Combining document representations for prior-art retrieval

Eva D'hondt, Suzan Verberne
Information Foraging Lab, Radboud University Nijmegen
e.dhondt@let.ru.nl, s.verberne@cs.ru.nl

Wouter Alink, Roberto Cornacchia
Spinque
wouter,roberto@spinque.com

Abstract

In this paper we report on our participation in the CLEF-IP 2011 prior art retrieval task. We investigated whether adding syntactic information in the form of dependency triples to a bag-of-words representation could lead to improvements in patent retrieval. In our experiments, we investigated this effect on the title, abstract and first 400 words of the description section. The experiments were conducted in the Spinque framework with which we tried to optimize for the combinations of text representation and document sections. We found that adding triples did not improve overall MAP scores, compared to the baseline bag-of-words approach but does result in slightly higher set_recall scores. In future work we will extend our experiments to use all the text sections of the patent documents and fine-tune the mixture weights.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: H.3.1 Content Analysis and Indexing; H.3.3 Information Search and Retrieval

General Terms

Retrieval, Text representation

Keywords

Prior Art Search, Patent retrieval, CLEF-IP track, dependency triples

1 Introduction

In this paper, we describe our contribution to the prior-art retrieval track which was organised in the context of the Intellectual Property (IP) benchmark at the CLEF 2011 Labs. For the third year in a row, we focused on improving patent retrieval with the aid of syntactic-semantic information in addition to the baseline bag-of-words approach. In a follow-up study to last year's participation in the CLEF-IP classification track [11], Koster et al.[6] found that patent classification can be improved by adding syntactic phrases in the form of dependency triples, to a bag-of-words representation. This year we were interested to see if a similar improvement could be found for prior art retrieval, by adding dependency triples to the words. In the CLEF-IP 2010 prior art track the Hildesheim team found that using (statistical) phrases as index units leads to a better improvement for titles than for the other document sections [3]. Keeping this in mind, our aim for this year's track was two-fold: (a) Examining the added value of using dependency triples on top

of words in prior-art retrieval; (b) Optimizing the combination of the different documents sections and text representations.

2 Data Description

The CLEF-IP 2011 corpus, a part of the MAREC collection, was provided by the IRF [1] and contains approximately 3 million documents, pertaining to more than 1 million patents[1]. Most documents (2.6 million) came from the European Patent Office (EPO) and a smaller subset (around 400,000) consisted of patent documents from the World Intellectual Property Organization (WIPO). The patent documents were stored in the IRF XML format [5]. A patent document contains metadata such as name of inventor, IPC-R code, date of application, ... as well as (a mixture of) English, German or French text sections for the title, abstract, claims and/or description sections of the patent. In our experiments we only used the English text sections and IPC-R codes. The organizers distributed a training set of 300 patents and –unlike the previous years– only one topic set containing 3973 documents.

3 Experimental Set-up

3.1 Patent Section Extraction

Using a perl script we extracted the English title, abstract, claims and description sections from the original XML files. We also saved the first 400 words of the description sections and the IPC-R codes[2] separately. All the sections were saved as plain text in temporary text files. If a document did not contain a section or if –according to the XML tags– the section was not in English, no corresponding text file was created. The XML documents contain many text-internal XML tags that indicate figures, references, formulae, etc. in the original patent document. All such tags and the texts that they enclose were filtered from the text using a perl script.

3.2 Patent Parsing

In a preprocessing step the image references and claims headers in the text were removed using the regular expressions described by [9] in order to facilitate syntactic parsing of the claims and description sentences. We then sentenced the remaining text using a Perl script and knowledge of most common abbreviations in patent texts. The sentences in the resulting text files were parsed using the AEGIR dependency parser [8, 10], version 1.8.2. One of the AEGIRs output formats is a dependency representation which is comparable to the Stanford typed dependencies [4], in the sense that it generates a set of binary relations between words for an input sentence, thereby converting some function words (such as prepositions) to relations. In addition to that, AEGIR performs a number of normalizing syntactic transformations, such as passive-to- active transformation.

Because of the large amount of data we used a time constraint of maximum 1 second per sentence. This resulted in a loss of parsing output that differed somewhat between the separate sections.

Due to the sheer size of the corpus we were not able to completely parse the description and claims sections of the entire corpus within the given time. We therefore had to limit our experiments on the impact of triples to the title, abstract and first 400 words of the description. The keywords used for the bag-of-words component in the experiments were extracted from the title, abstract and the full description.

[1] Please note the difference between a patent and a patent document: a patent is not a physical document itself but a name for a group of patent documents that have the same patent ID number.

