basic predictive models—two models by each tool. For this purpose, these tools were trained once on a history of 30 and once on a history of 60 minutes.

- *Partial least squares regression (PLSR)*
  PLSR, as a basic linear regression, holds substantial popularity in different applications due to its easy-to-apply nature and minimal computation time requirement. In a previous work, we applied PLSR for glucose quantification which provided promising results [12].
  In this work, PLSR was used as one of the regression tools. For the number of components, different values ranging from 1 to the length of the input variable were tried. Each time, the predicted residual sum of squares ($PRESS$) was calculated as follows. The number of components (A) resulting in the minimum value for $PRESS/(N-A-1)$ was then selected [20].

$$PRESS = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 \quad (1)$$

  where, N is the size of the evaluation set , y_i is reference value, and $\hat{y}_i$ is predicted value.

- *Multilayer perceptron (MLP)*
  An MLP [18] with an architecture of one hidden layer including 100 nodes and an output layer was implemented. ReLU was used as the activation function for he hidden layer, Adam as the optimiser, and mean absolute error as the loss function. Learning rate was 0.01, and the training process was based on 100 epochs.

- *Long short-term memory (LSTM)*
  We used a Vanila LSTM [10] composed of a single hidden LSTM layer with 200 nodes, a fully connected layer with 100 nodes, and an output layer. ReLU was the activation function for both hidden layers, mean squared error was the loss function, and Adam was the optimizer. The model trained on 100 epochs with a learning rate of 0.01.

3.2.2 Stacking systems

Ensemble learning is a machine learning technique that combines decisions from several models to create a new model. Stacking (Figure 1) is an ensemble approach that uses predictions from multiple base-learners (first level models) as features to train a meta-learner (second level model). This meta-learner then makes the final predictions on the test set [23].

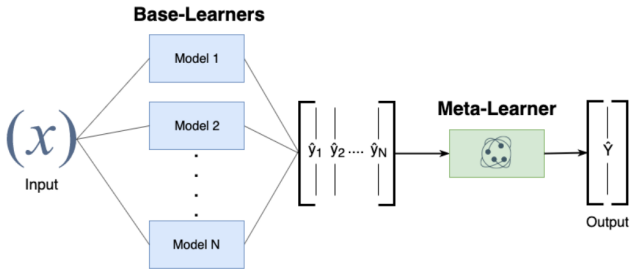


Figure 1. A stacking system uses predictions from multiple base-learners as features to train a meta-learner[5] .

In this paper, for each prediction horizon of 30 and 60 minutes, three stacking systems comprised of two uni-lag and one multi-lag were developed.

- *System 1*
  The three basic models trained on a history of 30 minutes were the base-learners of this uni-lag system and a PLSR was its meta-learner.

- *System 2*
  This system was also uni-lag. It was similar to system 1, except it used the three basic models trained on a history of 60 minutes in place of 30 minutes as base-learners.

- *System 3*
  In this multi-lag system, all the six basic models were considered as the base-learners and again a PLSR was the meta-learner. By performing a multi-lag approach the idea was to help capture a broader frequency range of BGL dynamics.

3.3 Evaluation

The test set was held out, and the train set was used to create the predictive models/systems. The developed models/systems were then utilised to predict the test data. The set of evaluation points starts 60 minutes after the beginning of the test set. First evaluation points would be otherwise similar to the training data, and it can affect the reliability of the results. Hence, the number of evaluated points for each patient is 12 less than the number of test examples mentioned in Table 1. Root mean square error (RMSE) and mean absolute error (MAE) were calculated as follows and then used as evaluation metrics.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(y_i - \hat{y_i})^2}{N}} \tag{2}$$

$$MAE = \frac{\sum_{i=1}^{N}|y_i - \hat{y_i}|}{N} \tag{3}$$

where, N, y_i , and $\hat{y_i}$ carry the same definition as in (1).

4 RESULTS AND DISCUSSION

This section presents the evaluation results for both the basic models and stacking systems. Models/systems with a performance depended on random initialization ran five times, and corresponding results have been reported in the form of mean and standard deviation. Extrapolated points were excluded when calculating the evaluation metrics. All models were built to predict future BGLs up to the end of the intended prediction horizon, but only the evaluation results for the horizon of interest are reported.

4.1 Prediction horizon of 30 minutes

4.1.1 Basic models

The results of the RMSE and MAE of the basic predictive models for the prediction horizon of 30 minutes are displayed in Table 2.

Based on the average of RMSE and MAE for all patients, *LSTM* trained on a history of 30 minutes showed the best performance among the basic models. *PLSR* with 60-minute lag was the second-best model. All models had satisfactory standard deviations.

LSTM yielded the best overall predictive accuracy among the three regression tools. However, the results of the other two tools were also comparable to that of *LSTM*. It is worth remarking that *PLSR*, as a linear regression tool, was able to generate results comparable to that of *LSTM* and even better than that of *MLP*.

Among all patients, patient 552 had the best overall evaluation results. The worst results, on the other hand, belonged to patients 584 and 540.

Table 2. Evaluation results of the basic predictive models for a 30-minute prediction horizon.

