Improving Machine Learning Approaches for Keyphrases Extraction from Scientific Documents with Natural Language Knowledge

Mikalai Krapivin, Aliaksandr Autayeu, Maurizio Marchese,
Enrico Blanzieri, and Nicola Segata

DISI, University of Trento, Italy

Abstract. We study the use of Natural Language Processing techniques to improve different machine learning approaches (Support Vector Machines (SVM), Local SVM, Random Forests), to tackle the problem of automatic keyphrases extraction from scientific papers. For the assessment we propose a large high quality dataset: 2000 ACM papers from the Computer Science domain. Evaluation shows promising results that outperform state-of-the-art Bayesian learning system KEA improving the average F-Measure from **22,02**% (KEA) to **29,78**% (Random Forest) on the same dataset without the use of controlled vocabularies. The assessment is performed by comparison with expert assigned keyphrases. Finally, we report a detailed analysis of the effect of the individual NLP features and data sets size on the overall quality of extracted keyphrases.

1 Introduction

Exponential growth of information in web era made autonomous digital libraries very popular. Autonomy means automatic information harvesting, processing, classification and representation and it brings several challenges.

A specific challenge lies in the domain of scholarly papers [1], accumulated in autonomous digital libraries [2] like CiteSeerX[1], Google Scholar[2] or Rexa[3]. Library spider crawls the web for scientific papers. Having retrieving the paper, the crawler must convert it into a text format. Then it extracts relevant metadata (like title, authors, citations) and finally documents are properly analyzed (classified, identified, ranked, stored). Metadata helps to categorize or classify papers, simplifying and enhancing users' searches, but often metadata is not available explicitly.

Information extraction from scholarly papers contains two broad classes of tasks:

- recognition of structural information which is present inside the paper body (like authors, venues, title, abstract, text parts like sections, tables figures, author assigned keyphrases[4]);

[1] `http://citeseer.ittc.ku.edu`
[2] `http://scholar.google.com`
[3] `http://rexa.info`
[4] the ones that follows after word "Keywords:" from a new line

– extraction of information which is only implicitly present, such as generic keyphrases or tags, which are not explicitly assigned by the authors.

In this paper we focus on the second, more challenging task of extraction of implicit information. We analyze the effect of the use of Natural Language Processing (NLP) techniques and the use of specific NLP-based heuristics on the improvement of current Machine Learning (ML) approaches.

Machine learning methods are successfully used to support automatic information extraction tasks for short news, mails, web pages [3], and also for the problem of keyphrases extraction [4–6]. Keyphrase is a short phrase representing a concept from a document. They are useful for search, navigation and classification of digital content.

This paper extends and improves on a preliminary work describing our initial concepts and presenting initial results [7] obtained using standard SVM approaches. In this paper:

1. We extend the use of state-of-the-art NLP tools [8] for the extraction, definition and use of linguistic-based features such as part of speech and syntactic relations extracted by dependency parsers [9].
2. We apply the proposed NLP-based approach to a number of different ML methods namely traditional SVM, innovative Local SVM and Random Forests.
3. We define and publish a large high quality dataset of 2000 documents [10], available through internet, with experts assigned keyphrases in Computer Science field.
4. We analyze in details the effect of different NLP features and dataset size on the overall quality of extracted keyphrases.
5. We perform a comparative analysis of the computed quality measures and obtained keyphrases with the various ML techniques (enhanced with NLP) and the popular Bayesian learning system KEA.

We organize the paper as follows: In Section 2 we present a brief review of the state-of-the-art in the domain and a discussion of relevant related work. Section 3 provides a detailed description of the dataset used in our experiments. Section 4 presents the details of the proposed extraction methodology (that we name hereafter ML+NLP), specific feature set, text processing tasks and result assessment methodology. In Section 5 we present the results of both approaches, Bayesian Learning (KEA) and Support Vector Machine training. In Section 6 we present a quantitative comparative analysis of the obtained quality measures as well as a qualitative analysis of the extracted keyphrases. Section 7 is devoted to the conclusions and discussion of future work.

2 Related work

The state-of-the-art system for keyphrases extraction is KEA [11]. It uses Naive Bayes classificator and few heuristics. The best results reported by KEA team show about 18% of Precision [11] in the extraction of keyphrases from generic

web pages. Usage of domain specific vocabularies may improve the result up to 28.3% for Recall and 26.1% for Precision [12].

Another approach is suggested by Tourney and uses GenEx algorithm [13]. GenEx is based on a combination of parameterized heuristic rules and genetic algorithms. The approach provides nearly the same precision and recall as KEA. In a more recent work [3], the author applies web-querying techniques to get additional information from the Web as background knowledge to improve the results. This method has a disadvantage: mining the Web for information and parsing the responses is a time and resource consuming operation. This is inconvenient for Digital Libraries with millions of documents. In this approach the author measures the results by the average number of correctly found phrases vs. total number of extracted phrases ratio.

Recent works by A. Hulth et al. took into account domain [6] and linguistic [5] knowledge to search relevant keyphrases. In particular, [6] used thesaurus trying to get domain knowledge. Recall reported in this work is very low, namely 4-6%. The approach proposed in [5] introduced a heuristic related to part-of-speech usage, and proposed training based on the three standard KEA features plus one linguistic feature. Authors reported relatively good results (F-Measure up to 33.9%). However, it is hard to compare their results with others due to the strong specificity of the used data set: short abstract with on average 120 tokens where around 10% of all words in the proposed set were keyphrases.

