Applying Key Phrase Extraction to aid Invalidity Search

Manisha Verma
Language Technologies Research Centre
International Institute of Information Technology
Hyderabad, India
manisha.verma@research.iiit.ac.in

Vasudeva Varma
Language Technologies Research Centre
International Institute of Information Technology
Hyderabad, India
vv@iiit.ac.in

ABSTRACT

Invalidity search poses different challenges when compared to conventional Information Retrieval problems. Presently, the success of invalidity search relies on the queries created from a patent application by the patent examiner. Since a lot of time is spent in constructing relevant queries, automatically creating them from a patent would save the examiner a lot of effort. In this paper, we address the problem of automatically creating queries from an input patent. An optimal query can be formed by extracting important keywords or phrases from a patent by using Key Phrase Extraction (KPE) techniques. Several KPE algorithms have been proposed in the literature but their performance on query construction for patents has not yet been explored. We systematically evaluate and analyze the performance of queries created by using state-of-the-art KPE techniques for invalidity search task. Our experiments show that queries formed by KPE approaches perform better than those formed by selecting phrases based on *tf* or *tf-idf* scores.

Categories and Subject Descriptors

Verticals and specialized search [**Miscellaneous**]:

Keywords

Patent Retrieval, Keyphrase Extraction

1. INTRODUCTION

Patents give exclusive rights to the inventor for using and protecting his intellectual property. For a patent to be granted, the invention has to be novel, non-obvious and useful. Since a lot of patents are present in digital form on the web and with the number of patents filed and granted each year increasing rapidly, patent examiners today use information retrieval tools to accomplish several search tasks.

A patent engineer routinely performs search tasks like prior art search, patentability search, novelty search and invalidity search. The objective of invalidity search is to find patents or other related resources which cover the proposed product or process and are still in force. The search result consists of a report of all such inventions. The search process is time consuming as several patents have to be read to ensure no relevant patent is missed. The examiner starts with a document and manually creates suitable queries to search patent databases. Since a lot of time is spent in constructing relevant queries, transforming the document into a query automatically would save the examiner a lot of effort. Hence, one should be able to input a document as a query instead of making several queries. Query formulation is still a manual process and automating it would require the right combination of query formation, refinement and expansion techniques. An important aspect about invalidity search is the number of relevant documents for a given patent. Since very few patents may infringe an invention, the number of relevant documents is usually small. Hence, it is not only important to construct a query which covers the scope of the invention but also retrieves all the relevant documents.

In this paper we explore state-of-the-art supervised and unsupervised Key Phrase Extraction (KPE) techniques to create queries from an input patent. Supervised KPE approaches need annotated data but publicly available patent data is not annotated and manual annotation would require domain expertise. Thus, we use a corpus based approach to automatically label key phrases in patents with relevance judgments to create training data for supervised KPE algorithms. We use NTCIR 6[1] collection of 1.3 million patents and 1000 query patents to conduct all the experiments.

In Section 2, we discuss the current state of the art in patent retrieval and key phrase extraction. Section 3, discusses motivation and contributions of the approach. Unsupervised and supervised key phrase extraction techniques used for experiments have been explained in Section 4. The approach to annotate patents with phrases is explained in same section. The experiments, result and analysis are explained in Section 5. Conclusion and Future work are discussed in Section 7 and Section 8 respectively.

2. RELATED WORK

Prior-art retrieval and invalidity search have received considerable attention from the research community recently. Several workshops by NTCIR [2] and CLEF [3] have been con-

ICAIL'11 , June 6-10, 2011, Pittsburgh, PA

[1] http://research.nii.ac.jp/ntcir/index-en.html
[2] http://research.nii.ac.jp/ntcir/publication1-en.html
[3] http://www.ir-facility.org/the_irf/clef-ip09-track

ducted to evaluate and improve the state of the art in patent retrieval. Patent retrieval poses a unique challenge as the language of patents is not only vague but also contains a lot of new terms and concepts introduced by the inventor. This results in a lot of content that discusses similar aspects but uses different vocabulary which makes search for similar patents a daunting task. Patents are lengthy but well-structured documents and contain a title, abstract, description, summary of invention and claims. The claims define the scope of protection granted by the patent. All patents have manually-assigned International Patent Classification (IPC) classification codes.