Patient ID	Basic Model	History (min)	RMSE (mg/dl)	MAE (mg/dl)
540	PLSR	30	22.11	16.58
		60	22.07	16.56
	MLP	30	21.98 ± 0.48	16.52 ± 0.33
		60	22.52 ± 0.78	16.76 ± 0.62
	LSTM	30	21.65 ± 0.28	16.06 ± 0.12
		60	**21.58 ±0.67**	**16.20 ± 0.61**
544	PLSR	30	18.08	13.34
		60	18.09	13.33
	MLP	30	18.22 ± 0.18	13.38 ± 0.37
		60	18.25 ± 0.28	13.21 ± 0.35
	LSTM	30	**17.63 ± 0.15**	**12.63 ± 0.10**
		60	18.42 ± 0.60	13.36 ± 0.44
552	PLSR	30	16.76	12.76
		60	16.79	12.78
	MLP	30	17.08 ± 0.36	12.91 ± 0.40
		60	17.03 ± 0.34	12.77 ± 0.17
	LSTM	30	**16.49 ± 0.10**	**12.29 ± 0.24**
		60	17.06 ± 0.70	12.88 ± 0.51
567	PLSR	30	20.98	15.12
		60	21.00	15.07
	MLP	30	21.24 ± 0.70	15.42 ± 0.76
		60	21.10 ± 0.46	15.13 ± 0.58
	LSTM	30	**20.66 ± 0.16**	**14.79 ± 0.25**
		60	20.77 ± 0.36	14.72 ± 0.40
584	PLSR	30	22.00	16.15
		60	21.97	16.12
	MLP	30	**21.67 ± 0.18**	**15.63 ± 0.16**
		60	22.43 ± 0.48	16.35 ± 0.61
	LSTM	30	22.23 ± 0.70	16.33 ± 0.67
		60	22.04 ± 0.22	16.11 ± 0.28
596	PLSR	30	17.79	12.77
		60	**17.62**	**12.67**
	MLP	30	17.74 ± 0.04	12.55 ± 0.05
		60	18.44 ± 0.26	13.49 ± 0.42
	LSTM	30	17.76 ± 0.67	12.74 ± 0.55
		60	17.71 ± 0.28	12.50 ± 0.33
Average	PLSR	30	19.62	14.45
		60	19.59	14.42
	MLP	30	19.65 ± 0.32	14.40 ± 0.35
		60	19.96 ± 0.43	14.62 ± 0.46
	LSTM	30	**19.40 ± 0.34**	**14.14 ± 0.32**
		60	19.60 ± 0.47	14.30 ± 0.43

4.1.2 *Stacking systems*

Table 3 shows the evaluation results of the stacking systems for a prediction horizon of 30 minutes. For all patients, the performance of the stacking systems surpassed that of the basic models. *System 3* proposed the best predictions overall based on average RMSE and MAE values. This system resulted in the best predictive accuracy for all patients except patient 544 and 584. All systems possessed small standard deviation values. The best result among all patients belonged to patient 552. The worst results, on the other hand, were those of patients 584, 540, and 567.

Table 3. Evaluation results of the stacking systems for a 30-minute prediction horizon.

Patient ID	Stacking System	RMSE (mg/dl)	MAE (mg/dl)
540	System 1	21.13 ± 0.08	15.72 ± 0.10
	System 2	21.11 ± 0.18	15.69 ± 0.14
	System 3	**20.93 ± 0.11**	**15.52 ± 0.13**
544	System 1	**17.47 ± 0.05**	**12.50 ± 0.05**
	System 2	17.92 ± 0.10	12.93 ± 0.08
	System 3	17.52 ± 0.05	12.50 ± 0.07
552	System 1	16.29 ± 0.06	12.13 ± 0.06
	System 2	16.43 ± 0.12	12.33 ± 0.16
	System 3	**16.21 ± 0.09**	**12.08 ± 0.08**
567	System 1	20.43 ± 0.07	14.47 ± 0.06
	System 2	20.51 ± 0.14	14.51 ± 0.16
	System 3	**20.43 ± 0.06**	**14.41 ± 0.06**
584	System 1	**21.61 ± 0.06**	**15.68 ± 0.04**
	System 2	21.83 ± 0.14	15.86 ± 0.08
	System 3	21.75 ± 0.08	15.76 ± 0.07
596	System 1	17.26 ± 0.03	12.19 ± 0.03
	System 2	17.47 ± 0.15	12.25 ± 0.11
	System 3	**17.22 ± 0.10**	**12.09 ± 0.04**
Average	System 1	19.03 ± 0.06	13.78 ± 0.06
	System 2	19.21 ± 0.14	13.93 ± 0.12
	System 3	**19.01 ± 0.08**	**13.73 ± 0.07**

4.2 Prediction horizon of 60 minutes

4.2.1 *Basic models*

Table 4 lists RMSE and MAE of the basic models for 60-minute prediction horizon. Among all models, *LSTM* trained on a lag of 30 minutes showed the best performance. *MLP* trained on 300 minutes was the second high-performance model. The value of standard deviation for all models were satisfactory. Among the implemented regression tools, *LSTM* resulted in the highest overall prediction accuracy. *PLSR* produced acceptable results in this case too. Data for patients 596 and 552 showed the highest overall predictability. In, contrast, patients 540, 567, and 584 had the lowest predictable data.

4.2.2 *Stacking systems*

Evaluation results of the stacking systems for a prediction horizon of 60 minutes are displayed in Table 5. *System 3* proposed the best overall predictions based on average RMSE and MAE values. The best result among all patients belonged to patient 596. All systems had low values of standard deviation.

Table 4. Evaluation results of the basic predictive models for a 60-minute prediction horizon.

Patient ID	Basic Model	History (min)	RMSE (mg/dl)	MAE (mg/dl)
540	PLSR	30	41.03	31.68
		60	41.03	31.70
	MLP	30	40.20 ± 0.38	30.90 ± 0.21
		60	41.94 ± 2.18	32.14 ± 1.53
	LSTM	30	40.36 ± 0.91	30.80 ± 0.64
		60	**39.65 ± 1.16**	**30.28 ± 0.84**
544	PLSR	30	31.80	24.71
		60	31.83	24.71
	MLP	30	31.58 ± 0.53	24.19 ± 0.99
		60	32.15 ± 0.63	24.13 ± 0.83
	LSTM	30	**30.61 ± 0.19**	**22.97 ± 0.26**
		60	31.79 ± 0.31	24.57 ± 0.73
552	PLSR	30	30.23	23.67
		60	30.24	23.68
	MLP	30	30.14 ± 0.09	23.27 ± 0.24
		60	30.59 ± 1.01	23.65 ± 0.63
	LSTM	30	**29.84 ± 0.25**	**22.52 ± 0.29**
		60	31.36 ± 1.43	23.72 ± 1.77
567	PLSR	30	37.47	28.28
		60	37.53	28.24
	MLP	30	36.81 ± 0.28	27.52 ± 0.50
		60	37.73 ± 1.28	28.57 ± 1.35
	LSTM	30	**36.56 ± 0.17**	**27.58 ± 0.28**
		60	37.17 ± 0.58	27.90 ± 0.72
584	PLSR	30	36.71	27.65
		60	36.84	27.75
	MLP	30	**36.32 ± 0.59**	**26.95 ± 0.66**
		60	37.35 ± 0.82	27.82 ± 0.92
	LSTM	30	37.14 ± 0.98	28.03 ± 1.14
		60	37.03 ± 0.99	27.42 ± 0.54
596	PLSR	30	29.63	22.05
		60	29.48	21.97
	MLP	30	29.68 ± 0.27	21.87 ± 0.31
		60	29.97 ± 0.39	22.08 ± 0.39
	LSTM	30	**28.98 ± 0.29**	**21.14 ± 0.19**
		60	29.71 ± 0.72	22.09 ± 0.80
Average	PLSR	30	34.48	26.43
		60	34.55	26.34
	MLP	30	34.12 ± 0.36	25.78 ± 0.49
		60	34.95 ± 1.05	26.40 ± 0.94
	LSTM	30	**33.92 ± 0.47**	**25.51 ± 0.47**
		60	34.45 ± 0.86	26.00 ± 0.90

Table 5. Evaluation results of the stacking systems for a 60-minute prediction horizon.