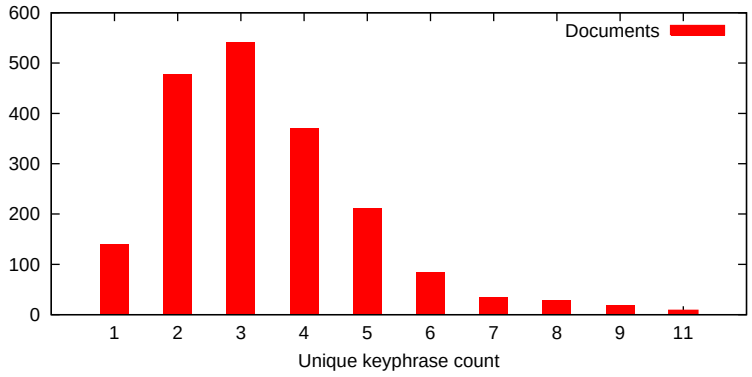
A recent interesting work with regard to the application of linguistic knowledge to the specific problem is reported in [14]. The authors used WordNet and "lexical chains" structures based on synonyms and antonyms. Then they applied decision trees as a ML part on about 50 journal articles as the training set and 25 documents as the testing set. They reported high precision, up to 45%, but did not mention recall. This makes difficult any comparison with other techniques, and, as it will be investigated below, we think that such dataset is too small and biased for comparison.

Other ML technique, least square SVM [4] shows 21.0% Precision and 23.7% Recall in the analysis of web mined scientific papers. Also in this case the described experiments are limited to a very small testing dataset of 40 manually collected papers, which is again very small set.

Our work contributes in three dimensions:

- using a large high quality dataset; this dataset is open and can establish a ground for further competition.
- empowering machine learning methods with NLP techniques and heuristics.
- applying ensemble learning methods that have not been applied to the problem before; and providing the deep investigation of the results (feature importance, dataset size importance, etc).

Fig. 1. Distributions of unique assigned keyphrases per document.



3 Dataset Description, Characterization and Linguistic Processing

3.1 Dataset Description and Characterization

The dataset presented [10] contains a set of papers published by the ACM in the Computer Science domain in 2003–2005. The documents are included in the ACM portal[5] and their full texts were crawled by CiteSeerX digital library. In our pre-processing tasks, we separated different parts of papers, such as title and abstract, thus enabling extraction based on a part of an article text. Formulas, tables, figures and eventual LaTeX mark up were removed automatically. We share this dataset and welcome interested communities to use it as a benchmarking set for information extraction approaches.

For our investigations we separate existing keyphrases into two categories:

- author assigned: located inside each document in the header sections after the prefix "Keywords:";
- editor assigned: manually assigned by human experts in a particular domain.

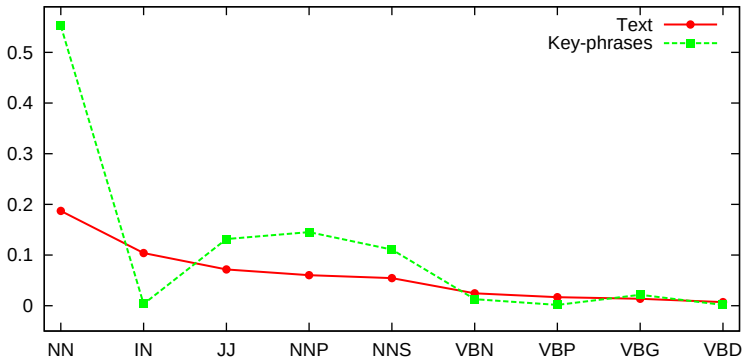
Our experimental dataset consists of 2000 documents with keyphrases assigned by ACM editors. It is important to note that in our preparation of the above dataset, we have selected only papers that contain at least one expert assigned keyphrase in the full text of a document. So we are not in the more challenging case of *completely* implicit extraction. In our dataset, each document has on average about 3 unique human assigned keyphrases (see Figure 1).

3.2 Linguistic Analysis of Keyphrases

We performed a NLP analysis of the keyphrases to study their linguistic properties. This is the base for the proposal of heuristics and the definition of the

[5] `http://portal.acm.org`; available also at `http://dit.unitn.it/~krapivin/`

Fig. 2. POS tags distributions for normal text and keyphrases.



features used in the machine learning step. This improves the quality of generated keyphrase candidates while simultaneously reducing their quantity.

We analyzed a sample of 100 random documents using OpenNLP tools. We applied tokenizer, Part of Speech (POS) tagger and chunker to explore differences between POS tags and chunk types for normal text documents and the corresponding keyphrases set. Fig. 2 shows POS tag distributions for the most common POS tags, such as nouns: NN, NNP, NNS; prepositions: IN, adjectives: JJ and verbs: VBN, VBP, VBG, VBD. Fig. 3 shows distributions for chunk types, such as noun phrases: B-NP, I-NP; prepositional phrases: B-PP; verbal phrases: B-VP, I-VP. To improve readability we have omitted values close to zero.

One can note from the figures that the distributions differ significantly between normal text and keyphrases sets. The major differences in POS tags distribution confirm that the majority of keyphrases consist of nouns, singular as well as plural, and adjectives. The difference in chunk types distribution also confirms and reinforces this hypothesis, adding to it that the overwhelming majority of keyphrases are noun phrases.

