myClass: A Mature Tool for Patent Classification

Jacques Guyot[1], Karim Benzineb[1], Gilles Falquet[2]

[1] Simple Shift,
Ruelle du P'tit-Gris 1, 1228 Plan-les-Ouates, Switzerland

[2] Computer Science Center, University of Geneva,
Route de Drize 7, 1227 Carouge, Switzerland

{jacques, karim}@simple-shift.com, gilles.falquet@unige.ch

Abstract

In this task 2,000 patents in three languages (English, French and German) were to be classified among approximately 600 categories. We used a classifier based on neural networks of the Winnow type. This classifier is already used for similar tasks in professional applications. We tested three different approaches to improve the classification accuracy: the first one aimed at solving the issue of poorly-documented categories, the second one was meant to enrich the overall training corpus and the third one was based on the processing of the corpus' collocations. Although we ranked first in this competition, none of the three approaches mentioned above provided for a clear improvement in classification accuracy; our results were essentially due to the implementation of the classification algorithm itself.

Introduction

When we first started studying the field of patent classification back in 2002, we were driven to choose a Winnow-type algorithm (training-based neural networks), for reasons which are explained in our first article [1]. Other possible choices included in particular the *k*NN and the SVM algorithms. *k*NN algorithms perform well but they are very slow to come up with a prediction. This is because they don't include any training phase; all the similarity calculations are performed at runtime, when a new document is to be classified. SVM algorithms also perform well from the point of view of classification accuracy, but they are slow to train. As for neural networks, they are relatively fast to train and fast to come up with predictions, and they have good performances. However, their parameters are slightly more complex to optimize. The first version of our classifier was developed in partnership with the University of Geneva [2]. The following versions were built internally at Simple Shift.

Classifying at finer-grain levels (Main Groups or Sub-groups) requires an interaction with the user, who must be allowed to shift from a level to the next one as required. Thus we had to build several hundreds of neural networks to allow for all the possible shifts. It quickly became obvious that the efficiency of the building process was a key factor of success to meet professional requirements. Being fast at building neural networks also facilitated the parameter tuning phase.

Patent classification requires to index very large training corpora (in this experience, the size of the raw corpus was 85 Gb). Thus we linked our classifier to a well-performing home-made indexer [3] and search engine. Both of those tools have functions which are specifically adapted to support the classifier, in particular through various linguistic processing such as pattern or collocation detection.

The classification task requires to work on a large number of features (the terms), some of which may turn out to be useless. We used an algorithm of the Balanced Winnow type [4] and applied the extensions described in [5]. Our main contribution was in the way we implemented the algorithm in terms of compression, optimization and parallelization.

Beyond its professional applications, the classifier and the indexer were used in academic research, including for thesaurus analysis [6] and prior art search in the field of patents [7], or for disambiguation tasks in the field of information retrieval [8].

A general introduction to patent classification can be found in [9].

The experiment and its resources

Patent classification may be seen as the task of affecting a category (a code) to the description of an invention (a text). In the case at hand, the classification used was the one defined by the World Intellectual Property Organization (WIPO), which is known as the International Patent Classification (IPC). It is a tree classification composed of five levels, each lower level containing finer-grain categories.

The Advanced Level version of the IPC (2009) has the following tree structure: 8 Sections, 129 Classes, 639 Sub-classes, 7,352 Main Groups and 61,847 Sub-groups. In the experiments described below, classification was performed at the Sub-class level.

The training corpus contained about 2.6 million documents covering about 1.3 million patents. One patent could be linked to several documents corresponding to various stages of the patent filing process. The documents bore a code such as A1, A2, etc. or B1, B2, etc. The A code means the patent application is being examined, while the B code means the patent is granted.

A test and evaluation set containing 2,000 patents was also circulated. Those patents belonged to the A category.

The training and testing sets contained patents in French, English and German. Some documents were monolingual, while others contained two or even three languages. At least 150,000 patents were available for each language. All patents were in XML format, so their information content was well structured.

Each document of the training corpus contained an indication of the category, or categories, it was affected to.

The task was to affect a code to each patent in the test corpus. The maximum number of predictions allowed was 1,000, which was greater than the number of existing categories at Sub-class. The challenge was of course to identify all the *correct* categories of a given test patent.