Patient ID	Stacking System	RMSE (mg/dl)	MAE (mg/dl)
540	System 1	39.47 ± 0.17	30.10 ± 0.17
	System 2	39.14 ± 0.28	29.76 ± 0.20
	System 3	**39.00 ± 0.20**	**29.65 ± 0.12**
544	System 1	**30.47 ± 0.10**	**22.92 ± 0.13**
	System 2	31.12 ± 0.12	23.72 ± 0.14
	System 3	30.54 ± 0.09	22.95 ± 0.17
552	System 1	29.39 ± 0.15	22.39 ± 0.13
	System 2	29.38 ± 0.20	22.46 ± 0.20
	System 3	**29.10 ± 0.13**	**22.10 ± 0.14**
567	System 1	**36.11 ± 0.11**	**27.08 ± 0.15**
	System 2	36.54 ± 0.14	27.36 ± 0.14
	System 3	36.31 ± 0.14	27.09 ± 0.08
584	System 1	**36.15 ± 0.16**	**27.04 ± 0.18**
	System 2	36.68 ± 0.19	27.43 ± 0.19
	System 3	36.52 ± 0.10	27.30 ± 0.14
596	System 1	**28.74 ± 0.16**	20.84 ± 0.12
	System 2	29.06 ± 0.21	21.13 ± 0.27
	System 3	28.75 ± 0.10	**20.78 ± 0.05**
Average	System 1	33.39 ± 0.14	25.06 ± 0.15
	System 2	33.65 ± 0.19	25.31 ± 0.19
	System 3	**33.37 ± 0.13**	**24.98 ± 0.12**

5 CONCLUSION

BGL prediction improved using stacking learning concepts. Initially, a time series problem was translated into a supervised learning task. Three conventional regression tools were trained with on different history length of 30 and 60 minutes, resulting in six basic predictive models. Predictions from the basic models trained with a history of 30 minutes were fed as features to a regression to build a combined learner. The learner was then used to make final predictions on the test set. The same scenario was repeated using the basic models trained on 60-minute lag observations. In both cases, the combined learner was able to make more accurate predictions on the test set. The overall performance further improved when predictions from all basic models—trained on both histories of 30 and 60 minutes—were considered as features to train a new learner.

6 SOFTWARE AND CODE

For data analysis we used Python 3.6, TensorFlow 1.15.0 and Keras 2.2.5. Pandas, NumPy and Sklearn packages of python were used. The codes are available at: `https://gitlab.com/Heydar-Khadem/multi-lag-stacking.git`

REFERENCES

[1] Florencia Aguiree, Alex Brown, Nam Ho Cho, Gisela Dahlquist, Sheree Dodd, Trisha Dunning, Michael Hirst, Christopher Hwang, Dianna Magliano, Chris Patterson, et al., 'Idf diabetes atlas', (2013).

[2] Ramzi Ajjan, David Slattery, and Eugene Wright, 'Continuous glucose monitoring: A brief review for primary care practitioners', *Advances in therapy*, **36**(3), 579–596, (2019).

[3] Muhammad Asad and Usman Qamar, 'A review of continuous blood glucose monitoring and prediction of blood glucose level for diabetes type 1 patient in different prediction horizons (ph) using artificial neural network (ann)', in *Proceedings of SAI Intelligent Systems Conference*, pp. 684–695. Springer, (2019).

[4] Arthur Bertachi, Lyvia Biagi, Iván Contreras, Ningsu Luo, and Josep Vehí, 'Prediction of blood glucose levels and nocturnal hypoglycemia using physiological models and artificial neural networks.', in *KHD@ IJCAI*, pp. 85–90, (2018).

[5] Julio Borges, *The Power of Ensembles in Deep Learning*, 2019. https://towardsdatascience.com/the-power-of-ensembles-in-deep-learning-a8900ff42be9.

[6] Danielle Bruen, Colm Delaney, Larisa Florea, and Dermot Diamond, 'Glucose sensing for diabetes monitoring: recent developments', *Sensors*, **17**(8), 1866, (2017).

[7] Razvan Bunescu, Nigel Struble, Cindy Marling, Jay Shubrook, and Frank Schwartz, 'Blood glucose level prediction using physiological models and support vector regression', in *2013 12th International Conference on Machine Learning and Applications*, volume 1, pp. 135–140. IEEE, (2013).

[8] Mol Ecol, 'HHS Public Access', **25**(5), 1032–1057, (2017).

[9] Eleni I Georga, Vasilios C Protopappas, Diego Ardigò, Demosthenes Polyzos, and Dimitrios I Fotiadis, 'A glucose model based on support vector regression for the prediction of hypoglycemic events under free-living conditions', *Diabetes technology & therapeutics*, **15**(8), 634–643, (2013).

[10] Sepp Hochreiter and Jürgen Schmidhuber, 'Long short-term memory', *Neural computation*, **9**(8), 1735–1780, (1997).

[11] George S Jeha, Lefkothea P Karaviti, Barbara Anderson, EO'Brian Smith, Susan Donaldson, Toniean S McGirk, and Morey W Haymond, 'Continuous glucose monitoring and the reality of metabolic control in preschool children with type 1 diabetes', *Diabetes Care*, **27**(12), 2881–2886, (2004).

[12] Heydar Khadem, Mohammad R Eissa, Hoda Nemat, Osamah Alrezj, and Mohammed Benaissa, 'Classification before regression for improving the accuracy of glucose quantification using absorption spectroscopy', *Talanta*, **211**, 120740, (2020).