We did another analysis using MaltParser to explore differences between dependencies of keyphrases and the ones of normal text. Fig. 4 compares keyphrases and normal text dependency distributions. We use the results of this analysis to compose features set.

3.3 Text processing

Before any extraction task, text needs to be pre-processed to assure a reasonable quality of extraction [15]. Modern scientific papers are mostly available in PDF format, thus first we need to convert them to plain text. Further preprocessing includes sentence boundary detection, tokenization, POS tagging, chunking, parsing, stemming and recognizing separate blocks inside the article, such as Title, Abstract, Section Headers, Reference Section, Body.

Fig. 3. Chunk types distributions for normal text and keyphrases.

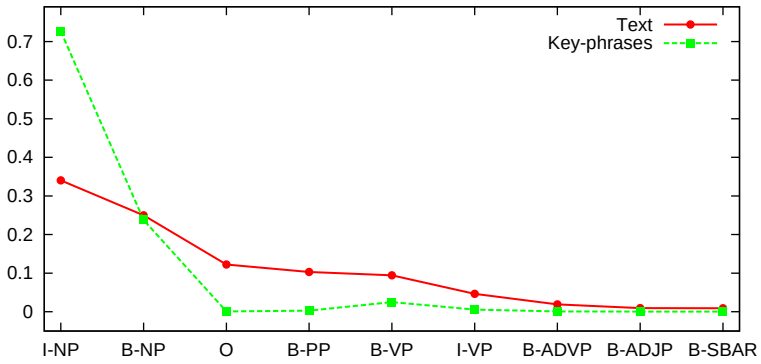
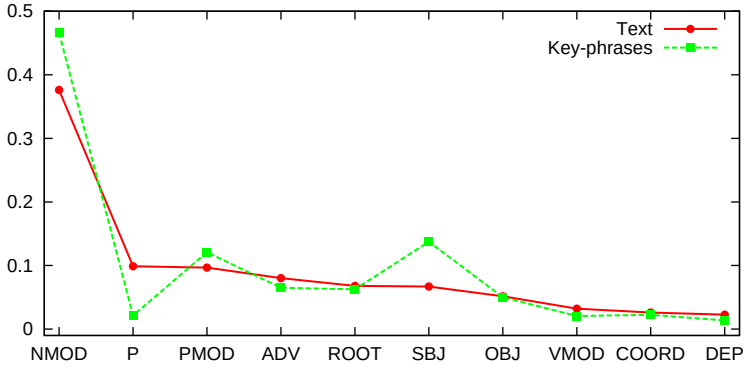


Fig. 4. Dependencies distribution for normal text and keyphrases.



We used OpenNLP suite [8] to do standard steps of text processing. Namely, we apply sentence boundary detector, tokenizer, part of speech tagger and chunker consequently. Then we apply a heuristic, inspired by the previous linguistic analysis of keyphrases.

The heuristic consists of two steps. First we filter by chunk type, leaving only NP chunks for further processing. Then we filter the remaining chunks by POS. We leave only chunks with tokens belonging to the parts of speech from the top of the distribution in Fig. 2, such as NN, NNP, JJ, NNS, VBG and VBN. Table 1 shows an example sentence and extracted keyphrase candidates. This heuristic extracts for further analysis only linguistically meaningful keyphrase candidates.

We use S-removal stemmer, embedded into KEA, to avoid problems with the same words written in different forms. Table 2 shows the examples of original and stemmed forms.

In addition, we apply MaltParser to extract dependencies which we use as additional features for machine learning.

Table 1. Keyphrase candidates extracted by the heuristic.

Sentence	Therefore, the seat reservation problem is an on-line problem, and a competitive analysis is appropriate.
Candidates	seat, seat reservation, seat reservation problem, reservation, reservation problem, problem, on-line problem, problem, competitive analysis, analysis

Table 2. Original and stemmed forms

Original	Stemmed
Multiple selections	multipl selection
text editing	text editing
SPFD-based global rewiring	spfd base global rewiring
Glauber dynamics	glauber dynamic
networking	networking
network	network

4 Enhancing Machine Learning with Natural Language Processing

4.1 Features selection

Proper feature space selection is a crucial step in information extraction. Many features may be used for accurate information extraction and their characteristics are strongly domain dependent.

The feature set we propose is detailed in Table 3. Features 1, 2 and 3 are common and widely used in most information extraction systems [16]. Less traditional features are: feature 4 quantity of tokens in a phrase, used in [1], feature 5 – the part of a text, successfully used in [4]. The features numbered in Table 3 from 6 to 20 are based on linguistic knowledge. We consider keyphrase containing a maximum of 3 tokens, with indices 1, 2 and 3 in the feature names referring to the first, second and third token of the candidate, respectively. Features 6 to 8 contain part of speech tags of the tokens of a keyphrase candidate.

