IntelliMeal - Enhancing Creativity by Reusing Domain Knowledge in the Adaptation Process

Kari Skjold, Marthe Øynes, Kerstin Bach, and Agnar Aamodt

Department of Computer Science
Norwegian University of Science and Technology (NTNU)
Trondheim, Norway
http://www.idi.ntnu.no

Abstract. This paper presents IntelliMeal, a case-based reasoning (CBR) system for recommending recipes. The main focus of the system is customizing recipes to a given user query re-using the domain knowledge of a CBR system within adaptation rules. In this work we implement a CBR system that works with a limited case base (21 recipes) and increase the amount of recipe recommendations by adapting these recipes using addition, creation, substitution, suitability and title name customization rules.

Keywords: Case-Based Reasoning, Rule Engine, Computational Creativity, Adaptation, Recipe Recommendation

1 Introduction

This paper presents IntelliMeal[1], a case-based recipe recommendation system addressing the open challenge of the 2017 Computer Cooking Contest (CCC). Since its initialization in 2008, the competition has been running almost every year with minor adjustments. Several research groups have contributed to the CCC over the years using information retrieval, information extraction and semantic technologies along with Case-Based Reasoning when developing recipe recommendation systems. The researchers have contributed with various approaches to the task. Four of the more influential systems are Taaable [2,4,6], CookingCAKE [5,9,10], JaDaCook [7,3], and CookIIS [8,11]. The Taaable researchers built their system around a collaborative, semantic Wiki, which also serves as the main knowledge base. The CookIIS researchers focused on the pre-processing of data to make the substitution of ingredients fluent and more realistic. CookingCAKE targets the preparation instructions by implementing cooking workflows, and lastly, the JaDaWeb researchers focused on the implementation of natural language understanding.

IntelliMeal is a knowledge engineering heavy system utilizing Case-Based Reasoning (CBR)[1]. The system aims to customize recipes for a given query consisting of desired and undesired ingredients. The main focus of the system

[1] www.intellimeal.no

is enhancing creativity in the adaptation process. The underlying goal of this paper is to investigate whether it is possible to build a CBR system that adapts recipes in a way that they satisfy a user's desires and expectations.

The case base consists of twenty-one sandwich recipe cases, hierarchical taxonomies and a set of adaptation rules. The taxonomies are separated into nine attributes or *ingredient categories*. Each taxonomy defines similarities between a restricted set of ingredients that belong to the given ingredient category. The taxonomies are extended versions of the taxonomies used in CookIIS [8,11], an earlier participant of the CCC. The attributes/ingredient categories are also used to construct each case and the ingredients included in the recipe.

The paper is structured as follows: Section 2 explains an adapted version of the CBR cycle [1], section 3 presents the evaluation of the systems in terms of similarity computation after adaptation and user evaluations of adapted recipes. In the final section we discuss our results and summarize our work.

2 Methodology and implementation

IntelliMeal implements all steps of the CBR cycle, but it includes a second retrieval phase after an ephemeral case base including temporarily adapted cases as been created. This allows IntelliMeal to assess the similarity of cases that have been modified based on desired and undesired ingredients specified in the query.

Figure 1 shows an overview of our implemented version of the cycle.

Step 1 The problem presented is the user query, which consists of desired and undesired ingredients. As figure 1 illustrates, the query is split in two: One undesired query containing the undesired ingredients and one desired query containing the desired ingredient.

Step 2 Our modified version of the CBR cycle involves *two* retrieval steps. The first process is the case base retrieval. It involves retrieving the cases from the case base with the highest similarity score to the user query. Hence, cases with the best *starting point* to end up in successful recipe recommendations. As the retrieval method employed in IntelliMeal is an important feature of the system, it is explained more detailed in section 2.1. The retrieved cases are further copied. The retrieved original cases are kept for later use while considering the copied versions in the reuse step.