Several approaches have been proposed to improve patent retrieval. Some systems use entire claim text as a query [11] or use information in the patent text to form queries [5, 13, 14] and modify existing retrieval models to improve performance [3, 7, 8].

Other approaches identify strong keywords for query construction from the input patent and then expand these queries using relevance feedback. Bashir et al.[2] analyze the bias of several retrieval systems and query expansion techniques. They propose a query expansion technique based on clustering to identify dominant relevant documents. An extension of the above work is proposed in [1] where several SMART similarity metrics are used to select better terms for query expansion. However, they construct queries of only 2, 3 and 4 words which may not be the case in the real world. And for each patent added to the database these queries will have to be reconstructed to measure the retrievability of the document. Morphological analysis has also been used in [9] to extract words from the claim for a query. These words are used to find related terms from 'detailed description of the invention' and the related terms are used for query expansion.

Xue et al. [18, 19] explore ways to create a query from a patent. They propose a generalized algorithm for extracting query words from a patent. They evaluate queries formed by words extracted from different sections in a patent. They empirically determine how many query words should be kept in the query. Different weighting methods are also used to weigh words in the query.

Retrieval models proposed in the literature will not perform well if the query is constructed with weak key phrases. Thus selection of right phrases becomes an important step. In [10] candidate n-grams are selected using a classifier. The authors manually annotate potential keywords to train the classifier. Extension of their approach to patents from several areas would again require domain expertise. Since this is a time consuming and expensive task, it is infeasible to annotate large volumes of patents in the absence of an expert.

3. MOTIVATION AND CONTRIBUTION

Initial experiments indicated that selecting phrases on basis of frequency counts (*tf* or *tfidf*) resulted in poor queries. Though several key phrase extraction approaches have been proposed in the literature, they have not been used to create queries for invalidity search task. Our contributions are :

1. We systematically evaluate and analyze the performance of queries created by using state-of-the-art unsupervised key phrase extraction techniques.

2. We propose a corpus based mechanism to annotate query keywords in patents for which related patents are known. These patents are used to train supervised key phrase extraction techniques.

4. KEY PHRASE EXTRACTION

The aim of Key phrase extraction (KPE) algorithms is to find out phrases or words that represent important units of information in a document. Key phrases are used in several applications like document categorization, clustering and summarization. A KPE algorithm takes a document as input and outputs phrases or words that represent the document.

A list of phrases, generated by a KPE algorithm, could succinctly represent a complex and lengthy patent. These phrases could then be used to form queries to search for similar patents. Informative phrases will be able to retrieve relevant patents whereas the results for weak phrases will be noisy and irrelevant. Thus, phrases extracted by KPE techniques could be used to search for similar patents.

A KPE technique is supervised or unsupervised. Unsupervised approaches use co-occurrence statistics or frequency counts to extract and score candidate phrases from a document. Unsupervised approaches do not need any labeled data. For certain corpora (e.g. research articles), key phrases annotated by the experts are available. Supervised approaches are trained on this data to extract phrases from new documents. We label some patents with key phrases to create training data for supervised approaches. For all approaches the top phrases are used to form a query to find similar patents and each phrase in the query has the same weight. The performance of queries formed by the supervised approaches is compared with those formed by the unsupervised approaches by using the relevance judgments of query patent. We explore two state-of-the-art supervised and unsupervised approaches respectively to extract phrases from a patent. The approaches are briefly explained in the following subsections.

4.1 Unsupervised KPE Approaches

We use two approaches - TextRank [12] and SingleRank [16]. These algorithms use co-occurrence statistics to score words and identify phrases from a document. These approaches use the information around a word to calculate its importance whereas *tf* or *tf-idf* scores do not reflect this information.