The next set of features uses dependencies given by the MaltParser. Each dependency contains a head, a dependent and a labeled arc joining them. Dependencies help us to capture the relations bonding keyphrase tokens together. They also help to grasp the position and the role of the keyphrase in the sentence. Features 9-11 contain part of speech tag of a head of the token for each token of the candidate. Features 12-20 refer to the relations within the keyphrase and relations attaching a keyphrase to the sentence. They consist of three groups, one group for each token of the keyphrase candidate, and have similar meaning. Let us consider in detail the first group, features 12-14. Feature 12 refers to the label of the arc from the first token of the candidate to its head. It grasps the relation between the keyphrase and the sentence or between the tokens of keyphrase. Features 13 and 14 grasp the cohesion of keyphrase and its relative position in

Table 3. The adopted Feature Set, $i \in [1..3]$

# Feature	# Feature
1 term frequency	6-8 i-th token POS tag
2 inverse document frequency	9-11 i-th token head POS tag
3 position in text	12,15,18 i-th token dependency label
4 quantity of tokens	13,16,19 distance for i-th incoming arc
5 part of text	14,17,20 distance for i-th outgoing arc

the sentence. Feature 13 refers to the distance between the first keyphrase token and its dependent if it exists. As a distance we take the difference between token indexes. Feature 14 refers to the distance between the first token and its head.

4.2 Machine Learning Methods used for comparison

Random Forest. Nowadays ensemble learning is getting more popular, one may be divided into two main branches: (i) bagging [17] and (ii) boosting [18]. The core idea of ensemble learning is in the construction of many decision trees (or other) classifiers and voting for the best result. Bagging and boosting main difference in a way of constructing decision trees. Random Forest is an extension of earlier "bagging" technique proposed by Leo Breiman in 2001 [19, 20]. The RF algorithm randomly takes piece of a training set to grow each tree, and does not need costly cross-validation procedure. Being scalable and relatively fast RF improves state-of-the-art in a different machine learning tasks.

Support Vector Machines (SVMs) [21] are classifiers with sound foundations in statistical learning theory [22] which are now considered the state-of-the-art classification method for a wide range of computational tasks. The reasons of their success are related to their ability to find the optimal solution, the possibility of highly non-linear mappings of the input space, the handling of noisy data with a soft-margin approach and their robustness to the curse of dimensionality. Differently from many text classification approaches based on the "bag of words" representation (where each text is encoded in a binary vector denoting which words are present) that causes a very high dimensionality of the data, we are working here with only 20 features. When the dimensionality is high a linear classifier is frequently the best choice, while with a reduced number of features a non-linear approach is needed. For this reason we adopt SVM with the Gaussian radial basis function (RBF) kernel.

FaLK-SVM [23] is a kernel method based on Local SVM [24] which is scalable for large datasets. There are theoretical and empirical arguments supporting the fact that local learning with kernel machines can be more accurate than SVM [25]. In FaLK-SVM[6] a set of local SVMs is trained on redundant neighborhoods in the training set selecting at testing time the most appropriate model for each query point. The global separation function is sub-divided in solutions of local optimization problems that can be handled very efficiently.

[6] FaLKM-lib [26] source is available at `http://disi.unitn.it/~segata/FaLKM-lib`

This way, all points in the local neighbourhoods can be considered without any computational limitation on the total number of SVs which is the major problem for the application of SVM on large and very large datasets.

KEA [11] represents the state-of-the-art for keyphrase extraction tasks and it is based on the bag-of-words concept. The Bayes theorem is used to compute a probability of a phrase to be a keyphrase using classical definition of probability: quantity of positive cases divided by overall quantity. So in the end each phrase in the text has a probability to be a keyphrase. After that KEA takes top q of phrases and calls them keyphrases. Naive Bayes learning is widely used for instance for the spam filter tasks.

There are some more accurate machine learning used for classification, for instance so called "Bayesian learning", most promising techniques are RVM, or relevance vector machine [27] or Gaussian Processes [28] that may potentially improve the accuracy of prediction, but they are too slow comparing even with SVM.

4.3 Training with unbalanced class cardinalities

In keywords and keyphrases extraction tasks it is natural that the number of key elements (and of candidate key elements) is dramatically lower than the number of non-key elements. In our dataset the keyphrases represent about the 1.8% of the total number of phrases taken into account, meaning that we have more than 55 times fewer positive examples than negative ones. Classification with unbalanced datasets is a challenging task which is very often handled assigning different misclassification costs to the classes.

In SVM classification this is possible associating different soft-margin regularization parameters (C) to the classes as discussed for the first time in [22]. Obviously the assignment of different regularization parameters to each class enlarges the set of parameters that needs to be tuned during model selection phase.

The same approach can be used for FaLK-SVM with the only difference being that the soft-margin regularization parameters (C) are set locally.

Random Forest also can handle unbalanced data using "weight" parameter in implementation [20].

4.4 Result assessment methodology

Usual IR performance measures are Precision, Recall and F-Measure. Defining with A the set of true keyphrases that have not been recognized as keyphrases, with B the correctly recognized keyphrases and with C the set of phrases incorrectly recognized as keyphrases, the formal definition of precision (P), recall (R) and F-measure (FM) are:

$$P = 100\% \cdot \frac{B}{B + C} \tag{1}$$

$$R = 100\% \cdot \frac{B}{B+A} \tag{2}$$

$$FM = \frac{2P \cdot R}{P+R} \tag{3}$$

Here it is important to underline, that we consider each phrase as occurrence individually. This means that one phrase may be included into the paper text several times as a keyphrase, and several times as not keyphrase. It is difficult to judge whether a phrase is a keyphrase or not in such approach. The heuristic behind our judging about keyphrase is as follows: if the phrase has been recognized as a keyphrase at least once, we treat one as a keyphrase. We will call P, R and F-measure based on it as *document-based* measures in the future which is also the methodology adopted by KEA [11]. Another approach is to take into account each phrase occurrence separately, obtaining *occurrence-based* precision, recall and F-Measure. This family of measures are not suitable for the final evaluation of a keyphrases extraction method since one phrase can be considered as a keyphrase several times. Even worse, it can be considered few times as a keyphrase, and few times as a non keyphrase. However, occurrence-based F-Measure is effective as measure to be maximized during SVM model selection because it follows the formal representation of numerical SVM data.