[2] We used the full IPC-R code up to the level of the subgroups, e.g. A01J 5/01.

[3] Parsing output from these sections was incomplete for the whole corpus and not used for the subsequent experiments.

Table 1: Estimate of missing parsing output – measured on representative set of 1000 documents

	title	abstract	description-400	*claims*[3]	*description*[3]
nr. of doc that contain section in EN	997	378	388	541	388
nr. of sentences	997	1323	5699	6508	68969
av.sentence length	8.32	33.58	27.63	52.25	30.20
% of unparsed sentences	0.00	0.01	0.02	1.2	0.03

3.3 Spinque Framework

We modeled and executed our runs as *search strategies* within the Spinque framework [2]. This is a prototype interactive retrieval environment where search processes are divided into two phases: the search strategy definition and the actual search.

The framework has a GUI-based drag-and-drop strategy editor which allows the user to construct the search strategies as graph structures, where edges represent data-flows consisting of terms, documents (e.g. patent-documents), document-sections (e.g. invention-title, abstract, description) and named entities (e.g. patents, IPC-R codes, companies). The nodes connected by such edges are pre-defined, general-purpose operational blocks, that either provide source data (the patent corpus and the topics corpus) or modify their input data-flow by applying operations such as selection based on IPC-R classes, extraction of specific sections from documents, or ranking of sections and documents, to name a few.

Search strategies defined in this framework are automatically translated into a probabilistic relational query language and executed on top of an SQL database engine. The ranking scores that are used as the basis for the probabilities were calculated with the Okapi BM25 ranking algorithm.

3.4 Experiments

3.4.1 Query term selection

This year, we performed query term selection on the triples, based on their relevance for a specific IPC-R class. The LCS software [7] that we used for the classification track builds class profiles which contain the term distribution (word and dependency triples) per IPC-R class. We extracted the subset of dependency triples that were most informative for correct classification, namely the top 25% of the triples ranked on their Winnow scores, from last year's class profiles and used them to filter the topic triples. Some class profiles for smaller IPC-R classes did not contain many triples (< 1000). In these cases all triples that contributed to classification were extracted. The aim of this filtering step is to remove the noisy, less informative topic triples from the query thus improving precision. Since a patent document is usually labeled with not just one single IPC-R code but rather belongs to multiple categories (on average a patent document contains 3 different IPC-R codes (on subclass level), the filtering is not so severe that it weeds out the individual differences between topic patent documents. In other words, the individual filtered topic documents are still very different from one another due to the relatively large subsets of terms from the class profiles that were used as filters and the different combinations of IPC-R classes per document. The filtering step reduced the average number of triples per topic document (over all sections) from 180 to 60.

3.4.2 Strategy building

The search strategies were constructed and evaluated in Spinque's strategy builder interface. Our strategies consisted of two steps: (1) As in last year's approach we first filtered the corpus on the IPC-R codes of the topic document to create a subcorpus per topic document that contains documents with at least one IPC-R class in common with the topic document; (2) Terms (words and/or triples) from the sections in the topic documents were then used to query the respective

sections of documents in the subcorpus. We did not perform any term selection for the bag-of-words approach. The resulting document lists were then merged into a larger results list. The ranking in that list depended on the documents scores (BM25 scores from their separate runs) multiplied by the weights given to each results list in the configuration. An example of a search strategy used in this track is shown in figure 1.

3.4.3 Determining the weighting configuration

The mixture weights in the Spinque framework allow for a reranking step while merging the result lists of the runs with individual sections. Finding the optimal mixture weights is a very time-consuming process, because of the large parameter space. Due to time constraints we were not able to train on many coefficient combinations for the mixtures. We used two different approaches to determine the weight configurations used in the submitted runs: (a) Normalisation over retrieval scores of individual sections; and (b) trial-and-error weighing.

3.4.3.1 Determining the relative importance of different sections

The mixture coefficients for the combinations of different text sections were found by running a subset of the training set topics on the respective text sections, that is, evaluating the title, abstract and descriptions sections independently from one another. We then took the Mean Average Precision (MAP) scores of these runs, normalised them to sum up to 1 and used the resulting ratios as coefficients for the mixtures.

Table 2: MAP scores – mixture coefficients for sections mixtures

	words_only		triples_only	
	MAP	coeff	MAP	coeff
title	0.0607	0.4	0.0109	0.2
abstract	0.0586	0.38	0.0334	0.62
description	0.0342	0.22	N/A	N/A
description-400	N/A	N/A	0.0097	0.18

3.4.3.2 Determining the relative importance of triples and words in the combined runs

The coefficients for mixing the words_only and triples_only runs were found using the 'trial and error' method on the training set. Starting from a 50/50 combination we used binary search to arrive at the optimal configuration: a words_only (0.8) and triples_only (0.2) combination.