4.1.1 TextRank

TextRank algorithm represents a document as a graph. Each vertex in the graph corresponds to a word. There is an edge between any two words occurring together. A weight, w_{ij}, is assigned to the edge connecting two vertices, v_i and v_j , and its value is the number of times the corresponding word co-occur within a window of W words in the document. The score of a vertex reflects its importance. The score for v_i, $S(v_i)$, is initialized with a default value and is computed in an iterative manner until convergence using this recursive

formula:

$$S(v_i) = (1-d) + d \times \sum_{v_k \epsilon Adj(v_i)} \frac{w_{ji}}{\sum_{v_k \epsilon Adj(v_j)} w_{jk}} S(v_j) \quad (1)$$

where $Adj(v_i)$ denotes v_i's neighbors in the graph and d is the damping factor. Intuitively, a word will receive a high score if it has many high-scored neighbors. After convergence, the $T\%$ top-scored vertices *(words)* are selected to as keywords. Adjacent keywords are then collapsed and output as a key phrase.

4.1.2 SingleRank

SingleRank follows the same approach as TextRank but differs in the following aspect. While in TextRank phrases containing the top-ranked *words* are selected, in SingleRank, we do not filter out any low scoring words. Each candidate key phrase, which can be any longest-matching sequence of nouns and adjectives, is given a score by summing the scores of its constituent words obtained from the graph. Top N highest-scored phrases are output as key phrases.

4.2 Supervised KPE Approaches

For some corpora, key phrases annotated by experts are available. Supervised approaches are trained on this data and the model is used to extract phrases from new documents. We use two approaches - KEA and RankPhrase to form queries for input patents.

4.2.1 KEA

KEA [17] is a popular supervised approach for key phrase extraction. KEA makes a list of candidate phrases by extracting n-grams of a predefined length (e.g. 1 to 3 words) that do not start or end with a stopword. It calculates feature values for each candidate phrase, and uses naive bayes classifier to predict important phrases. A model is trained using documents with known key phrases, and then used to find phrases in new documents. We use the existing implementation of KEA[4] and train it on phrases generated by our annotation approach. Features used to represent a phrase are given below

1. **Tf-Idf** : It assigns a score to each word w in a document d based on its frequency tf_w (term frequency) in d and remaining documents in the corpus *idf* (inverse document frequency). It is given by $tfidf = tf_w \times log(\frac{N}{D_w})$ where N is the number of documents in the corpus and D_w is the number of documents in which w occurs.

2. **First occurrence** : It is computed as the percentage of the document preceding the first occurrence of the term in the document. Terms that tend to appear at the start or at the end of a document are more likely to be key phrases.

3. **Length of a phrase** : It is the number of words in a phrase.

[4]http://www.nzdl.org/Kea/

4.2.2 RankPhrase

Recently it has been proposed that key phrase extraction is a problem of ranking and not that of classification [6]. Instead of using a classifier, they use a Learning-To-Rank approach to train a ranking model on phrases and the model ranks phrases from a new document. A ranked list of phrases is used for training a model. This model then predicts the order for phrases of a new patent. We use the features proposed in KEA to represent phrases in the document.

4.2.3 Annotating Data

To use supervised KPE approaches, patents will have to be manually annotated for key phrases. This is a difficult task since (1) manual annotation would require domain expertise, given the number of fields in which patents are written, it is infeasible to annotate patents for each domain and (2) it is laborious, expensive, time consuming and prone to inter-expert labeling variability. Thus, we use a corpus based approach to annotate important key phrases in a patent.

NTCIR 6 dataset provides a list of relevant documents (relevance judgments) for some query patents. We use these query patents to create training data for supervised approaches. We consider those phrases of the query patent to be important, which when treated as queries, can retrieve related documents from the corpus.

To identify candidate phrases, we use the relevance judgments of the query patent and a search engine. The process of labeling phrases is as follows: A chunker is used to extract all the noun phrases from a patent and each phrase is stemmed using a stemmer. Stop words are removed from the phrases using a pre defined list. Each phrase in the resultant list is fired as a query in the search engine. If the phrase is able to retrieve documents that are relevant to the query patent, it is informative i.e. it captures some important information about the query patent, hence it can be a candidate key phrase for that patent. Thus, each phrase is treated as a query and its Mean Average Precision (MAP) and Recall are calculated using the relevance judgments of the input patent. The phrases with MAP and Recall greater than zero are considered to be informative. To remove noisy chunks (symbols, abbreviations etc) from the list, we consider chunks of length greater than θ *letters*. We select phrases with high MAP and Recall values and then rank them based on *tf-idf* scores. Top phrases are used to represent the query patent. The algorithm is summarized in Algorithm 1.