[13] Cindy Marling and Razvan Bunescu, 'The ohiot1dm dataset for blood glucose level prediction: Update 2020'.

[14] Cindy Marling and Razvan C Bunescu, 'The OhioT1DM Dataset For Blood Glucose Level Prediction.', in *3rd International Workshop on Knowledge Discovery in Healthcare Data*, pp. 60–63, (2018).

[15] John Martinsson, Alexander Schliep, Björn Eliasson, Christian Meijner, Simon Persson, and Olof Mogren, 'Automatic blood glucose prediction with confidence using recurrent neural networks', in *3rd International Workshop on Knowledge Discovery in Healthcare Data, KDH@ IJCAI-ECAI 2018, 13 July 2018*, pp. 64–68, (2018).

[16] Cooper Midroni, Peter J. Leimbigler, Gaurav Baruah, Maheedhar Kolla, Alfred J. Whitehead, and Yan Fossat, 'Predicting glycemia in type 1 diabetes patients: Experiments with XGBoost', *CEUR Workshop Proceedings*, **2148**, 79–84, (2018).

[17] Sadegh Mirshekarian, Razvan Bunescu, Cindy Marling, and Frank Schwartz, 'Using lstms to learn physiological models of blood glucose behavior', in *2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 2887–2891. IEEE, (2017).

[18] Fionn Murtagh, 'Multilayer perceptrons for classification and regression', *Neurocomputing*, **2**(5-6), 183–197, (1991).

[19] Shauna S Roberts, 'Type 1 diabetes', *Diabetes Forecast*, **55**, 19, (2002).

[20] Svante Wold, Michael Sjöström, and Lennart Eriksson, 'Pls-regression: a basic tool of chemometrics', *Chemometrics and intelligent laboratory systems*, **58**(2), 109–130, (2001).

[21] Ashenafi Zebene Woldaregay, Eirik Årsand, Taxiarchis Botsis, David Albers, Lena Mamykina, and Gunnar Hartvigsen, 'Data-driven blood glucose pattern classification and anomalies detection: machine-learning applications in type 1 diabetes', *Journal of medical Internet research*, **21**(5), e11030, (2019).

[22] Jinyu Xie and Qian Wang, 'Benchmark machine learning approaches with classical time series approaches on the blood glucose level prediction challenge.', in *KHD@ IJCAI*, pp. 97–102, (2018).

[23] Zhi-Hua Zhou, *Ensemble methods: foundations and algorithms*, CRC press, 2012.

Online Blood Glucose Prediction Using Autoregressive Moving Average Model with Residual Compensation Network

Ning Ma and **Yuhang Zhao** and **Shuang Wen** and **Tao Yang** and **Ruikun Wu** and **Rui Tao** and **Xia Yu** and **Hongru Li**[1]

Abstract. Blood glucose (BG) prediction plays an important role in daily BG control. Accurate prediction of short-term glucose concentration can provide early warning for hyperglycemia and hypoglycemia events. This paper proposed a novel framework that combined an online prediction model with a residual compensation network. The autoregressive moving average (ARMA) model was used for online blood glucose prediction and the neural network was applied for compensation of prediction error. The advantages of this combined framework are: (1) the online ARMA model is efficient and robust to capture time-varying glucose dynamics, (2) the residual compensation network is capable to estimate errors from the online prediction model. The performance of this method was evaluated by the root mean squared error (RMSE) and the mean absolute error (MAE) in the dataset of OhioT1DM.The results were shown in detail that the mean values of the best RMSE of six patients at 30-min and 60-min horizon were 20.03 and 34.89 respectively, and the best MAE at 30-min and 60-min horizon were 14.52 and 24.61. Compared with the ARMA model, the combined predictor with a residual compensation network shows better prediction accuracy. Thus, we concluded that the proposed framework was an available approach for online blood glucose level prediction (BGLP).

1 INTRODUCTION

Nowadays, daily BG management is a significant challenge for a patient with diabetes. Further improvement of glucose control can be realized through prediction, which allows users to take actions ahead of time to minimize the occurrence of adverse glycemic events [3]. Thus, accurate blood glucose prediction plays an important role in blood glucose control. However, multiple factors influence glucose variability and lead to different responses between individuals under the same conditions. The prediction of short-term glucose concentration has become an urgent problem for researchers. In the past, various machine learning approaches were proposed to develop data-driven glucose predictive models [22]. John et al. [13] used Recurrent Neural Networks that trained in an end-to-end fashion to predict future blood glucose levels through historical blood glucose data. Jaouher et al. [2] applied an Artificial Neural Networks model to predict future blood glucose levels and hypoglycemic events of Type 1 Diabetes Mellitus (T1DM). The results proved that the model was accurate, adaptive, and encouraging by clinical implementation. Reymann et al. [19] trained a Support Vector Regression model with an online software simulator. They provided the foundation for the further development of the mobile prediction.

Nevertheless, every prediction algorithm has its own advantages and disadvantages. The ARMA model can be constructed easily by several steps, but they lack the ability to deal with the nonlinear patterns [15]. Due to the extremely non-stationary characteristic of the time series, the single artificial intelligence models sometimes stuck into the local minimum and fail to achieve satisfactory performance. With the development of equipment, the generation of data flow is continuous. Tracking the time-varying characteristic of the system is crucial. Regarding the non-stationary time series, most scholars adopted one online learning method to model the complex system. The input of the data can adjust the parameters of the model in real-time [12]. The data of blood glucose is non-stationary, aperiodic, and individuality. Therefore, the use of only one method for BG prediction may give one-sided results [14]. We need to combine various prediction methods to cover the disadvantages.

In this paper, we proposed a novel framework that combined an online prediction model with a residual compensation network. The ARMA model was used for online blood glucose prediction and the neural network was applied for compensation of prediction error. The advantages of this combined framework are: (1) the online ARMA model is efficient and robust to capture time-varying glucose dynamics, (2) the residual compensation network is capable to estimate errors from the online prediction model. The accuracy of this method was evaluated by short-term glucose prediction in the data set of OhioT1DM.

This paper is structured as following five parts: section I presents a brief literature review that discusses related works on short-term glucose prediction technique; section II presents our method for data preprocessing; section III introduces the principle of the online ARMA model and neural network, as well as the overall framework; section IV discusses the performance of our method on clinical data, and section V concludes the paper.