Step 3 The reuse step is the most comprehensive step of the cycle. It is also the main focus of IntelliMeal. Hence, it is explained more detailed in section 2.3. The goal is to customize cases (i.e. recipes) so that they better fit the user query. However, with restrictions to avoid distasteful recipe results. As figure 1 illustrates, domain knowledge, rules, and the queries are used in the adaptation process. The result from the reuse step is adapted versions of the cases in the case base, further referred to as adapted instances.

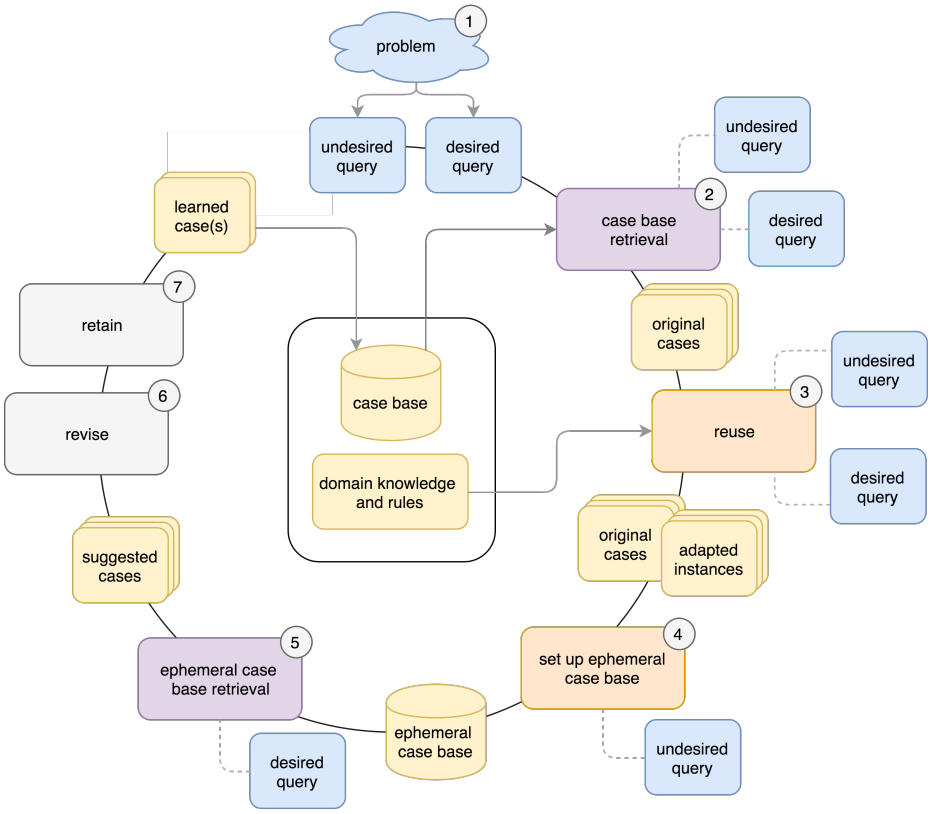

Fig. 1. Modified version of the CBR cycle

Step 4 The strategies used to set up an ephemeral case base mainly concerns using the undesired query to discard cases that are not satisfactorily. As figure 1 illustrates, both original cases from the case base as well as the adapted instances resulting from the reuse step takes part in the setup. The result is an ephemeral case base containing a selection of both types.

Step 5 The ephemeral case base retrieval involves comparing the cases in the ephemeral case base to the desired query. The result is a mixture of original cases and adapted instances, together with their resulting similarity score.

Step 6 The revision step involves getting feedback on recommendations. More specifically, the user gets the opportunity to confirm that adapted recipe recommendations are tasty.

Step 7 Only when an adapted recipe is confirmed tasty, the recipe instance will be added as a case to the case base together with the original cases. This refers to the retain step of the cycle.

2.1 Retrieval

Our retrieval method compares one query instance to all cases in a case base. The method considers one case at a time. It iterates through every attribute in the query and ignores attributes that are undefined. The rest of the attributes are considered valid and takes part in the similarity calculation.