3.4.4 Submitted runs

We chose to submit four separate runs:

1. **triples_only**: A baseline run to gauge the impact such precise index terms as Dependency triples can have on retrieval.
2. **Words_only**: A standard bag-of-words baseline run. Keywords were stemmed using the Porter stemmer (version 1).
3. **Combination_1**: Combining the results list of the words_only (stemmed) and triples_only (unstemmed) runs in a 80/20 configuration.
4. **Combination_2**: Even though triples are lemmatized by the parser, the patent domain consists of many highly specialized subdomains which deploy their own jargon. Consequently the patent documents usually contain a lot of words which may not feature in the parser lexicon [10]. The AEGIR parser recognises these words using robust rules which lead to

good estimates of POS tags (important for correct syntactic analysis later on) but applies no lemmatisation beyond the basic singular-plural differences. We therefore submitted an extra run to examine the impact of stemming of the triples.

4 Results

In this section we present the results of our submitted runs in terms of MAP, Precision and Recall for the general (Table 3) and English language-specific test data (Table 4).

5 Discussion

5.1 Impact of dependency triples on retrieval

As expected, triples by themselves are too specific to be used for retrieval: the triples_only run achieved a very high set_precision but fairly low set_recall. On average, only 250 documents were retrieved per topic document in this run. The MAP scores for the different sections on a subset of the training data in table 2 show decided differences between the sections.

However, in the combination runs, merging the triple_only and the bag-of-words result lists presented some interesting results: While dependency triples are usually seen as a way of improving ranking, we achieved the highest set_recall scores (measured with the language-specific English relevance assessments) compared to the other participants. An analysis of the result list of the combination_1 run shows that around 5% of relevant documents retrieved in this run (2.5% of all the relevant patents) were found using triples, but were not found in the words_only approach. This may show that using dependency triples, i.e. information which abstracts away from the surface form of the sentence, can contribute to retrieval where a bag-of-words approach falls short. However, at this point, the contribution is very small. An alternative explanation is that the dependency triples have improved the ranking of documents in the results list that fell underneath the cut-off point of retrieving 1000 patents per query in the words_only run. In which case, there is a complete overlap between the results from the triples_only run and the documents found by the words_only approach and the improvement in set_recall score for the combined is an artefact of our choice of threshold.

Furthermore, another 36% of the relevant documents in combined_1 run were found by both the words and triples approaches. We would expect these documents to feature high in the combined results list thus improving the MAP score (compared to the words_only run). However, we did not find much difference in the rankings and a slight *decrease* in MAP score. We expect that finetuning the 80/20 words-triples mixture coefficients on a held-out set of the test corpus may improve the rankings.

In the combination_2 run we experimented to try and raise recall by using stemming in the triples as well in the keywords, but we found that precision suffers much in that trade-off: While we did find more relevant documents, they were all pooled at the bottom of the results list. Moreover, the MAP score was significantly lower than for the combination_1 run. It is clear that the mixture weights should be tuned separately for combinations with stemmed triples.

5.2 Impact of the different sections

We did not have the opportunity to examine the impact of the different sections in much detail. Rather we focussed on optimising the impact of those sections were dependency triples were the most successful in their own right (see section 3.4.3). However, this independency assumption is problematic: While it was a good starting point, namely in the mixtures the most weight was given to those sections that were most likely to have relevant documents high in the list, this strategy cannot properly account for interaction between sections and suffers from the uneven distribution of (English) text data in the corpus. In future work we will use further tuning via trial and error method to try and find a local if not global optimum.

6 Conclusion

In our participation to the CLEF-IP11 prior art retrieval track we examined the impact of adding dependency triples obtained with the AEGIR parser to a bag-of-words approach. Triples by themselves are very specific terms, as reflected by the high precision score achieved in the triple_only run. Interestingly, we found that adding triples lead to a slight improvement in recall, rather than in precision as we had expected. It is not quite clear if this is due to the normalisation features of triples or an indirect effect of their higher precision. We also experimented with stemming of the triples, but this led to a severe loss of precision. In future work we will extend our experiments by adding data of all the description sections and the claims section, both for the words en triples approach. We will also keep working on tuning the mixture coefficients by a 'trial-and-error' method, rather than basing the coefficients on individual retrieval performance of the sections.

References

[1] Home - IRF. http://www.ir-facility.org/.