Corpus processing

Previous experience had shown that working on the complete patent description actually degraded the classifier's performance. Indeed, in some cases this description may be very large and include up to several hundreds of thousands of characters. Thus we decided to keep only the first 4,000 characters of the full-text description. The following patent fields were selected for indexing:

- Inventor
- Applicant
- Title
- Abstract
- Claims
- Description (only the first 4,000 characters).

On the basis of the initial corpus, we built a catalogue which listed, for each patent of the training corpus, all the IPC categories to which it was affected. We used the <classification-ipcr> and <classification-ipc> tags to identify those categories. Although we noticed that the various documents linked to a given patent were sometimes affected to differing categories, we did not harmonize the catalogue (*i.e.* so that a given patent would have been affected to streamlined categories). Therefore the data in our catalogue was of the following type:

```
EP-0000001-A1 H04L F25B F28D B27F G06F G11B B23P
EP-0000001-B1 B27F H04L F25B F28D F24J G06F G11B
B23P
EP-0000002-A1 C07D H04L A01N
EP-0000002-B1 C07D H04L A01N
EP-0000003-A1 E05B
EP-0000004-A1 A47J B04B
...
```

Language processing

In our previous experiments over multilingual corpora, the results had shown that building a classifier for each separate language did not provide better performances than building a single, multilingual classifier. In fact, mixing all the languages could even improve the performances because some technical terms could be shared between several languages (for example the word "laser"), and thus increase the number of available examples. Thus we decided to build a single classifier which was trained over all the languages.

Although German is an agglutinative language, we did not use any specific linguistic processing because our experience had shown that when the training corpus is large enough, those types of processing don't make much of a difference. (However when the corpus is small we do use 5-gram indexing on German).

We provided a list of stop-words for each of the three languages so as to eliminate from the index all the words which have a low classifying power (articles, prepositions, etc.)

Objective of the experiment

The objective of our experiment was to systematically test the impact of three improvement tracks on the precision of the classification:

Extending the training corpus: We added a collection of 1.6 million patents, covering the three languages, to the initial training corpus of 2.6 million patents. (This method is called "External_Collection" below).

Balancing the distribution of training examples across the categories: In the categories which had the smallest numbers of training patents, we copied the available examples a number of times until we reached a minimum threshold. (This method is called "Over_Sampling" below).

Identifying collocations in the patent texts: In all the training examples, collocations (defined here as a single concept described by two consecutive words, such as "spark plug") were automatically identified and indexed as a single feature. (This method is called "Collocations" below).

Indexing and collocation extraction

The first step was to index the corpus. The final index included over 5 billion terms, which were composed of over 14 million different terms.

The relatively high number of different terms is explained by the fact that the corpus included three languages with a large technical terminology (chemistry, biology, electronics...) The corpus also contained a number of typos.

We filtered out the words which occurred less than 4 times. After that operation, only about 3 million different terms remained available to train the classifier.

The extraction of collocations was performed on that filtered index. For each indexed term, our index stores not only the document where the term was found, but also its position in the text. Thus identifying a collocation comes down to asking the index the following question: "For terms X and Y, how many documents include term X in position *n* and term Y in the following position?" The choice of terms X and Y is made among the 3 million filtered terms, which amounts to $9x10^{12}$ possibilities. However, a large number of possibilities may be eliminated by testing only the occurrences of X and Y which actually appear in the corpus (in this case, it reduced the calculation space to 5 billion possibilities).

The process of collocation extraction was the most calculation-intensive in the entire classification task: it took a whole day and added about one million new terms to the corpus. We decided to select only the collocations which occurred more than 16 times in the corpus. We also kept the collocations which included a stop-word (such as *"pomme de terre"*, which is "potato" in French).

Results of the experiment

We performed eight runs so as to systematically test all the possible permutations in our planned tests. C0 is the baseline run, i.e. the only one which is strictly based on the CLEF-IP corpus with no additional processing. Practically all the runs got an 83% success score when asked to find a patent category with a single prediction (P_1 measure), and 93% of the categories were quoted in the top 25 predictions (R_25 measures).