5 Experimenntal evaluation

In this section we give the details of the experiments carried out for the analysis of the discussed keyphrase approaches. To assess the results we used standard IR performance measures: Precision, Recall and F-Measure [22, 1].

5.1 Dataset splitting

We divided the whole dataset of 2000 documents into 3 sets: training set (TR), validation set (VS) and testing set (TS) respectively with 1400, 200 and 400 documents each. To investigate the optimal dataset size we further divided the training set into 7 subsets of 200 documents each. All sets are selected randomly assuring however the balancing with respect to the year of publication, namely assuring that all the sets have the proportional quantity of papers published in a given year.

5.2 Experiment 1. Comparison of ML Methods Enhanced by NLP

Random Forest: four parameters to tune are the number of trees in the ensemble I, the splitting parameter K, the balancing parameter w and the depth of a tree d. Experimentally we discovered the following tricks to reduce training tries:

- take 3 different K parameters: default, half and double of default;

Table 4. The results for SVM, FaLK-SVM, RF and KEA. Best values in bold

	Precision	Recall	F-Measure
FaLK-SVM	24.59%	35.88%	29.18%
SVM	22.78%	**38.28**%	28.64%
Random Forest	**26.40**%	34.15%	**29.78**%
KEA (best q)	18.61%	26.96%	22.02%

- stop the algorithm as soon as an increase in the number of trees does not improve significantly the solution;
- the depth of tree usually should not overcome the quantity of selected features, and should not be much smaller than them.

We found experimentally the best tuning ranges. We used the fast open source implementation[7] compatible with WEKA [15].

SVM: the hyper-parameters we tune are the regularization parameters of the positive and negative classes (C^+ and C^- respectively) and the width σ of the RBF kernel. These parameters are selected using 10 fold cross-validation with a three-dimensional grid search in the space of the parameters. The model selection is performed maximizing in this parameter space the occurrence-bases F-Measure. For SVM training and prediction we use LibSVM [29].

FaLK-SVM: in addition to the SVM parameters (C^-, C^+ and σ) we have to set the neighborhood size k used for the local learning approach. Model selection is thus performed as described for SVM but using a four-dimensional grid-search.

KEA: KEA has one tuning parameter q which is threshold, experimentally we have found that $q = 5$ produces the best F-Measure (see Table 5.2).

Table 5. KEA results for different threshold q values. The best precision, recall and F-Measure among all q values are in bold.

q	P, %	R, %	F-Measure, %
6	17.31	**30.10**	21.98
5 (default)	18.61	26.96	**22.02**
3	21.47	18.41	19.98
2	**24.87**	14.41	18.25

Table 4 summarizes the results. We see that the best result in F-Measure is achieved with the Random Forest using all 20 proposed NLP features. FaLKM and SVM follow very closely while KEA is much lower. The difference between three best methods is not very big (RF outperforms SVM by about 4%), therefore it is important to understand what are the most important factors: particular features or peculiarities of the dataset. These has led us to investigate both dimensions in the next set of experiments.

[7] `http://code.google.com/p/fast-random-forest/`

Fig. 5. F-Measure behaviour with dataset size growth.

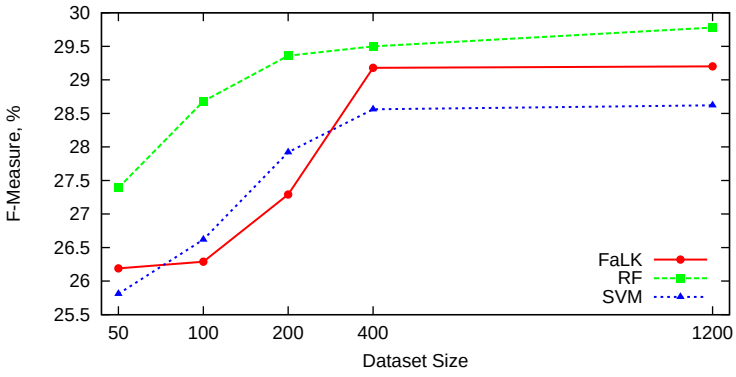
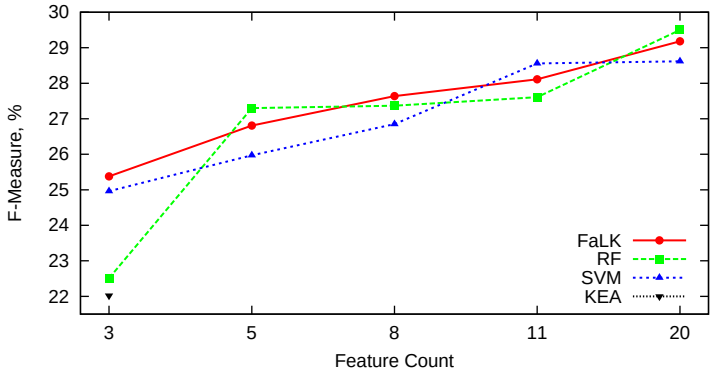


Fig. 6. F-Measure behaviour with feature count growth.