[2] W. Alink, Roberto Cornacchia, and Arjen de Vries. Searching clef-ip by strategy. In Carol Peters, Giorgio Di Nunzio, Mikko Kurimo, Thomas Mandl, Djamel Mostefa, Anselmo Peas, and Giovanna Roda, editors, *Multilingual Information Access Evaluation I. Text Retrieval Experiments*, volume 6241 of *Lecture Notes in Computer Science*, pages 468–475. Springer Berlin / Heidelberg, 2010. 10.1007/978-3-642-15754-7_56.

[3] Daniela Becks, Thomas Mandl, and C. Womser-Hacker. Phrases or Terms? The Impact of Different Query Types. In *Proceedings of the Conference on Multilingual and Multimodal Information Access Evaluation (CLEF 2010), CLEF-IP workshop*, page 99, 2010.

[4] Marie-Catherine de Marneffe and Christopher D. Manning. The Stanford typed dependencies representation. In *Coling 2008: Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation - CrossParser '08*, number ii, pages 1–8, Morristown, NJ, USA, 2008. Association for Computational Linguistics.

[5] IRF. Clef Ip 2011 Track Guidelines. Technical report, IRF, 2011.

[6] Cornelis H.A. Koster, Jean G. Beney, Suzan Verberne, and Merijn Vogel. Phrase-Based Document Categorization. In W. Bruce Croft, Mihai Lupu, Katja Mayer, John Tait, and Anthony J. Trippe, editors, *Current Challenges in Patent Information Retrieval*, volume 29 of *The Kluwer International Series on Information Retrieval*, pages 263–286. Springer Berlin Heidelberg, 2011.

[7] Cornelis H.A. Koster, Marc Seutter, and Jean G. Beney. Multi-classification of patent applications with Winnow. In *Perspectives of Systems Informatics, 5th International Andrei Ershov Memorial Conference*, pages 546–555, 2003.

[8] Nelleke Oostdijk, Suzan Verberne, and Cornelis Koster. Constructing a broad-coverage lexicon for text mining in the patent domain. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC 2010). European Language Resources Association (ELRA)*, 2010.

[9] Peter Parapatics and Michael Dittenbach. Patent claim decomposition for improved information extraction. In W. Bruce Croft, Mihai Lupu, Katja Mayer, John Tait, and Anthony J. Trippe, editors, *Current Challenges in Patent Information Retrieval*, The Kluwer International Series on Information Retrieval. Springer Berlin Heidelberg.

[10] Suzan Verberne, Eva D'hondt, Nelleke Oostdijk, and Cornelis Koster. Quantifying the challenges in parsing patent claims. *Proceedings of the 1st International Workshop on Advances in Patent Information Retrieval (AsPIRe 2010)*, pages 14–21, 2010.

[11] Suzan Verberne, Merijn Vogel, and Eva D'hondt. Patent classification experiments with the Linguistic Classification System LCS. In *Proceedings of the Conference on Multilingual and Multimodal Information Access Evaluation (CLEF 2010), CLEF-IP workshop*, number Section 2, page 49. Sl: sn, 2010.