For each valid attribute, the similarity between the query attribute and the corresponding case attribute is calculated by using the taxonomies. Then, they are weighted with a configured attribute weight. When calculating the total similarity between the query and a case, the attribute similarities are summed up and divided by the number of valid attributes. The retrieval method returns all cases ordered by their retrieved similarity score.

2.2 Rule engine

A rule engine was created for this specific system as a set of adaptation rules. There are two rule formats: *Simple Rule* and *Substitution Rule.*

If one rule were to specifically target only one ingredient, many rules would have to be written. Therefore, the rule engine is also able to consider all children of the ingredient. The children are fetched from the taxonomies. This enables one single rule to apply to hundreds of ingredients. For example, consider a rule saying any type of *meat* can substitute for any type of *fish*. These ingredients have 191 and 74 ingredients, respectively. Hence, this one rule will form 14 134 various combinations.

However, the functionality to ignore the children of a specific ingredients was also implemented. This can be used by writing a * after the ingredient. With this, a rule containing *meat** and *fish** would only form one combination.

Rule *requirements* refer to the ingredients in the recipe that have to be present for the rule to be valid. There can be zero or as many requirements as desired for a rule to fire. One requirement is satisfied if the recipe considered contains either the stated ingredient requirement or one of its children. A rule is valid when all requirements are satisfied.

Simple rules are the most basic rule type used in IntelliMeal. Based on given set of conditions (requirements req) an action is taken. For this rule type, all requirements have to be satisfied, before the rule can be applied.

$$req_1, ..., req_n \rightarrow ingredient \tag{1}$$

This rule type is used to create *addition rules*, *deletion rules*, *suitable rules* and *title rules*.

Substitution rules are more complex since they take both requirements and the ingredient to be substituted into account. Rules described by equation (2) takes the user requirement together with an ingredient to be substituted and the

action describes which new ingredient can be included given a conditions, e.g. the existence or non-existence of other ingredients.

$$req_1, ..., req_n + ingredient_{old} \rightarrow ingredient_{new} + condition \tag{2}$$

An example for this rule is if the recipe contains tuna, any type of undesired supplements can be substituted out in favor for mayonnaise.

Furthermore, equation (3) shows a bidirectional rule that allows forward and backward substitutions:

$$req_1, ..., req_n + ingredient_{old} \longleftrightarrow ingredient_{new} \tag{3}$$

2.3 Reuse

In general, the reuse step involves three steps illustrated in figure 2: 1) Adaptation with the undesired query. 2) adaptation with the desired query and 3) suitable adaptation. The overall goal with the two first adaptation processes is to customize the retrieved recipes to better fit the user query. Suitable adaptation considers the modifications applied in the two first processes, examining whether more substitutions are necessary with the goal of making the recipe as a whole successful.

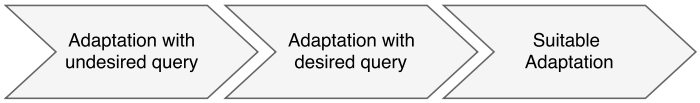

Fig. 2. Reuse process

Adaptation with the undesired query involves getting rid of undesired ingredients, while the goal with adaptation with the desired query is to add desired ingredients to recipes. These two adaptation processes are very similar. The difference is their opposite goals. Both processes loops through undesired/desired ingredients, respectively. For each ingredient, three substitution methods are considered: 1) Simple deletion/addition, respectively, 2) substitutions, and 3) similarity substitutions.

Deletion and addition rules are written in the *simple rule* format. When given requirements are satisfied, an ingredient may be deleted from/added to the recipe. Adaptation with the undesired query is carried out by deletion rules, while the adaptation with the desired query uses the addition rules.

Substitution rules are written in the *substitution rule* format. When some requirements are satisfied, a modification may be done to a recipe ingredient. That means, one ingredient is removed from the recipe and one ingredient is added. For adaptation with the undesired query, an undesired ingredient is removed and a *substitution alternative* is found. For adaptation with the desired

query, a desired ingredient is added and a *substitution offer* within the recipe is found.