5.3 Experiment 2. Training set size analysis

An increase in training set size may bring an improvement of prediction quality. However, training on a large amount of data is computationally expensive. Thus it is relevent to estimate which dataset size is enough to obtain the best prediction performance. To study this, we carried out experiments at increasing training set sizes as summarized in Figure 5. One can see that i) F-Measure improves as the training set size increases; ii) the improvement levels off after ca. 400 documents. We can conclude that for the task of keyphrases extraction it is important to have rather large training sets, but training sets with more than 400 documents are very likely to experience computational difficulties without a relevant increase in prediction ability.

5.4 Experiment 3. NLP features analysis

In this experiments we analyze the individual effect of the features on the prediction ability. We performed experiments omitting features one by one and monitoring the effect on the overall quality. Assuming that our features are logically grouped we decided to exclude the following groups of features sequentially:

- Arcs (11 features left)
- Head POS Tags (8 features left)
- POS tags (5 features left)
- TFxIDF and relative position (3 features left).

Figure 6 summarizes the results. We see that in case of Random Forest using only first three features decreases F-Measure essentially to KEA results. This is very interesting, because bayesian learning of KEA is only possible considering just 3 features (an increase of features quantity will break KEA). In our comparison of four methods we have two "statistical learning" methods (SVM and FaLK-SVM), and two "probabilistic" methods which give close results having the same three basic features that regard simple count of tokens. Figure 6 shows that the various methods capture different features in different ways: arcs are important for Random Forest and POS tags a most important for SVM. Moreover, while there is a tendency for the overall quality to level off, the experiments do not indicate clearly to have reached a "plateau" behavior (other relevant features may be found).

6 Comparative analysis of extracted keyphrases

Aside from the detailed quantitative analysis of the standard quality measures used to compare the different approaches performance, some important insights on the specific characteristics of each approach, can be gained by a direct analysis of the extracted keyphrases. To this end we have focused our attention on the best results obtained respectively by KEA and by our proposed approach (RF+NLP).

In Table 6 we have collected statistics from our experiments related to correctly and incorrectly recognized keyphrases. Figure 7 and Figure 8 show the distribution of the numbers of correctly and incorrectly extracted keyphrases per document by all four approaches, respectively.

From these data, apart from the generic improvements in the recognition performance of the proposed SVM+NLP approach already presented in Table 4 and discussed in the previous section, we can identify some other interesting characteristics and differences in the two approaches, namely:

- There is an overlap between the extracted true keyphrases in the two approaches: approximately 57% of RF+NLP correct keyphrases are also extracted by the KEA system. However, there exists a significant number of distinct keyphrases extracted only by KEA and only by RF+NLP. This can also be seen in last 2 rows of the Table 6. Let us call varieties of

Table 6. Comparison between Found/Not found keyphrases counts for the best results.

Keyphrases type	Keyphrases count
KEA Correct	360
RF Correct	463
KEA Incorrect	1575
RF Incorrect	1280
ACM Total	1332
Correct keyphrases overlapped for KEA *and* RF	264
Correct keyphrases uniquely for KEA *or* RF	559

correctly extracted by KEA keyphrases as (KEA-c) and by RF+NLP approach as (RF+NLP-c). The intersection of (KEA-c) and (RF+NLP-c) (see next-to-last row) contains 264 keyphrases, while the union of (KEA-c) and (RF+NLP-c) (the last row) is nearly two times bigger. That allows us hypothesize that a combination of the two approaches may improve keyphrases extraction performance.

- As already noticed in Table 4 RF+NLP recognizes less incorrect keyphrases than KEA or other methods. In addition we can notice from Figure 8 that ML+NLP approaches have a lot more documents with few or zero incorrectly recognized keyphrases, while KEA often makes 4 or 5 errors;
- Both approaches provide a similar "coverage" of correctly extracted keyphrases per document. Here "coverage" means the property of a given extraction approach to identify "at least" one correct keyphrase per document. In fact the total number of documents with correctly extracted keyphrases increases from 66% in KEA to 73% in RF+NLP.

From these observations we can conclude that KEA looses to ML+NLP approaches in precision and in recall both by 44% and 30% respectively. From the recall viewpoint KEA is more competitive with ML+NLP due to the higher number of extracted keyphrases it proposes, thus causing a much lower precision. Thus, even if there is a rather relevant number of correct keyphrases extracted only by KEA than can potentially enhance the ML+NLP recall, combining the two approaches might not be a good idea, because incorrectly recognized keyphrases must be merged too and thus the final precision (and the F-Measure) sensibly decreases.

As a last analysis, we have explored qualitatively the keyphrases extracted by RF+NLP method. First, we have looked at the best results: as examples of the type of correct matches between original ACM keyphrases stems and correctly extracted keyphrases stems, we collected in Table 7 the top 5 results[8]: 2 documents have 100% match, and 3 documents with almost all keyphrases recognized. We see that unrecognized keyphrases are synonyms or related to the recognized ones, for instance instead of "bayesian network" we have "bayesian multinet", or "hierarchical representation" instead of "knowledge representation".