Our results are the following :

(MAP in %, P_1 in %) (E=External_Collection, O=Over_Sampling, C=Collocation)

	Map%	P_1%
c0():	78.69	83.35
c4500(O):	77.82	81.90
ce0(E):	79.10	83.35
ce6800 (EO):	78.61	83.00
cc0(C):	79.16	83.55
cc4500(OC):	78.86	83.05
cec0(EC):	79.51	83.50
cec6800 (EOC):	79.34	83.50

Comments

- None of our approaches managed to improve markedly the P_1 score. This shows that most of the classification precision comes from the implementation of the Winnow algorithm it-self.
- Smoothing out the distribution of training examples by using the over-sampling approach has in fact slightly degraded the performance. However, it did improve the classification into poorly-fed categories (but this criterion was not taken in ac-count in the competition).
- Doubling the number of examples did not improve much the performance.
- Using collocations did allow to add relevant information and somewhat improved the performance.

System Efficiency

All the experiments were performed on a workstation which cost less than 1,000 euros. The processing times of the classification task (without the collocation extraction) were the following:

- 2 hours to index the corpus
- 1 hour to train the application (for each run)
- 3 minutes to classify the 2,000 topics.

This implementation of the classifier was parallelized and used the 4 cores of the processor, thus making the most out of the multicore architecture. It should therefore be possible to reach even higher performances with the new AMD processors which are built on 12 cores.

The final neural network built for the C0 (baseline) experiment had a size on disk smaller than 100 Mb. This allows to distribute it, along with the rest of the classifier, on a simple CD-ROM (this is sometimes required by some of our customers).

Conclusion

Our approach is to avoid, as much as possible, any linguistic processing which is language-dependent so as to guarantee similar performances for all languages (including, for example, Spanish, Russian and Chinese). To make up for the possible shortcomings of this approach, we chose to remain as exhaustive as possible and to apply the minimum number of filtering-out processes.

Such a strategy seemed to be efficient in this competition since our results were ranked in first position.

The three approaches tested in this experiment either slightly degraded or slightly improved the baseline performance of the classifier. Thus the good performance was essentially due to our implementation of the Winnow algorithm. "myClass" has obviously evolved over the past eight years in professional contexts and might now be considered a mature classifier.

Bibliography

[1] Fall CJ, Benzineb K. Literature survey: Issues to be considered in the automatic classification of patents, WIPO, 2003.

[2] C. J. Fall, K. Benzineb, J. Guyot, A. Törcsvéri, P.Fiévet. Computer-Assisted Categorization of Patent Documents in the International Patent Classification, *in* Proceedings of the International Chemical Information Conference (ICIC'03), Nîmes, France, October 2003.

[3] Guyot, J., Falquet, G., Benzineb, K. Construire un moteur d'indexation. Technique et science informatique (TSI), Hermes, Paris. 2007 (*in French*).

[4] Littlestone, N. Learning Quickly When Irrelevant Attributes Abound: A New Linear-Threshold Algorithm. Mach. Learn. 2, 4 (Apr. 1988), 285-318.

[5] Ido Dagan , Yael Karov , Dan Roth. Mistake-Driven Learning in Text Categorization, *in* EMNLP-97, The Second Conference on Empirical Methods in Natural Language Processing, 1997.

[6]Nogueras-Iso, J., Lacasta, J., Falquet, G., Guyot, J., Teller, J. Ontology learning from thesauri: An experience in the urban domain, *in* Gargouri, F. and Jaziri, W. (Eds.) Ontology Theory, Management and Design: Advanced Tools and Models. IGI Intl publishing. (to be published in 2010).

[7] Jacques Guyot, Gilles Falquet, Karim Benzineb. UniGE Experiments on Prior Art Search, Lecture Notes in Computer Science (LNCS), Springer (to be published in 2010).

[8] Guyot, J., Falquet, G., Radhouani, S., Benzineb K. Analysis of Word Sense Disambiguation-Based Information Retrieval. In 8th Workshop of the Cross-Language Evaluation Forum, CLEF 2008, Revised Selected Papers. C. Peters, et al. (Eds.). Lecture Notes in Computer Science (LNCS) 5706, Springer, 2009.

[9] Guyot, J., G., Benzineb. Automated Patent Classification, *in* Current Challenges in Patent Information Retrieval , Springer (to be published in 2011).