Similarity substitutions are the last type of substitutions considered. These substitutions are based on taxonomies. The system fetches ingredients that are similar to the ingredient that is considered. For adaptation with the desired query, the goal is to find an ingredient that is similar within the recipe, then remove this and add the desired ingredient. For adaptation with the undesired query, a similar alternative is added, and the undesired ingredient is removed. However, the substitution only goes through if an *adaptation threshold* is satisfied. A threshold is set for each ingredient category. It defines how similar the substitution alternative and the considered ingredient needs to be for the substitution to go through.

After the undesired and desired adaptation process, the style of the recipe may have changed. The idea with suitable adaptation is to make the new ingredients fit the recipe better. The process starts by iterating all modifications of the recipe instance. For every new ingredient, the system checks whether a so-called *suitable rule* applies. Suitable rules are of the type *Simple Rule*, which means that there may be requirements for the rule to be valid. Both addition rules, substitution rules, and similarity substitutions are considered in the process. If several suitable ingredients are found, a random one among them is chosen.

Suitable adaptation also involves modifying titles. When ingredients are replaced by new ingredients in a recipe, the recipe title may be out of context. Two types of adaptation methods were implemented to rename the title to better fit the new recipe. First, the system considers pre-defined rules. For the record, the rules are called *title rules* and are of the type *Simple Rule*. The rule may have requirements to be valid. If no specific rules on ingredients apply, a different approach is considered. The method focuses on the previous title, and apply to titles containing ingredients. The system aims to match the content of the title with the substitutions that are carried out. If substituted ingredients are found in the title, the same substitutions are carried out in the title.

3 Evaluation

Several evaluation methods were assessed to conduct the system's evaluation and three evaluation goals were set: 1) evaluate the calculated similarity score, and hence, the order of the suggested recipes, 2) show that adapted recipes better match the user query, and 3) evaluate whether the user can distinguish the computer created recipes from the human created recipes.

For the first goal, we compared the system's ranking of recipes with ten test subjects ranking of recipes, for three given queries. The difference was measured by calculating the difference between each recipes ranking by the system and its ranking by the test subject. Per query, this resulted in a score from 0 to 25, where 0 represents equal ranking, and 25 represents the opposite order between the system and the test subject's ranking. On average, the result of this process showed that the discrepancy in ranking between the test subject's and the system

was 5.52 out of 25. Also, the test subject's and the system shared 3.57 out of 5 recipes as their top 5.

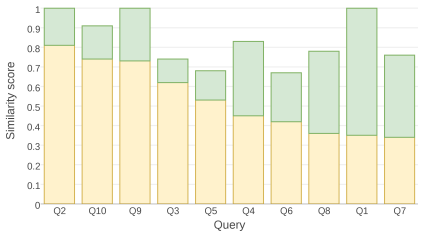

Fig. 3. Adaptation (green) changing the original similarity score (yellow)

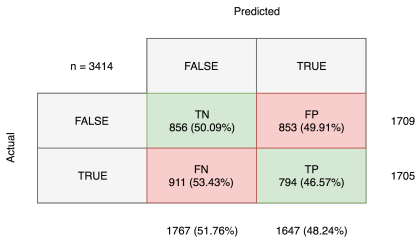

Fig. 4. *Bot or not* user predictions vs. actual recipe creations

For the second goal, a three step process were conducted: 1) doing a set of queries with the adaptation process turned off and note the similarity scores achieved for the top five results, 2) doing the same set of queries with the adaptation process turned on, and 3) compare the similarity scores. The evaluation results showed that the average similarity score increases for all the test queries. Figure 3 illustrates the evaluation results. In the figure, the yellow bars show the average similarity score for the top five results with no adaptation, while the green bars show the average increment for the top five results with adaptation on. On average, the mean similarity score for the top five recipes suggested per query increased with 0.32.