2 DATA PREPROCESSING

The data used in this paper is provided by the BGLP challenge. OhioT1DM dataset recorded 8-week CGMs data and corresponding daily events from 6 patients with type 1 diabetes, including numbers 540, 544, 552, 567, 584, and 596, respectively. During data collection and transmission, the errors in calibration or measurements may be produced many missing or outlier data points in clinical data. Although, time series models do not consider any physiological factors

[1] Northeastern University, China, email: lihongru@ise.neu.edu.cn

and only use recent BG data and other inputs that may affect BG levels. The missing data will have a significant effect on the accuracy of the models [21].

Online models emphasize the real-time input of data streams, hence, the missing data can only be estimated using past data [7]. Our workflow for dealing with missing data problem is as follows. Based on the CGM data, a time grid with a 5-minute sample period was derived and the missing data were filled with zeros. Firstly, we made a statistical analysis of the size and number of missing data segments through excel software. In both the test set and the training set, there are more than 5% and even 20% missing data. Among them, the loss of blood glucose between 1-100 is relatively common, which may be caused by the replacement of CGM in patients. Secondly, with the statistical results, a backward pushing method or mean value method was implemented for each missing. With the increase of filling times, the cumulative error will inevitably increase.For the training set, the missing CGM values were filled with spline and the historical average at the same point. When the two values are different, the weighted method is used to fill. The test set is processed as follows :(a) the first three positions of the missing segment are filled with extrapolation method;(b) starting from the fourth position of the missing segment, weight the first-order Taylor series extrapolation and average (the historical average at the same point and historical average) to fill;(c) from position 12 of the missing paragraph uses backward induction. Finally, unbroken data would be obtained for prediction. Although many models with multiple inputs (insulin dose, food intake, etc.) can effectively predict the future BG levels. However, the data-collection process of those inputs heavily relies on the subjective inputs provided by the user who wears a CGM device. Since the user may not be professional, the data may be inaccurate and have errors. Due to such limitations, we predicted the future BG level only based on the historical BG data.

3 METHODS AND REALIZATION

In this section, we will introduce the models that are used in the framwork and explain how the proposed framework works for prediciton.

3.1 ARMA model

3.1.1 ARMA model

ARMA, which includes the autoregressive (AR) model and moving-average (MA) model, is an important method to study the time series [17]. It is widely used in the prediction of finance and wind power [1], [20]. The ARMA could establish linear and nonlinear dynamic models by associating input and output data. And it can be expressed as follows:

$$y_t = \sum_{i-1}^{p} \beta_i y_{t-i} + \sum_{j-1}^{q} \alpha_j \epsilon_{t-j} + \epsilon_t \tag{1}$$

Where p is the order of the autoregressive part, β_i is the autoregressive parameter, q is the order of the moving average part, α_j is the moving average parameter, and ϵ_t is the error term at time t. In general, the offline parameter determination uses the Least-squares and the online uses Kalman filter.

3.1.2 Online model

The ARMA includes three iterative steps including model identification, parameter estimation, and diagnostic checking. Stationarity is a necessary condition in building an ARMA model which is useful for forecasting. In the identification step, data transformation is often required to make the time series stationary. Meanwhile, the sliding window technique is more robust to the stochastic changes in the data trend and can be applied to smaller datasets [25]. Hence, the sliding window technique was added to the ARMA model. Discarding old data from the training window can limit the influence of distant past trends during model training and can promote the learning of new trends in the data.

Differencing was applied to it to remove the trend and stabilize the variance because of the trend of the blood glucose data. After that, the sliding window updated the BG data. The method can reduce the training time of the model because the number of training sets is always fixed. As far as we know, determining the order of models is a key to the ARMA model. Akaike's Information Criterion (AIC) is widely used to optimize the model parameters in those models. AIC is an estimation for the likelihood of a model. However, AIC does not have any indication of the absolute quality. The Bayesian Information Criterion (BIC) is a similar criterion for model selection [9]. Then, AIC and BIC were used to select an appropriate order in this paper. For the two results, we limit the interval value of the global model order, to conduct experiments to find the optimal model parameters. The last step of model building is the diagnostic checking of model adequacy. If the model is not adequate, a new tentative model should be identified, which is again followed by the steps of parameter estimation and model verification. The ability of the ARMA model in learning small data sets and tracking fast is taken full advantage and can achieve the online update learning.

3.2 Residual compensation network

3.2.1 Neural network

Backpropagation (BP) neural network is a model that can approximate various nonlinearities in the data. It is a kind of multi-layer feedforward neural network trained according to the error propagation algorithm and there are three layers including the input layer, hidden layer, and output layer. In essence, the BP neural network takes the network error square as the objective function and uses the gradient descent method to calculate the minimum value of the objective functio [6]. Modifying the weight and threshold is the core of the BP neural network. It aims to get the model whose output is consistent with expected results. In this paper, the input layer of the neural network is the predictive value of blood glucose, and the output is the prediction error. The structure is shown in Figure 1.

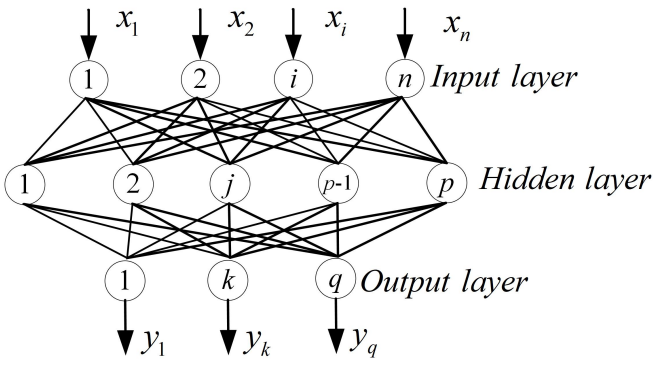


Figure 1. The basic structure of BP neural network.

In Figure 1, n is the number of nodes in the hidden layer, p and q

are the number of nodes in the input layer and output layer respectively. The number of hidden layers can be determined according to the empirical formula:

$$n = \sqrt{p+q} + a \tag{2}$$

Where a is the adjustment constant between 1 and 10. The number of input layers is determined by correlation analysis. Then, the best number of hidden layers is determined by the experiment to follow equation (2). It is generally believed that increasing the number of hidden layers can reduce the network error and improve the accuracy, but also complicate the network, thus increasing the network training time and the tendency of overfitting.