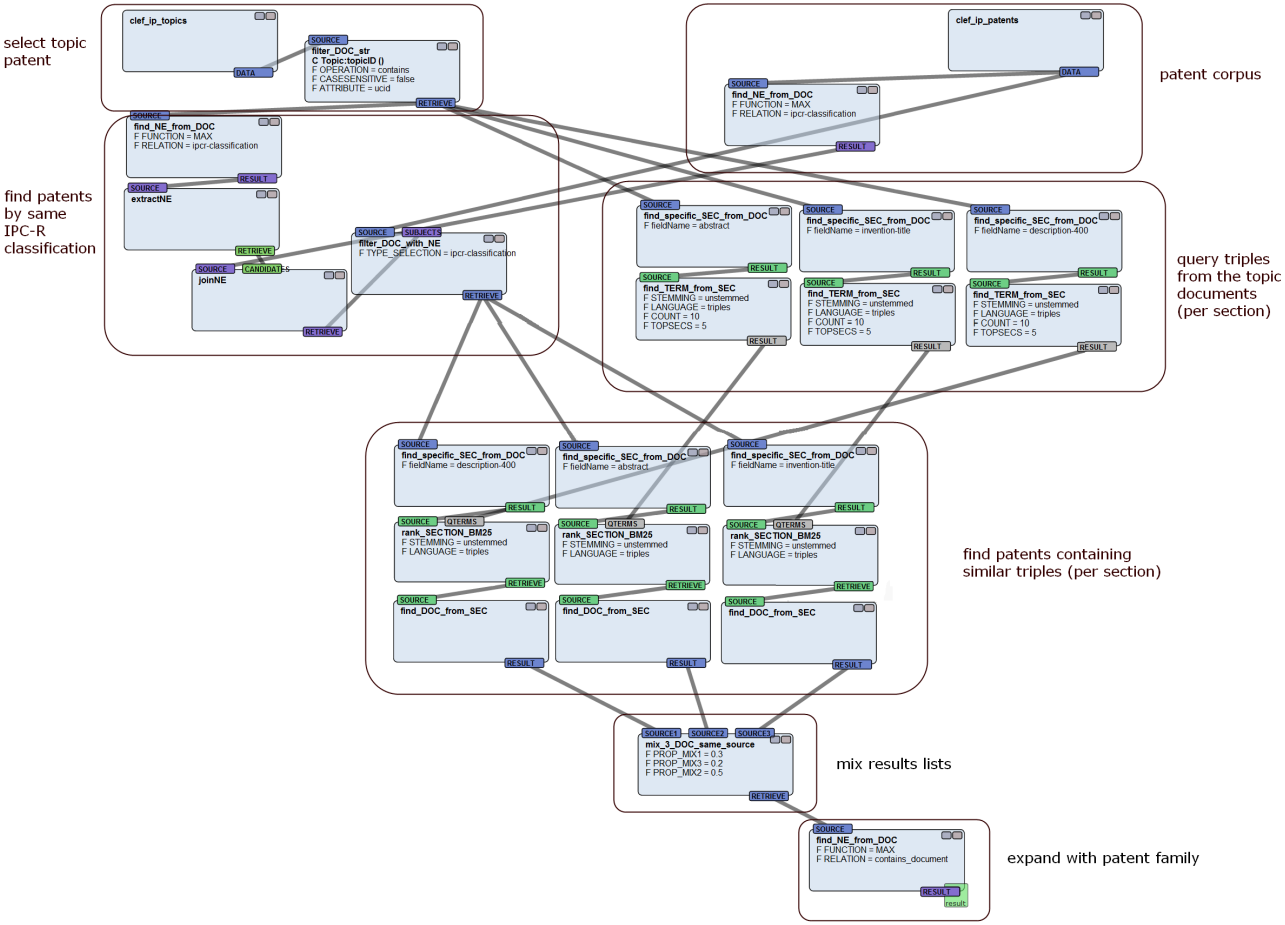

Figure 1: Search strategy for triples_only run

Table 3: Results on general test set

	MAP	set_P	P5	P10	P20	P50	P100	P500	set_ R	R5	R10	R20	R50	R100	R500	nDCG
triples_only	0.0343	**0.0143**	0.0461	0.0353	0.0259	0.0160	0.0103	0.0028	0.2091	0.0358	0.0543	0.0779	0.1187	0.1506	0.2010	0.0981
words_only	**0.0697**	0.0071	**0.0793**	**0.0620**	**0.0457**	**0.0287**	**0.0196**	**0.0070**	0.5827	**0.0626**	**0.0959**	**0.1410**	**0.2167**	**0.2936**	0.5109	**0.2196**
combined_1	0.0665	0.0071	0.0750	0.0585	0.0438	0.0280	0.0193	**0.0070**	0.5877	0.0591	0.0908	0.1344	0.2117	0.2895	0.5144	0.2166
combined_2	0.0524	0.0071	0.0568	0.0450	0.0355	0.0244	0.0179	**0.0070**	**0.5934**	0.0451	0.0708	0.1106	0.1872	0.2715	**0.5156**	0.2018

Table 4: Results on English language-specific test set

	MAP	set_P	P5	P10	P20	P50	P100	P500	set_ R	R5	R10	R20	R50	R100	R500	nDCG
triples_only	0.0386	**0.0136**	0.0510	0.0397	0.0291	0.0184	0.0126	0.0036	0.2601	0.0386	0.0583	0.0830	0.1318	0.1745	0.2475	0.1164
words_only	**0.0737**	0.0062	**0.0884**	**0.0656**	**0.0480**	**0.0302**	**0.0207**	0.0076	0.6011	**0.0654**	**0.0948**	**0.1398**	**0.2142**	**0.2889**	0.5177	**0.2297**
combined_1	0.0713	0.0063	0.0821	0.0621	0.0459	0.0294	0.0203	0.0076	0.6046	0.0620	0.0901	0.1314	0.2070	0.2827	0.5180	0.2269
combined_2	0.0582	0.0064	0.0634	0.0496	0.0387	0.0274	0.0200	**0.0080**	**0.6303**	0.0492	0.0751	0.1129	0.1967	0.2825	**0.5417**	0.2195