[8] Keyphrases in the table are stemmed with S-Removal stemmer

Table 7. Examples of *top* results for RF+NLP approach

#	ACM Keyphrases stems	RF+NLP Keyphrases stems
1	data clustering, constructive induction, bayesian network, em algorithm	*data clustering, constructive induction, bayesian network, em algorithm*, bayesian multinet
2	creg, register allocation, graph coloring, register	*creg, register allocation, graph coloring*
3	software prefetching, software pipelining, vliw machine, locality analysi	*software prefetching, software pipelining, vliw machine,* modulo scheduling
4	two-variable fragment, controlled language, natural language, logic	*two-variable fragment, controlled language, natural language*
5	order-sorted logic, knowledge representation, terminological knowledge, resolution system	*order-sorted logic, knowledge representation, terminological knowledge*, label-based formula, hierarchical representation

Table 8. Examples of *bad* results for RF+NLP approach

#	ACM Keyphrases stems	RF+NLP Keyphrases stems
1	distributed environment, persistent object, distributed system, modularity, object-oriented approach, distributed programming system, replication, migration	*distributed environment, persistent object, distributed system*, database system
2	flexible transaction, concurrency control, transaction management, serializability	*flexible transaction, concurrency control*, heterogeneou distributed database, distributed database, distributed database environment, database environment, database system, multidatabase system, multidatabase
3	knowledge-based query processing	knowledge-based approach
4	security, partial equivalence relation, semantic, powerdomain, noninterference	secure information, secure information flow, information flow, sequential program, security properti
5	reflection, program transformation	generic reification technique, reflective language

Fig. 7. Comparison of KEA and ML+NLP distributions of correctly extracted keyphrases per document.

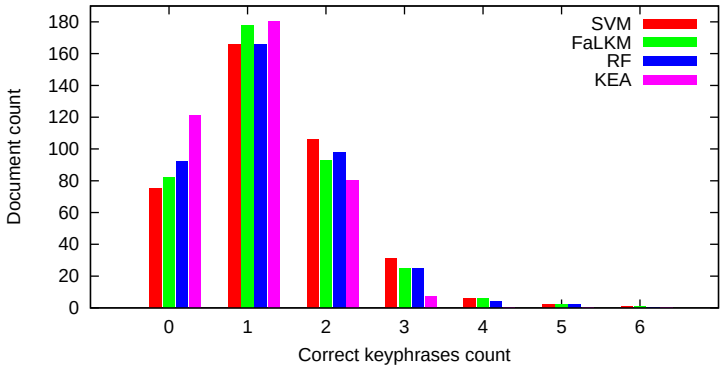
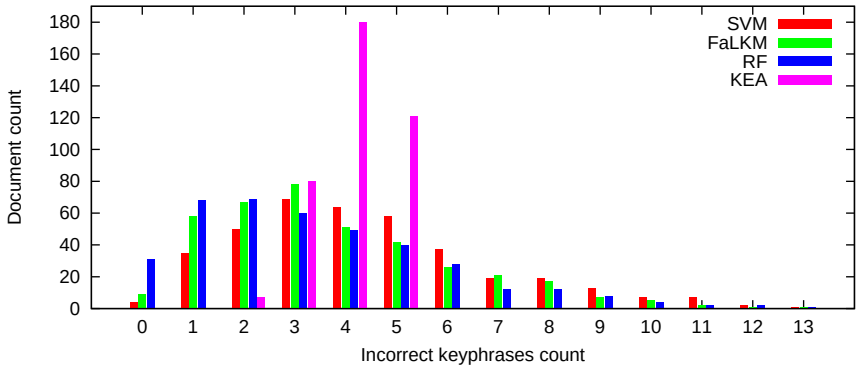


Fig. 8. Comparison of KEA and ML+NLP distributions of incorrectly extracted keyphrases per document.



More interesting is to look at the *bad* results of our approach, that is at the results at the bottom of the distribution where no correct keyphrases (the ones included in the ACM list) has been extracted. Table 8 provides 5 examples of such results. In this case we cannot say "the bottom 5" because we have a tail of equally *bad* results, so we have randomly selected 5 examples among all documents with no correctly extracted keyphrases. For completeness we report that we had 92 such documents out of 400 total. A preliminary qualitative analysis of the data in the table confirms the lack of correct keyphrases. However, the extracted keyphrases seem related to the human assigned keyphrases that we use for assessing the quality of our approach. Specifically, it seems that the human assigned keyphrases may be more general and tend to have a higher level of abstraction while the keyphrases extracted from the actual text via RF+NLP tend

to be more specific. For instance: "security → secure information", "reflection → reflective language".

This behavior, if confirmed by further and more comprehensive data, would enhance the trust in the use of automatic keyphrases extraction. Automatically extracted keyphrases cannot really be used *instead* of assigned ones, but they potentially could enhance them. We know that it is not possible to draw out general trends from such a limited number of instances. However, exploration and analysis of the intrinsic qualities of automatically extracted keyphrases seems an interesting direction for future work.

7 Conclusions

In this paper we applied NLP-based knowledge to a number of different ML methods: traditional SVM, a local variant of SVM and Random Forests for automatic keyphrases extraction from scientific documents. The proposed NLP-based approach shows promising results. We have performed a detailed evaluation of the performance of all ML methods by comparing the extracted keyphrases with human assigned keyphrases on a subset of 2000 ACM papers in the Computer Science domain. The evaluation shows that:

i) The best results of NLP-based ML methods always outperform KEA in all quality parameters: Precision, Recall and overall F-measure; in particular, it improves the average F-Measure from 22.02% (KEA) to 29,78% (Random Forrest) without the use of controlled vocabularies;
ii) A combination of KEA and RF may produce interesting result because both capture different keyphrases.
iii) Feature removal leads to stable decrease of F-Measure for all considered machine learning methods.
iv) Training set size increase improves F-Measure and reaches a "plateau" with training set size around 400 documents.
v) Random Forests is a good tradeoff between quality of keyphrases extraction and computational speed.
vi) NLP helps to avoid "strange" keyphrases candidates often extracted by a purely statistical systems; For instance KEA in some cases recognizes two keyphrases: "ad" and "hoc" instead of capturing "ad hoc" as the whole phrase.
vii) NLP is computationally expensive, but it provides more accurate keyphrases and proposed NLP heuristics cut search space by 50%.