3.2.2 Framework of residual compensation

Both the ARMA model and BP neural network have achieved successes in their own linear or nonlinear domains. Neither of them is suitable for all circumstances. The statistical methods have their linear limitations, which means that they cannot simulate the real-time series with nonlinear mode well [5]. On the other hand, a single BP neural network is not enough to capture the time patterns contained in highly complex time series. In the training process of the BP neural network, there may be problems of the model following error and uncertainty, resulting in the generation of overfitting or underfitting model [11]. A hybrid methodology can be a good strategy for practical use [24] , [4]. It combines different models to capture different aspects of the underlying patterns. Ji et al. [10] used the ARMA model to predict linear components of the time series, and the TDNN model to predict nonlinear components. Results showed that the model had the advantages of both two methods and the prediction accuracy of the model was improved. However, only the optimal combinations of different models can obtain the best hybrid models, the framework of the hybrid models becomes very important.

In this paper, we proposed a novel framework that combined an online ARMA model with a residual compensation network (RCN-ARMA) to predict BG. The blood glucose data which belongs to chaotic time series contains linear and nonlinear components [8]. Due to the randomness and volatility of BG, the ARMA model inevitably produces large errors in the prediction of nonlinear non-stationary time-series data, which has a certain tendency and periodicity [23]. The BP neural network has good data error tolerance, but it is insufficient for linear prediction. Since the ARMA model cannot capture the nonlinear structure of the BG data. The residuals of the linear model will contain information about the nonlinearity. The BP neural network is valid for satisfying the prediction effect of most non-linear properties. Hence, the BP neural network was applied to predict residuals. The framework aimed to reduce the uncertainty of model selection and improve the model forecasting performance by dealing with both linear and nonlinear patterns in time series. The flow diagram of the RCN-ARMA is shown in Figure 2.

The specific prediction process of RCN-ARMA is as follows:

Step 1. The sliding window updates the input for the ARMA model. AIC and BIC are used to confirm the order of ARMA. Then predict the blood glucose by the online ARMA model.

Step 2. Compared to the predicted value with the raw data, the residual time series, which is used to the compensation network, can be constructed.

Step 3. The correlation analysis of the predicted values and residual time series is carried out to determine the input of the residual compensation network [16]. According to the results of the correlation analysis, the range of input variables may be different from 6 patients.

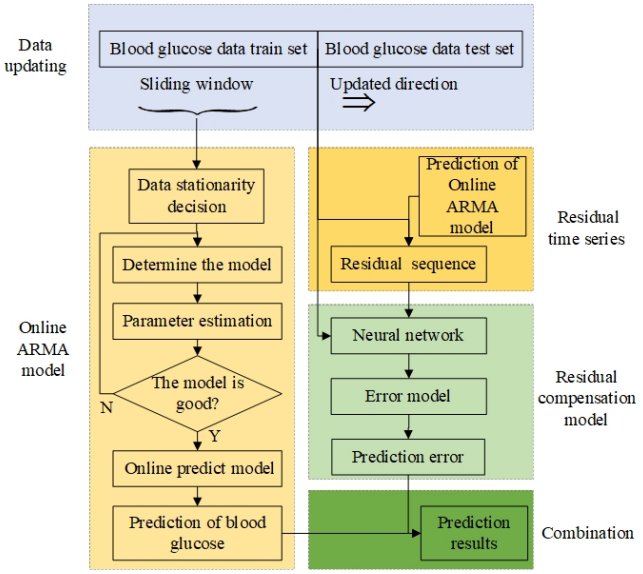


Figure 2. Flow diagram of online ARMA model with residual compensation network.

Step 4. As an important supplement to model prediction, a common three-layer neural network is applied to predict the residual. The neural network predicts the errors in the future based on a series of errors in the past and can overcome the influence of various uncertainties changes on system stability.

Step 5. Analysis of blood glucose predictions and residual time series in statistically. The output display value range of the CGM is [40,400], and the error is basically within the range of [-50,50]. For this reason, some rules are employed to correct discrete data points appropriately in the research.

Step 6. Combine the results of the two-step prediction and get the final After the five steps, we have got the prediction results of the BG and the error. Combine the results of the two-step prediction by the direct sum method and get the final BG prediction.

4 RESULTS AND DISCUSSION

4.1 Evaluating indicator

For model evaluation, general and commonly used evaluation methods are sensitivity, specific, root mean square error (RMSE), and mean absolute error (MAE) [18]. In this paper, two widely used evaluation indexes were applied to compare the prediction capacity. The error indexes define as below:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2} \tag{3}$$

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|\hat{y}_i - y_i| \tag{4}$$

Where: $\hat{y}_i$ represents the predicted value, y_i represents the real value and N represents the size of the data set. Two rules were applied in the evaluation: 1) as long as the corresponding timestamp had the raw data, the RMSE and MAE indexes of the test set would be calculated; 2) If there was a null value in the input model data, it meant that

the data was given insufficiently, and the value at this time would not be recorded. We only use the first and the code used during the experiment is available on Github. In this paper, the predictions of the model were recorded from the thirteenth point of the test set. And the results were recorded as two decimal places rounded.

4.2 Results

In this section, the results and analysis of the proposed framework are presented. The online AR, BP, and ARMA models were used for 30-min ahead predictions. The mean values of the RMSE and MAE for six patients are shown in Table 1. Then the RCN-ARMA was used to 30-min and 60-min ahead predictions. The experiments were conducted on patients with different inputs by establishing an online ARMA model and a residual compensation network (RCN-ARMA). The optimal value of the sliding window was selected by the experimental method and keeps the same in two networks. Due to the heterogeneity of the patients themselves, the selected parameters had some differences. The 30-min ahead predictions of ARMA and RCN-ARMA for 540, 567 patients are graphically shown in Figure 3 and Figure 4. Table 2 shows the RMSE and MAE of the different contributors for 30-min and 60-min ahead predictions. Based on the results in table2, mean RMSE and MAE of 30-min and 60-min ahead predictions respectively with the online ARMA and RCN-ARMA are shown in Table 3. The above tables contain the results of three cases, and the reliability of the conclusions is enhanced through a comparison of multiple cases.