The process described above may select phrases which may not be informative but still retrieve documents relevant to the query patent. Our objective is not to identify only the important information but also to identify those words which will help in retrieving relevant documents. For example, an expert will select phrases specific to the invention, these phrases may or may not form good queries. While creating queries manually, one might miss terms which may not be specific to the invention but will help in retrieving similar patents. The terms that may not be specific to the invention, will get selected based on MAP and Recall values. Ranking these terms based on *tf-idf* scores will ensure that they are also informative.

Algorithm 1 Algorithm to annotate patents for key phrases

Input: Chunked noun phrase list: CL, Key phrases list: KPL [ ], Noun Phrase : NP
  for all NP_i in CL **do**
    Remove stop words
    if $len(NP_i) \geq \theta$ **then**
      add NP_i to $tempList$
    end if
  end for
  for all NP_i in $tempList$ **do**
    Search NP_i in the corpus D
    Retrieve relevant documents
    if $MAP(NP_i) > 0 \wedge Recall(NP_i) > 0$ **then**
      add NP_i to KPL
    end if
  end for
  Sort KPL in descending order of $MAP(NP_i)$
  Sort KPL in descending order of *tf-idf*
  Final Key phrases ← Top 30 phrases in KPL

Table 1: Distribution of 1000 Queries

Domain	# pat	Domain	# pat
Transport	31	Electronics	262
Chemistry	45	Engine/Pumps	15
Textile	2	Separating/Mixing	6
Instruments	543	Agriculture	75
Mining	20		

The apparent advantage of our approach is its simplicity. With the help of a search engine and available relevance judgments for patents, one can create a list of suitable key phrases in no time. These phrases can be used to train models for supervised KPE algorithms, to suggest keywords to a user who is searching for similar patents etc. Another advantage is the ability of the approach to cover patents in several domains. One might argue that a patent may not contain any phrase with non-zero MAP and Recall. But during the experiments it was observed that since the patents are such lengthy documents, they contain a lot of vocabulary and each patent would undoubtedly yield some phrases which have a MAP or Recall value greater than zero.

5. EVALUATION

5.1 Corpus

We test the performance of each algorithm on the NTCIR 6 dataset of 13 million USPTO patents from 1993 to 2000. We take 1000 sample patent applications as queries. The list of relevant documents for each of these applications is provided with the dataset. The average number of phrases per patent is 1001. The patents contain four main sections - title, abstract, claim and description which is further divided into summary and brief description. The division of query patents according to their IPC codes is given in Table 1.

5.2 Experimental Setup

Lucene[5] has been used to index the data and Snowball Analyzer[6] with a manually determined list of 146 stop words has been used to analyze the corpus and queries. The data was indexed with four fields - title, abstract, description and claim. Opennlp[7] has been used to POS tag and chunk patents. Vector space model has been used to retrieve documents. For every query patent, key phrases have been extracted using the approaches described in Section 4. To determine the number of phrases in a query, we formulate queries with 10 to 60 phrases. Phrases were selected from whole patent (*all*) and description (*desc*) based on *tf* and *tf-idf*. R@100 for queries are shown Figure 1. With more words, the change in the performance is not significant, but the time spent on search is significantly increased. Since there is a very minute difference between Recall of query with 40 and 60 words, we limit the number of phrases in the query to 40 phrases. The queries are formed by selecting phrases based on *tf-idf* from description section of the patent. Top 40 phrases are used to form a boolean 'OR' query to search for similar patents. Note that all the phrases in the query have the same weight.

We use publicly available implementation of TextRank and SingleRank [4]. The damping factor is set to 0.85. In Tex-

