TRUMIT: A Tool to Support Large-Scale Mining of Text Association Rules

Robert Neumayer[1], George Tsatsaronis[2], and Kjetil Nørvåg[1]

[1] Norwegian University of Science and Technology,
Department of Computer and Information Science, Trondheim, Norway,
{neumayer,noervaag}@idi.ntnu.no
[2] Biotechnology Center
Technical University of Dresden, Germany
george.tsatsaronis@biotec.tu-dresden.de

Abstract. Due to the nature of textual data the application of association rule mining in text corpora has attracted the focus of the research scientific community for years. In this paper we demonstrate a system that can efficiently mine association rules from text. The system annotates terms using several annotators, and extracts text association rules between terms or categories of terms. An additional contribution of this work is the inclusion of novel unsupervised evaluation measures for weighting and ranking the importance of the text rules. We demonstrate the functionalities of our system with two text collections, a set of *Wikileaks* documents, and one from TREC-7.

1 Introduction

Association Rule Mining (*ARM*) is a well-researched field of data mining. Rules can help to uncover hidden or previously unknown associations. A rule in the form of $A => B$, denotes an implication of element or item B by item A. Association rules have successfully been used in a wide range of domains, e.g., market basket analysis, law enforcement, biotechnology.

Lately, the benefits of applying ARM to text have appeared in query refinement and applications in text search and has become an objective of vital interest to the area of text mining and the practitioners of the field. *Text Association Rule Mining* (*TARM*) faces new challenges with respect to the volume of the data and the number of distinct items. Another major challenge is the interpretation of the rules as well as the evaluation and ranking of these rules according to their importance.

In this paper we address those issues, and we present the *text rule mining testbench* (TRUMIT). Our system implements all the stages of *TARM*, namely pre-processing, the actual mining of rules, and thorough analysis and visualization of the generated rules. We give a special focus on the pre-processing, and more precisely on the annotation step. We integrate a range of different annotation types including simple tokenization and matching, part-of-speech (POS) tagging, named entity recognition (NER), or more advanced types of semantic annotation based on the WordNet[3] and the OpenCalais[4] cat-

[3] http://wordnet.princeton.edu/
[4] http://www.opencalais.com/

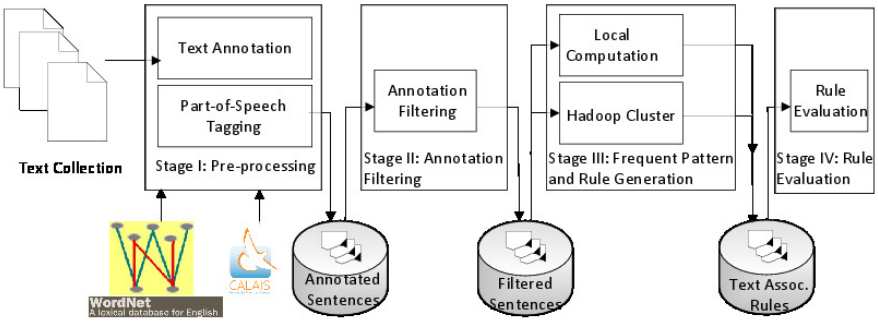

Fig. 1. Overview of the system including its four processing stages

egories. The input text may be filtered according to the different annotation types. After pre-processing, the system allows for *Frequent Pattern Mining* (*FPM*) via the *Hadoop Map Reduce framework* before we extract the actual rules. Finally, we integrate existent, but also new, evaluation metrics for the extracted text rules. The system provides an easy-to-use interface for inspecting the generated rules, which may also function as a search interface for the respective collection.

2 Text Association Rule Mining

In its original form, association rule mining discovers regularities in data [1]. We consider a set of transactions $D = \{d_1, d_2, ..., d_n\}$, each transaction $d_i \in D$ comprises a set of items, i.e., $d_i = \{i_1, i_2, ..., i_m\}$. We also denote with I the set of all distinct items i_j that may occur in any transaction $d_i \in D$. Any subset $I_m \subseteq I$ is called an *itemset*.

In our context we denote a text document as T_i which belongs to a document collection T; we define as D the set of all text sentences. The set of all possible items of a transaction is the set of distinct terms T. Additionally, we create a second level of items, comprising all of the annotations A of all terms $t_j \in T_i$. Examples of such annotations may be *Person*, *Date*, or *Company*, which generates category rules of the form *Person* $=>$ *Company*, or *Company* $=>$ *Date*. Extraction of such rules unfolds the meaning of *Bill Gates* $=>$ *Microsoft*, or *Google* $=>$ *1998*. We include annotations provided by *OpenCalais* and we consult the *WordNet* thesaurus to annotate nouns with their respective domain terms. However, the system's architecture allows for easy integration of new annotation plug-ins. An example category rule, by including *WordNet* domain terms, could be *Animal* $=>$ *Company*, which might denote the information that *Company* conducts animal experiments in the life sciences domain.