Table 1. Mean values of the RMSE and MAE for different models (prediction horizon (PH) =30 minutes.)

Method	AR	BP	ARMA
RMSE	21.80	33.45	21.44
MAE	15.93	24.54	15.17

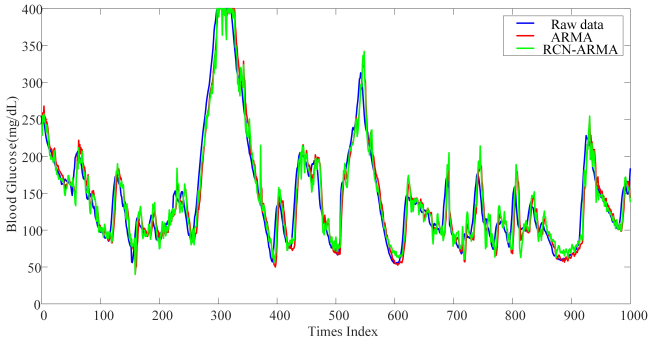


Figure 3. Forecasting results of patient 540 for 30-min ahead predictions.

Table 1 shows that different models have different prediction effects on blood glucose prediction. The ARMA model is better than the other two models in prediction. The reason is that the online ARMA model has an advantage in tracking real-time changes of data.And the AR model which does not contain the moving average model (MA) is a special form of ARMA model. There is a big difference in MAE between the two. Therefore, we choose online ARMA as the base model. Figure 3 and Figure 4 clearly illustrated that (a) the predicted value of ARMA has obvious lag on the whole, which is

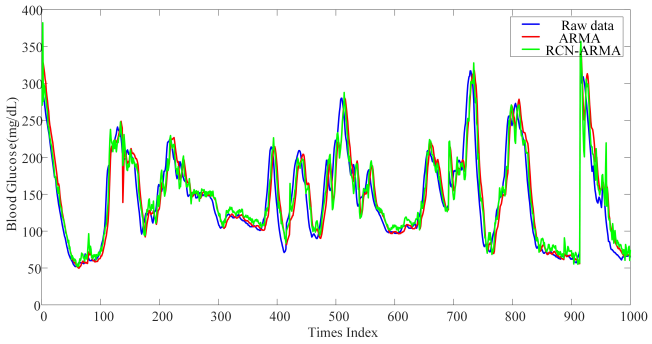


Figure 4. Forecasting results of patient 567 for 30-min ahead predictions.

one of the main reasons affecting the prediction effect of the model; (b) the addition of the error compensation model improves the hysteresis of the predicted value of the model; (c) the mixed prediction results show sharp fluctuations and a certain amount of peak data that are negative effects of adding compensation.

Table 2. RMSE and MAE of the RCN-ARMA model for 6 patients (PH=30 and 60 minutes).

ID	540	544	552	567	584	596
30-RMSE	22.19	17.66	17.40	21.12	23.88	17.93
30-MAE	16.29	13.27	12.95	14.94	16.99	12.68
60-RMSE	40.03	31.873	30.06	38.42	38.71	30.27
60-MAE	30.32	24.25	22.88	29.58	29.03	22.39

Table 3. Mean values of the RMSE and MAE for ARMA and RCN-ARMA model.

	PH=30 minutes		PH=60 minutes	
Method	RMSE	MAE	RMSE	MAE
ARMA	21.44	15.17	38.78	28.42
RCN-ARMA	20.03	14.52	34.89	26.41
Drop value	1.41	0.65	3.89	2.01

As can be seen from Table 2 and Table 3: (a) for different patients, the model prediction effect is different and reflects the specificity of blood glucose data; (b) prediction ability of the model got worse with the increase of the prediction step. This is a major issue that needs to be addressed urgently; (c) through the correlation analysis of predicted value residuals, it implies that a significant correlation relationship exists for the multi-step ahead forecast error series of ARMA. Thus, it is very useful for the error forecast models to select effective input variables in this multi-step ahead forecasting model; (d) compared with the online ARMA model, the evaluating indicator of RCN-ARMA all decreased, especially for 60-min ahead predictions; (e) from the drop value, the change of two different step size evaluation indexes gradually increases. The overall effect decreases with the increase of prediction step size for both models. The improvements of the proposed combined framework compared with a certain individual model increase with increasing prediction steps for the continuous multi-step ahead forecasting.

4.3 Discussion

To further compare the performance difference between models, the effectiveness of the proposed model is demonstrated by the promoting percentage of between models. The data collected from 6 patients is used as our case study. The simulation results demonstrate that the proposed forecasting framework improves the short-term blood glucose forecasting accuracy significantly compared with the reference models. The residual compensation network can timely predict the errors to supplement the missing nonlinearity information of the ARMA model. The proposed framework not only retains the advantage of the ARMA model for fast-tracking a small amount of data but also covers the shortage of nonlinear learning which mainly affects the overall improvement of the results. For the neural networks, the advantage of the framework can be reflected by its ability of nonlinear prediction. It proves that the framework can better capture the nonlinear and linear characteristics of the time series. Compared with using a single algorithm, this framework is more comprehensive. At present, both time series and machine learning algorithms have their disadvantages. People have been studying the corresponding matching algorithm to solve the disadvantage of the algorithm. This framework of models can be promoted to an individual model by fixing known flaws using a complementary model.

5 CONCLUSION

In this study, a new framework for blood glucose prediction based on the online ARMA model with residual compensation network was proposed. The online ARMA model was applied for predicting dynamic changes of blood glucose in real-time, and the residual model was used to track the errors of the online model. Prediction results of 6 patients, the RCN-ARMA had much higher prediction accuracy than the ARMA model. The proposed framework improved the ability of ARMA model prediction and proposed a better short-term prediction performance. Because the accuracy of the ARMA model in blood glucose prediction is improved, the application of the ARMA model in the artificial pancreas (AP) system will have better safety and stability. From the time series prediction results, the framework is also applicable to the integration of other prediction models to achieve clinical applications. The aim was to cover the missing useful prediction information caused by the shortcomings of the single model. Future work, we will further select an appropriate evolutionary algorithm to optimize the model parameters.

6 FUNDING

This research was supported by National Natural Science Foundation of China (No.61973067 and No.61903071).

