The UniGe Experiments on the Search for Earlier Patents

Jacques Guyot, Gilles Falquet, Karim Benzineb

Computer Science Center, University of Geneva - Route de Drize 7, 1227 Carouge, Switzerland

jacques.guyot, gilles.falquet@unige.ch; karim@simple-shift.com

Our goal was to retrieve all the patents related to a given patent (the topic) in a corpus of about two million patents. Each patent had on average less than 10 quotes to find. In this experiment we used the classical, cosine-based approach to calculate the similarity.

From the original corpus we extracted the following elements for each patent:

- The Applicant and Inventor fields;
- The invention Title in English, French, and German (if existing);
- The invention Abstract in the three languages (if existing);
- The invention Claims in the three languages (if existing).

As for the topics (*i.e.* the patents whose quotes we were looking for), we kept all the fields.

Although the corpus was multilingual, we evaluated our tools in English only and in the XL mode (10'000 test patents). We performed five runs and experimented several weight evaluation methods for the cosine-based approach (TF*IDF, OKAPI, FAST). We also tested a filtering process on the document length (since some documents were very short) and a filtering process on the patent class.

For the class filtering, we used an automated supervised classifier to assign one or several IPC category (a the "subclass" level) to the topic and we built a catalog of the categories which were assigned to each corpus document. The documents which were retrieved by the cosine method but which did not have any common category with the topic were filtered out. We obtained the following results:

Clepip-unige RUN1 (map 10.52%): Weighting: **TF*IDF**, Length Filtering: **yes**, Class Filtering: **yes**

Clepip-unige RUN2 (map 10.58%): Weighting **TF*IDF**, Length Filtering: **yes**, Class Filtering: **no**

Clepip-unige RUN3 (map 10.52%): Weighting **TF*IDF**, Length Filtering: **no**, Class Filtering: **yes**

Clepip-unige RUN4 (map 7.49%): Weighting: **OKAPI**, Length Filtering: **yes**, Class Filtering: **yes**

Clepip-unige RUN5 (map 7.61%): Weighting: **FAST**, Length Filtering: **yes**, Class Filtering: **yes**

We were rather disappointed to note that the class filtering did not help to eliminate the noise. This filtering method is highly efficient when the query is short. However, in this case the query was a whole patent, so the classification filtering did not bring any improvement since the cosine-based similarity calculation acted implicitly as a *k*NN (*k* Nearest Neighbours), which is itself an alternative to automated classification.

As for the length filtering, it did produce a small performance improvement during the test runs but it seemed to have no effect at all in the experiment runs. We also performed additional runs with other weighting methods but all their results were below those of the classical approach.

Still, this experiment showed that our tools are efficient to process this type of tasks: the processing time was about one second per topic on a desktop PC(Intel Q9550-2.8Ghz/8GB RAM). On the other hand, the class filtering approach (as we implemented it) did not allow filtering out the noise because many documents use the same words to describe inventions with differing aspects but relating to the same classes.