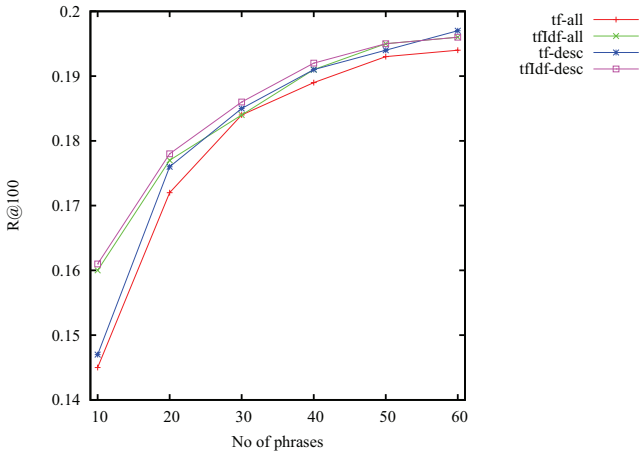

Figure 1: Effect of number of phrases on Recall

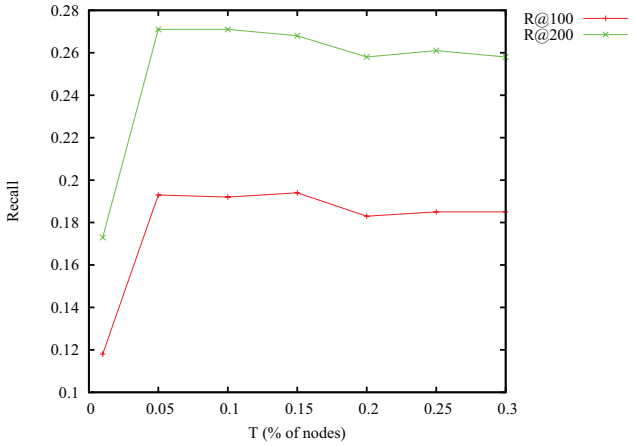

Figure 2: Effect of T% (TextRank) on Recall

[5]http://lucene.apache.org/java/docs/index.html
[6]http://snowball.tartarus.org/
[7]http://opennlp.sourceforge.net/

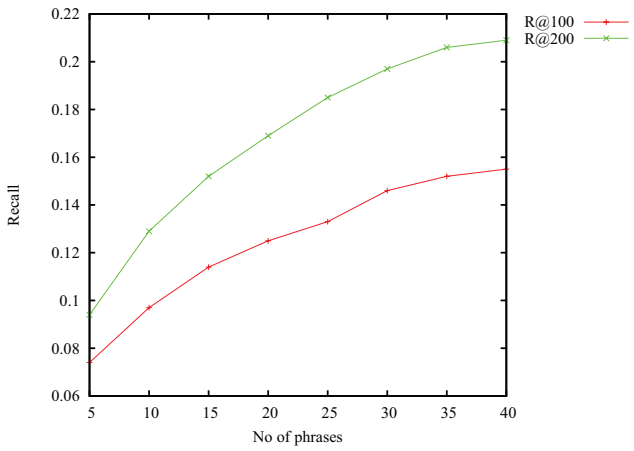

Figure 3: Effect of N (SingleRank) on Recall

tRank, the number of vertices used to create phrases (T%) was varied from 0.05% to 30%. Experiments show that better phrases are generated when top 0.15% nodes from the graph are collapsed to form phrases. In SingleRank, we vary N from 5 to 40 phrases. It was found that N=30 resulted in better phrases. R@100 and R@200 of queries formed by varying N in SingleRank and T% in TextRank are shown in Figure 3 and Figure 2 respectively. We tested both approaches on 1000 query patents and use window size of 2, 3, 4 and 5 words to create the co-occurrence graph. TextRank and SingleRank have been tested on individual sections (abstract, description and claims) in the patent also. Since individual sections contain less vocabulary, small values of T result in very few phrases. To prevent this, the value of T was increased to 50%, 30% and 30% for abstract, description and claims respectively.

Query patents, with relevance judgments, are used to create training data for supervised approaches. The value of θ is set to 4 for testing the algorithm. The supervised KPE approaches have been trained and tested on these queries. We use publicly available implementation of KEA. SVMRank[8] has been used to train and test the model to rank phrases of a query patent. For supervised approaches we use 5-fold cross validation. KEA and RankPhrase have been trained on 5 subsets of queries; each has 400 patents for training and 200 for testing. Since invalidity search is a recall oriented task and a patent examiner usually scans top 200 documents [15] we report *Recall* values at 10, 30, 100 and 200.

Performance of queries formed by selecting phrases from the description section (from [18]) on the basis of - frequency of a phrase in the patent *(tf)*, *tf-idf* and *idf* is the baseline for our work. We perform paired t-test to calculate the statistical significance.

