CBRkit Evaluation

Mirko Lenz, Lukas Malburg, and Ralph Bergmann

Case-Based Reasoning for Used Car Dealers

Application Scenario

A large car dealer with more than one million used cars has problems providing his customers with suitable car offers from this large amount of data. The dealer is not satisfied with the results of a standard keyword-based search and often must check many offers manually. Therefore, a case-based reasoning system should be used. For this purpose, the dealer provides its database with used car entries as a case base.

In the following, you will get some time to solve this task by setting up a CBR system for this purpose. On the one hand, you configure the already available ProCAKE framework based on the Java programming language, and on the other hand you will use the newly developed CBRkit framework based on Python. The goal of this evaluation is to assess how easy it is possible to solve the following tasks by the frameworks. For this purpose, you should solve the tasks and subsequently determine how easy it was. A Likert scale is used to assess several criteria for both frameworks.

ProCAKE Setup

We provide a ready-to-use IntelliJ project in the folder "~/IdeaProjects/procake-demo". The evaluation must be processed in the following packages:

- "/src/main/resources/de/uni_trier/wi2/procake_demos/domains/cars":
  Here you can find different case bases as well as the domain model and the similarity model.
- "/src/main/java/de/uni_trier/wi2/procake_demos/cars":
  Here you find the class "CarsRetrievalDemo.java" for similarity calculation.

In the following, you will find several tasks that you should work on. The structure of the tasks is given in the class "CarsRetrievalDemo.java". Do not change the paths or file names used there as well as the method signatures.

CBRkit Setup

We provide a ready-to-use PyCharm project in the folder "~/PycharmProjects/cbrkit-demo". You will work in the file "cbrkit_demo/retrieval.py". The case base and the taxonomies are stored in the "data" directory.

TASK 0 (INTRODUCTION) MAX. 5 MIN. FOR EACH FRAMEWORK

ProCAKE and CBRkit each come with a user documentation. For easier access, links to the respective sites have been added to your taskbar. Open both manuals and browse their contents.

TASK 1 MAX. 5 MIN. FOR EACH FRAMEWORK

Both projects already contain a retriever which allows to perform similarity computations in a case base. To use it, a query is required. Your task is to define suitable attributes within the query to define concrete requirements for cases from the case base. *Design your request so that red mini-vans ("mini-van") from the year 2010 are retrieved.*

Afterward, run the retriever with the query you created. Ensure that the 10 best results with the highest similarity are displayed.

Notes for ProCAKE: Open the class "CarsRetrievalDemo.java". The retriever is defined in the method "runTask" and the (currently empty) query in the method "createDefaultQueryObject". You can find the structure of the object in the XML file "model.xml" (the domain model/vocabulary), which is stored in the resources' path.

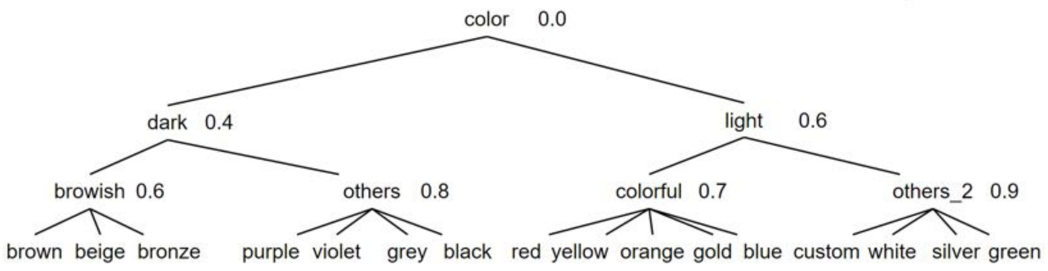
Notes for CBRkit: Open the file "retrieval.py". The retriever is defined in the method "run_default" and the (currently empty) query in a global object called "query".

TASK 2 MAX. 10 MIN. FOR EACH FRAMEWORK

With the query being the defined, the retrieval can be performed. Some similarity measures are already defined in the projects, but two are missing. Please add them.

a) **Attribute *year*:** Create a local similarity function and model the local similarity measure so that the similarity is 1.0 if the difference between the year of manufacturing of the query is lower (not equal) than 10 compared to the year of manufacturing of the case. If the difference is greater than or equal to 10, the similarity measure should decrease linearly until the similarity value is 0.0 at a difference of more than 50.

b) **Attribute *paint_color*:** Create a local similarity function and model the local similarity measure so that the given taxonomy is realized. Use the strategy "average" to compute similarities of inner nodes. **Please note:** The taxonomy is already defined for both frameworks, your task is to assign the weights as shown in the figure below.



Notes for ProCAKE: To calculate the similarities, the xml file "sim.xml" is used, which is in the resources' folder. In this file, specify local similarity measures for the values listed above. In this context, it is important to note that as strategy for inner nodes in case and inner nodes in the query, the average strategy should be used.

Notes for CBRkit: The object "global_similarity" contains all similarity measures and the aggregation function. In this context, please use the average strategy for the taxonomy.

TASK 3 MAX. 20 MIN. FOR EACH FRAMEWORK

Implement a MAC/FAC retrieval in which

1. a very simple similarity measure is used as a filter in the MAC phase, and
2. the normal similarity measure is applied in the FAC phase.

For this purpose, use a similarity function in the MAC phase that only consists of the local similarity measure for *paint_color* and filter for cars with red color (i.e., the MAC phase should only return cases having a similarity of exactly 1.0). For the subsequent FAC phase, calculate the similarity for the remaining cases based on all attributes with the similarity measures created in the previous tasks.

Notes for ProCAKE: There is a boilerplate method "runMACFACTask" which you can extend for this task.

Notes for CBRkit: There is an empty method "run_macfac" which you should use for this task.

BONUS TASK MAX. 15 MIN. (CBRKIT ONLY)

Currently, the attribute *miles* is not part of the retrieval. To change that, please complete the following two tasks:

a) **Query:** Add the attribute *miles* with a value of 9000 to the query.

b) **Similarity measure:** Add a custom similarity measure that implements the following function:

$$\mathrm{sim}(q,c) = \begin{cases} 1.0, & |q-c| < 10000 \\ 0.7, & 10000 \le |q-c| < 25000 \\ 0.0, & |q-c| \ge 25000 \end{cases}$$