For the third goal, an online quiz was implemented[2]. The quiz displayed one recipe at a time for the user, and the user was to guess whether the recipe was human or computer created. The quiz ran for seven days (168 hours) and gathered in total 3414 responses distributed over 42 recipes. Figure 4 shows a confusion matrix of the quiz feedback. To clarify, *true* refers to an adapted recipe. Results showed that people guessed that a human had created the computer adapted recipes in 53.43% of the cases. Also, people recognized the original cases as created by a computer in 50.09% of the cases. This result reveals that most users are not able to distinguish the recipes from one another.

4 Discussion and Conclusion

Earlier participants in the CCC have chosen various approaches for their systems. All four systems presented involve a hierarchical taxonomy. Some systems generate substitution rules from their taxonomy or cooking communities, while JaDaWeb has a table of ingredients that can substitute each other across categories. However, none explicitly define removal, addition or suitable rules like

[2] `www.intellimeal.no/botornot`

implemented in IntelliMeal. The rules and the similarity substitutions complement each other's weaknesses. Together, the components result in comprehensive adaptation of recipes. The rules employed in IntelliMeal have the ability to serve both specific and general purposes. Incorporating taxonomies into the rule engine has enabled this. Also, rules may be specified for ingredients across attributes. The result is more radical modifications of recipes, where the general style of a recipe may change completely.

In conclusion, IntelliMeal, extends the traditional CBR cycle by adding a second retrieval from an ephemeral case base populated with cases from a multilayer rule engine. The adaptation mechanism gets creative in the way that it exploits domain knowledge defined in taxonomies to adjust similarity and to expand recipe case base. Therewith, we were able to achieve overall satisfying results with a limited case base. The measurable outcome of this project presented in section 3 is exceedingly satisfying. Adaptation of cases increased similarity scores for a given user query in all test cases, and humans had difficulties distinguish computer and human created recipes from one another.

References

1. Aamodt, A., Plaza, E.: Case-based reasoning: Foundational issues, methodological variations, and system approaches. Artificial Intelligence Communications 7(1), 39–59 (1994)
2. Badra, F., Cojan, J., Cordier, A., Lieber, J., Meilender, T., Mille, A., Molli, P., Nauer, E., Napoli, A., Skaf-Molli, H., Toussaint, Y.: Knowledge acquisition and discovery for the textual case-based cooking system wikitaaable. Computer Cooking Contest Workshop Proceedings (2009)
3. Ballesteros, M., Martín, R., Díaz-Agudo, B.: Jadaweb: A cbr system for cooking recipes. Computer Cooking Contest Workshop Proceedings (2010)
4. Blansché, A., Cojan, J., Dufour-Lussier, V., Lieber, J., Molli, P., Nauer, E., Skaf-Molli, H., Toussaint, Y.: Taaable 3: Adaptation of ingredient quantities and of textual preparations. Computer Cooking Contest Workshop Proceedings (2010)
5. Fuchs, C., Gimmler, C., Günther, S., Holthof, L., Bergmann, R.: Cooking cake. Computer Cooking Contest Workshop Proceedings (2009)
6. Gaillard, E., Lieber, J., Nauer, E.: Improving ingredient substitution using formal concept analysis and adaptation of ingredient quantities with mixed linear optimization. Computer Cooking Contest Workshop Proceedings (2015)
7. Herrera, J., Iglesias, P., Sánchez, A., Díaz-Agudo, B.: Jadacook 2: Cooking over ontological knowledge. Computer Cooking Contest Workshop Proceedings (2009)
8. Ihle, N., Hanft, A., Althoff, K.D.: Extraction of adaptation knowledge from internet communities. Computer Cooking Contest Workshop Proceedings (2009)
9. Minor, M., Bergmann, R., Görg, S., Walter, K.: Adaptation of cooking instructions following the workflow paradigm. Computer Cooking Contest Workshop Proceedings (2010)
10. Müller, G., Bergmann, R.: Cookingcake: A framework for the adaptation of cooking recipes represented as workflows. Computer Cooking Contest Workshop Proceedings (2015)
11. Newo, R., Bach, K., Hanft, A., Althoff, K.D.: On-demand recipe processing based on cbr. Computer Cooking Contest Workshop Proceedings (2010)