6. RESULTS AND DISCUSSION

The results for queries formed by phrases based on *tf*, *idf* and *tf-idf* are shown in Table 2. Unsupervised approaches have been tested with window size of 2, 3, 4 and 5 words to create the co-occurrence graph. The performance of queries, formed by selecting phrases from individual sections and entire patent Table 3 and Table 2 respectively. Results of supervised approaches trained on the annotated data are given in Table 4.

Queries based on Inverse document frequency (*idf*) contain rare phrases, i.e. phrases which occur in less number of documents. There are two reasons why they do not perform well. First is, though two patents may claim the same invention they use different terminology to describe it, hence the vocabulary overlap between them is minimal. Due to this reason several phrases in a patent have high *idf* values. But extremely low occurrence of a phrase in the corpus does not indicate that it is important and queries with such phrases may not necessarily be informative. Second reason is that *idf* values do not reflect the frequency with which a phrase has been used in the input patent. For example, a phrase may be present in a small fraction of documents *(high idf)* in the corpus but occurs only once *(low tf)* in the document. This phrase, if used to search for similar patents may not retrieve relevant documents.

Queries formed by selecting phrases based on their frequency in the document *(tf)* and *tf-idf* perform better. This is due to the presence of document level information about the phrase in its *tf* and *tf-idf* score. The author of a patent uses some phrases repeatedly to describe a part or component of the invention. Such important phrases have higher frequency than others. Selecting phrases based on *tf* ensures that these informative phrases are present in the queries. Queries based on *tf-idf* contain phrases which have both high *tf* and *idf*. As a result, *tf-idf* queries perform better than both - *tf* and *idf* queries.

Both unsupervised approaches have been tested on 1000 query patents with window size of 2, 3, 4 and 5 words to create the co-occurrence graph. Unsupervised approaches yield slightly better queries than *tf-idf*. This is due to the co-occurrence information used by both TextRank and SingleRank. The performance of unsupervised approaches degrades as the window size increases which is intuitive, since, increasing the size weakens semantic relation between farthest words in the phrase. The is considerable difference in the performance of TextRank and SingleRank. MAP and R@100 of TextRank are more than that of SingleRank. This can be explained by the following: TextRank requires that every word of a key phrase must appear among the top ranked unigrams. SingleRank, on the other hand does not require all unigrams of a key phrase to be present in the top-ranked list of words. TextRank has a fairly strict criterion, in comparison to SingleRank, which helps in lowering the importance of those phrases which do not contain any top ranked word from the graph, this in turn helps in reducing the noise and better key phrases are retrieved using TextRank. Recall values of TextRank are also higher than SingleRank and *tf-idf*.

The performance of TextRank and SingleRank is dependent on the graph constructed from the patent text. Both the approaches do not perform well when individual sections of a patent are used to extract key phrases. This can be explained as follows: (1) Individual sections represent the document partially which provides incomplete estimate of co-

[8]http://svmlight.joachims.org/

Table 2: Performance of queries formed by unsupervised approaches

		MAP	R@10	R@30	R@100	R@200
tf		0.0414	0.0365	0.0860	0.1740	0.2390
idf		0.0140	0.0201	0.0325	0.0510	0.0640
tf-idf		**0.0428**	**0.0365**	**0.0870**	**0.1781**	**0.2412**
TextRank	2	**0.0458**	**0.0456**	**0.0969**	**0.1885**	**0.2606**
	3	0.0455	0.0449	0.0969	0.1859	0.2576
	4	0.0452	0.0444	0.0958	0.1848	0.2561
	5	0.0454	0.0440	0.0960	0.1853	0.2560
SingleRank	2	**0.0340**	**0.0316**	**0.0689**	**0.1380**	**0.2010**
	3	0.0336	0.0314	0.0687	0.1360	0.1925
	4	0.0333	0.0310	0.0675	0.1362	0.1930
	5	0.0332	0.0309	0.0671	0.1357	0.1910

Table 3: Performance of unsupervised approaches on patent fields (TR: TextRank, SR: SingleRank)