REFERENCES

[1] L. Arisena, A.and Noviyanti and S. Achmad Zanbar, 'Portfolio return using black-litterman single view model with arma-garch and treynor black model', in *Journal of Physics: Conference Series*, volume 974, Surabaya, Indonesia, (2018).

[2] J. Ben Ali, T. Hamdi, N. Fnaiech, V. Di Costanzo, F. Fnaiech, and J.-M. Ginoux, 'Continuous blood glucose level prediction of type 1 diabetes based on artificial neural network', *Biocybernetics and Biomedical Engineering*, **38**, 828 – 40, (2018).

[3] R.H. Botwey, E. Daskalaki, P. Diem, and S.G. Mougiakakou, 'Multi-model data fusion to improve an early warning system for hypo-/hyperglycemic events', in *36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 4843 – 6, Piscataway, NJ, USA, (2014).

[4] P.S.G. de Mattos Neto, G.D.C. Cavalcanti, and F. Madeiro, 'Nonlinear combination method of forecasters applied to pm time series', *Pattern Recognition Letters*, **95**, 65 – 72, (2017).

[5] J. F. L. de Oliveira, L. D. S. Pacifico, P. S. G. de Mattos Neto, E. F. S. Barreiros, C. M.O. Rodrigues, and A. T. A. Filho, 'A hybrid optimized error correction system for time series forecasting', *Applied Soft Computing Journal*, **87**, (2020).

[6] Sh. Ding and Q.H. Wu, 'A matlab-based study on approximation performances of improved algorithms of typical bp neural networks', in *Applied Mechanics and Materials*, pp. 1353 – 1356, (2013).

[7] E. I. Georga, V. C. Protopappas, D. Polyzos, and D. I. Fotiadis, 'Online prediction of glucose concentration in type 1 diabetes using extreme learning machines', in *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS*, pp. 3262 – 3265, (2015).

[8] T. Hamdi, V. Di Costanzo, F. Fnaiech, E. Moreau, R. Naeck, and J.M. Ginoux, 'Glycemic evolution of type 1 diabetic patients is a chaotic phenomenon', in *IECON Proceedings (Industrial Electronics Conference)*, pp. 5177 – 5181, Florence, Italy, (2016).

[9] R.J. Hyndman and G. Athanasopoulos, *Forecasting: Principles and Practice, 3rd edition*, OTexts: Melbourne, Australia, 2019.

[10] W. Ji and K. C. Chee, 'Prediction of hourly solar radiation using a novel hybrid model of arma and tdnn', *Solar Energy*, **85**, 808 – 817, (2011).

[11] X.Y. Kong, Li Ch., Ch.Sh. Wang, Y.S. Zhang, and J. Zhang, 'Short-term electrical load forecasting based on error correction using dynamic mode decomposition', *Applied Energy*, **261**, (2020).

[12] M. Langkvist, L. Karlsson, and A. Loutfi, 'A review of unsupervised feature learning and deep learning for time-series modeling', *Pattern Recognition Letters*, **42**, 11 – 24, (2014).

[13] K. Li, J. Daniels, L Chengyuan, P. Herrero, and P. Georgiou, 'Convolutional recurrent neural networks for glucose prediction', *IEEE Journal of Biomedical and Health Informatics*, **24**, 603 – 613, (2020).

[14] K.Z. Li, Ch.Y. Liu, T.Y. Zhu, P. Herrero, and P. Georgiou, 'Glunet: A deep learning framework for accurate glucose forecasting', *IEEE Journal of Biomedical and Health Informatics*, **24**, 414 – 423, (2020).

[15] Y.F. Li, H.P. Wu, and H. Liu, 'Multi-step wind speed forecasting using ewt decomposition, lstm principal computing, relm subordinate computing and iewt reconstruction', *Energy Conversion and Management*, **167**, 203 – 219, (2018).

[16] Zh.T. Liang, J. Liang, Ch.F. Wang, X.M. Dong, and X.F. Miao, 'Short-term wind power combined forecasting based on error forecast correction', *Energy Conversion and Management*, **119**, 215 – 226, (2016).

[17] N. S. Nalawade and M.M. Pawar, 'Forecasting telecommunications data with autoregressive integrated moving average models', in *2015 2nd International Conference on Recent Advances in Engineering and Computational Sciences, RAECS 2015*, Chandigarh, India, (2015).

[18] S. E. Noujaim, D. Horwitz, M. Sharma, and J. Marhoul, 'Accuracy requirements for a hypoglycemia detector: An analytical model to evaluate the effects of bias, precision, and rate of glucose change', *Journal of Diabetes Science & Technology*, **1**(5), 652–668, (2007).

[19] M.P. Reymann, E. Dorschky, B.H. Groh, C. Martindale, P. Blank, and B.M. Eskofier, 'Blood glucose level prediction based on support vector regression using mobile platforms', in *38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 2990 – 3, Piscataway, NJ, USA, (2016).

[20] H. Sharadga, S. Hajimirza, and R. S. Balog, 'Time series forecasting of solar power generation for large-scale photovoltaic plants', *Renewable Energy*, **150**, 797 – 807, (2020).

[21] C.F. Tsai and Y.C. Chen, 'The optimal combination of feature selection and data discretization: An empirical study', *Information Sciences*, **505**, 282 – 293, (2019).

[22] A. Z. Woldaregay, E. Arsand, S. Walderhaug, D. Albers, L. Mamykina, T. Botsis, and G. Hartvigsen, 'Data-driven modeling and prediction of blood glucose dynamics: Machine learning applications in type 1 diabetes', *Artificial Intelligence in Medicine*, **98**, 109 – 134, (2019).

[23] J. Yang, L. Li, Y.M. Shi, and X.L. Xie, 'An arima model with adaptive orders for predicting blood glucose concentrations and hypoglycemia', *IEEE Journal of Biomedical and Health Informatics*, **23**, 1251 – 1260, (2019).

[24] G.P. Zhang, 'Time series forecasting using a hybrid arima and neural network model', *Neurocomputing*, **50**, 159 – 75, (2003).

[25] L.Y. Zhang, J.B. Zhao, and W. Li, 'Online and unsupervised anomaly detection for streaming data using an array of sliding windows and pdds', *IEEE Transactions on Cybernetics*, 1–6, (2019).