With regards to related work, an approach for activity and emotion rule mining is presented in [3]. The authors mine simplified rules from a large collection of blog entries based on multiple minimum support. A tool to mine *maximal association rules* is introduced in [2]. A limited set of named entities is used for association rule mining. Experiments are performed on collections up to 10.000 documents. Another toolkit for *TARM* is presented in [4]. The authors worked on ARM in temporal document collec-

tions, and extended previous work by performing mining based on semantics, as well as by studying appropriate evaluation techniques. The focus was on the temporal aspect of the extracted rules. Scalability issues were not taken into account to a satisfactory extent.

To the best of our knowledge there does not exist a tool that includes all of the main *TARM* stages, shown in Fig. 1, which are included in our system. Existing tools are either not focused on text, or lack a capable user interface, or they cannot offer generic annotation integration, or are not competitive in terms of scalability. We aim to satisfy all of these requirements.

3 TRUMIT: Text Rule Mining Testbench

TRUMIT comprises four distinct processing stages, shown in Fig. 1. All intermediate results are stored on disk making it easier to work with large scale collections. In the following we will describe the processing stages of TRUMIT.

Text annotation: This stage integrates a wide range of annotation plugins based on *Apache UIMA*. Currently, the following annotator types are supported: *language annotator*, *open calais annotator*, *POS annotator*, *Stanford NER annotator*, *Wordnet domain annotator*[5], *maui keyword annotator*[6], *lexical emotion annotator*, and *Wikipedia miner annotator*[7]. For reasons of simplicity in the figure we only demonstrate two of the used annotators (*WordNet* and *OpenCalais*). The system architecture at this stage is general enough to host any other annotator for the pre-processing step, through *Apache UIMA*.

Annotation filtering: At this second stage, we allow for flexible filtering of annotations by the user. Filtering is possible according to certain POS tags, entities, keywords or any other annotation technique used in the pre-processing step.

Frequent pattern mining and rule generation: The output of the filtering stage can subsequently be used for frequent itemset generation. To this end, a *Hadoop* job for the *fp-growth* implementation of the *Apache Mahout*[8] library is started and, either computed locally or sent to a map reduce cluster. In this way we can guarantee the computation without being restricted to the hardware configuration of the client machine. From the result of the *map reduce* job we generate rules and assign scores based on *rule interestingness* or *evaluation criteria* that our system supports. Currently, the system includes *confidence*, *support*, *semantic relatedness*, and *similarity variance*.

Rule evaluation: Once the text association rules are computed and scored, in this final stage we provide an interactive way to the user of browsing and analysing them. Through component, the rules can be sorted according to the evaluation criteria described previously. We also provide means to easily search for documents matching certain rules. We show an example of how category rules can be mapped to rules based on text only in Fig. 3.

[5] `http://nlp.stanford.edu/software/CRF-NER.shtml`
[6] `http://code.google.com/p/maui-indexer`
[7] `http://wikipedia-miner.sourceforge.net/`
[8] `http://mahout.apache.org`

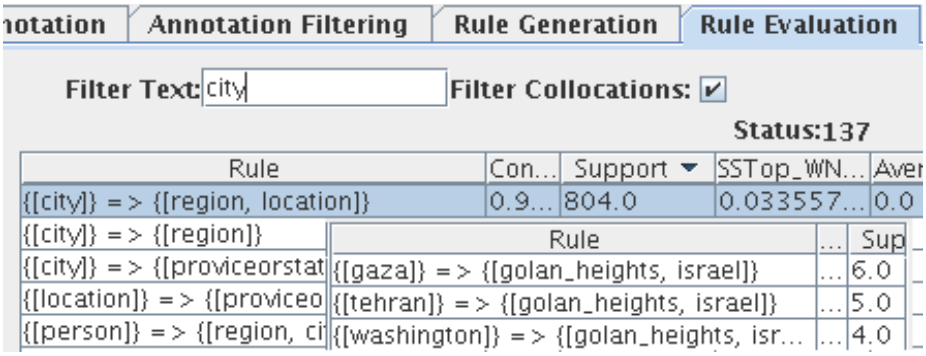

Fig. 2. TRUMIT user interface

4 Demonstration

In this demo we will show the full workflow by the example of the *cablegate* collection which is currently released by *wikileaks*. To this end we will illustrate how annotation can be performed with a range of plugins. Further we show the integration of both local rule generation and the map reduce framework for frequent pattern mining. We will show the benefits of additional evaluation measures for text association rules by example. Additionally, we will show analysis of rules which were computed offline for the 500.000 document TREC7 ad-hoc collection. The demonstration shows that our testbench offers a helpful tool integrating state-of-the-art libraries and technologies in its back-end.

References

1. Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. Mining association rules between sets of items in large databases. *SIGMOD Record*, 22:207–216, June 1993.
2. Amihood Amir, Yonatan Aumann, Ronen Feldman, and Moshe Fresko. Maximal association rules: A tool for mining associations in text. *Journal of Int. Inf. Systems*, 25(3):333–345, 2005.
3. Takeshi Kurashima, Ko Fujimura, and Hidenori Okuda. Discovering association rules on experiences from large-scale blog entries. In *Proceedings of ECIR'09*, pages 546–553, 2009.
4. Kjetil Nørvåg and Ole Kristian Fivelstad. Semantic-based temporal text-rule mining. In *Proc. of CICLing'09*, pages 442–455, Mexico City, Mexico, March 1-7 2009.