	MAP	R@10	R@30	R@100	R@200
abst TR	0.0285	0.0245	0.0560	0.1230	0.1775
desc TR	**0.0400**	**0.0312**	**0.0737**	**0.1555**	**0.2234**
claim TR	0.0321	0.0274	0.0641	0.1390	0.2000
abst SR	0.0305	0.0269	0.0629	0.1325	0.1890
desc SR	**0.0335**	**0.0321**	**0.0682**	**0.1356**	**0.1915**
claim SR	0.0328	0.0315	0.0648	0.1335	0.1900

occurrence statistics and (2) Since entire document text provides a better estimate of edge weights in the graph, it results in better ranking of vertices (*words*). In individual sections, the longer sections will perform better than shorter sections. This is reflected in the experiment results too. The entire patent text yields better phrases than individual sections. In individual sections, TextRank and SingleRank perform better when *description* is used to form co-occurrence graph. 'Description' section in a patent contains more vocabulary than other sections, this results in better edge weights in the co-occurrence graph, which in turn results in better phrases. In supervised approaches, queries created by using phrases extracted by KEA show 29% and 37% improvement in MAP and 27% and 29% improvement in R@100 over TextRank and *tf-idf* respectively. There is a substantial improvement over all the other KPE approaches as well. This indicates that queries formed by combining phrases output by KEA, were better than those created from other approaches. Since the queries performed well, better phrases were selected by the KEA algorithm. Since the phrases were informative, it can be deduced that KEA was provided reasonable training data. Though RankPhrase approach performs better than unsupervised approaches, it does not match the performance of KEA in extracting phrases. This can be attributed to the length of patent documents. Since the patent documents are lengthy, the number of phrases used is more, hence some important phrases will not be ranked correctly by the approach which lowers the performance of the queries as some important phrases are missed. Our experiments indicate that queries made by using phrases from KPE techniques certainly improve invalidity search, only that improvement is more when supervised approaches are used for extraction. To find out the performance of key phrase extraction techniques in creating queries, it was important that experiments are conducted with both supervised and unsupervised KPE approaches. The experiments show that queries formed by using KPE approaches can indeed improve patent retrieval.

Table 4: Performance of queries formed by supervised approaches

	MAP	R@10	R@30	R@100	R@200
KEA	**0.059**	**0.054**	**0.121**	**0.230**	**0.315**
RankPhrase	0.052	0.053	0.108	0.200	0.284

7. CONCLUSION

Automatic construction of queries from patents would be useful in applications like invalidity search. The current approach is to create a query from the patent by selecting top K keywords based on some score. In this work we tried to find out the performance of queries made by using phrases extracted by popular key phrase extraction techniques. We used both supervised and unsupervised key phrase extraction algorithms to extract phrases from a patent application and form queries to search for similar patents. The performance of these queries is compared with those formed by selecting phrases based on *tf*, *idf* and *tf-idf*. The results indicate that *tf-idf* is not a good metric to select key phrases to form queries from input patents. Queries created by using unsupervised and supervised approaches perform better than those formed by *tf* or *tf-idf*. To train supervised KPE approaches labeled data is required. Since there is no annotated data for candidate keywords in patents, we propose an approach to annotate important key phrases in patents. Supervised approaches are trained on this data. The experiments indicate that key phrase extraction techniques indeed improve invalidity search results. In supervised approaches, queries created by using phrases extracted by KEA show 29% and 37% improvement over TextRank and *tf-idf* respectively. Since queries generated by supervised approaches perform better than those generated by unsupervised approaches, it can be inferred that our annotation approach is able to label informative phrases in a patent.

8. FUTURE WORK

For future work, the queries generated by these approaches could be expanded or weighed to improve retrieval. We shall evaluate the performance our annotation approach and KPE techniques on multilingual patent datasets. In future, we will explore how structure of a patent, frequency count and co-occurrence information can be incorporated in one key

phrase extraction algorithm to improve performance. Combination of unsupervised and supervised approaches to create queries from patents will also be explored in future.

9. REFERENCES

[1] S. Bashir and A. Rauber. Analyzing document retrievability in patent retrieval settings. In S. Bhowmick, J. KÃijng, and R. Wagner, editors, *Database and Expert Systems Applications*, volume 5690 of *Lecture Notes in Computer Science*, pages 753–760. Springer Berlin / Heidelberg, 2009.

[2] S. Bashir and A. Rauber. Improving retrievability of patents with cluster-based pseudo-relevance feedback documents selection. In *CIKM '09: Proceeding of the 18th ACM conference on Information and knowledge management*, pages 1863–1866, New York, NY, USA, 2009. ACM.

[3] A. Fujii. Enhancing patent retrieval by citation analysis. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 793–794, New York, NY, USA, 2007. ACM.

[4] K. S. Hasan and V. Ng. Conundrums in unsupervised keyphrase extraction: Making sense of the state-of-the-art. In *Proceedings of COLING 2010: Posters Volume*, pages 365–373, 2010.

[5] K. V. Indukuri, A. A. Ambekar, and A. Sureka. Similarity analysis of patent claims using natural language processing techniques. In *ICCIMA '07: Proceedings of the International Conference on Computational Intelligence and Multimedia Applications (ICCIMA 2007)*, pages 169–175, Washington, DC, USA, 2007. IEEE Computer Society.

[6] X. Jiang, Y. Hu, and H. Li. A ranking approach to keyphrase extraction. In *SIGIR '09: Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 756–757, New York, NY, USA, 2009. ACM.

[7] I.-S. Kang, S.-H. Na, J. Kim, and J.-H. Lee. Cluster-based patent retrieval. *Inf. Process. Manage.*, 43(5):1173–1182, 2007.

[8] J. Kim, I.-S. Kang, and J.-H. Lee. Cluster-based patent retrieval using international patent classification system. In Y. Matsumoto, R. Sproat, K.-F. Wong, and M. Zhang, editors, *Computer Processing of Oriental Languages. Beyond the Orient: The Research Challenges Ahead*, volume 4285 of *Lecture Notes in Computer Science*, pages 205–212. Springer Berlin / Heidelberg, 2006.

[9] K. Konishi, A. Kitauchi, and T. Takaki. Invalidity patent search system of ntt data. In *Working Notes of the Fourth NTCIR Workshop Meeting, NII*, 2005.

[10] P. Lopez and L. Romary. Experiments with citation mining and key-term extraction for Prior Art Search. In *CLEF 2010 - Conference on Multilingual and Multimodal Information Access Evaluation*, Padua Italie, 2010.

[11] H. Mase, T. Matsubayashi, Y. Ogawa, M. Iwayama, and T. Oshio. Proposal of two-stage patent retrieval method considering the claim structure. *ACM Transactions on Asian Language Information Processing (TALIP)*, 4(2):190–206, 2005.

[12] R. Mihalcea and P. Tarau. TextRank: Bringing order into texts. In *Proc. of EMNLP*, 2004.

[13] T. Takaki, A. Fujii, and T. Ishikawa. Associative document retrieval by query subtopic analysis and its application to invalidity patent search. In *CIKM '04: Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pages 399–405, New York, NY, USA, 2004. ACM.

[14] S. Tiwana and E. Horowitz. Extracting problem solved concepts from patent documents. In *PaIR '09: Proceeding of the 2nd international workshop on Patent information retrieval*, pages 43–48, New York, NY, USA, 2009. ACM.

[15] Y. H. Tseng and Y. J. Wu. A study of search tactics for patentability search: a case study on patent engineers. In *PaIR '08: Proceeding of the 1st ACM workshop on Patent information retrieval*, pages 33–36, New York, NY, USA, 2008. ACM.

[16] X. Wan, J. Yang, and J. Xiao. Towards an iterative reinforcement approach for simultaneous document summarization and keyword extraction. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 552–559, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[17] I. H. Witten, G. W. Paynter, E. Frank, C. Gutwin, and C. G. Nevill-Manning. Kea: practical automatic keyphrase extraction. In *DL '99: Proceedings of the fourth ACM conference on Digital libraries*, pages 254–255, New York, NY, USA, 1999. ACM.

[18] X. Xue and W. B. Croft. Automatic query generation for patent search. In *CIKM '09: Proceeding of the 18th ACM conference on Information and knowledge management*, pages 2037–2040, New York, NY, USA, 2009. ACM.

[19] X. Xue and W. B. Croft. Transforming patents into prior-art queries. In *SIGIR '09: Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 808–809, New York, NY, USA, 2009. ACM.