The proposed hybrid ML+NLP approach may also be valid with different data like news, emails, abstracts and web pages. One limitation of the present work (and of all the works based on the instance learning) is in the assumption of the presence of the searched keyphrases inside the documents (assumption that has been used in the construction of our dataset). Indeed, our learning method cannot find (without additional supporting knowledge) a specific keyphrase in a document when the document does not contain at least one instance of the

keyphrase. To tackle such a challenging keyphrase assignment task one needs to take into account documents or keyphrases similarities. For example, one may forecast that documents with similar topics may have similar keyphrases. Alternatively, we have to move from syntactic to semantic relations between words in order to access (implicitly) related keyphrases. Possible future work may also focus on deeper analysis of a quality of extracted keyphrases, because "incorrect" ones are not necessarily the "bad" ones. There is a chance they may be better than author or editor assigned keyphrases.

8 Acknowledgments

Authors thank Professor C. Lee Giles[9] (Penn State University, USA) for sharing meta information about the papers and full text of the papers in PDF and Professor Raymond Ng[10] (University of British Columbia, Canada) for the useful discussions.

References

1. Han, H., Giles, L., Manavoglu, E., Zha, H., Zhang, Z., , Fox, E.: Automatic document metadata extraction using support vector machines. In: JCDL. (2003)
2. Lawrence, S., Giles, C.L., Bollacker, K.: Digital libraries and autonomous citation indexing. IEEE Computer **32**(5) (1999) 67–71
3. Tourney, P.: Mining the web for lexical knowledge to improve keyphrase extraction: Learning from labeled and unlabeled data. Tech. Rep., NRC-44947/ERB-1096 (August 2002)
4. Wang, J., Peng, H.: Keyphrases extraction from web document by the least squares support vector machine. In: ICWI. (2005)
5. Hulth, A.: Improved automatic keyword extraction given more linguistic knowledge. In: EMLNP. (2003) 216–223
6. Hulth, A., Karlgren, J., Jonsson, A., Bostrom, H., Asker, L.: Automatic Keyword Extraction Using Domain Knowledge. Springer (2004)
7. Krapivin, M., Marchese, M., Yadrantsau, A., Liang, Y.: Automated key-phrases extraction using domain and linguistic knowledge. In: ICDIM. (2008)
8. Morton, T.: Using Semantic Relations to Improve Information Retrieval. PhD thesis, University of Pennsylvania (2005)
9. Nivre, J., Hall, J., Nilsson, J., Chanev, A., Eryigit, G., Kübler, S., Marinov, S., Marsi, E.: Maltparser: A language-independent system for data-driven dependency parsing. Natural Language Engineering **13**(2) (2007) 95–135
10. Krapivin, M., Autayeu, A., Marchese, M.: Large dataset for keyphrases extraction. Technical Report DISI-09-055, DISI, Trento, Italy (May 2008) http://eprints.biblio.unitn.it/archive/00001671/01/disi09055-krapivin-autayeu-marchese.pdf.
11. Witten, I., Paynte, G., Frank, E., Gutwin, C., Nevill-Manning, C.: Kea: Practical automatic keyphrase extraction. In: DL. (1999)

[9] `http://clgiles.ist.psu.edu/`
[10] `http://people.cs.ubc.ca/~rng/`

12. Medelyan, O., Witten, I.: Thesaurus based automatic keyphrase indexing. In: JCDL. (2006)
13. Braams, J.: Learning to extract keyphrases from text. Technial report **NRC**(ERB-1057) (February 1999)
14. Ercan, C., Cicekli, I.: Using lexical chains for keyword extraction. Information Processing and Management **43**(6) (2007) 1705–1714
15. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann, San Francisco (2005)
16. Turney, P.: Coherent keyphrase extraction via web mining. In: IJCAI. (2003) 434–439
17. Breiman, L.: Bagging predictors. Machine Learning **24**(2) (1996) 123–140
18. Schapire, R.E., Freund, Y., Bartlett, P., Lee, W.S.: Boosting the margin: A new explanation for the effectiveness of voting methods. The Annals of Statistics **26**(5) (1998) 1651–1686
19. Statistics, L.B., Breiman, L.: Random forests. In: Machine Learning. (2001) 5–32
20. Chen, C., Liaw, A., Breiman, L.: Using random forest to learn imbalanced data. Technical report, Department of Statistics, UC Berkeley (2004)
21. Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning **20**(3) (1995) 273–297
22. Osuna, E., Freund, R., Girosi, F.: Support Vector Machines: Training and Applications. In: CVPR. (1997) 130
23. Segata, N., Blanzieri, E.: Fast Local Support Vector Machines for Large Datasets. In: MLDM 09. Number 5632 in LNCS, Leipzig, Germany, Springer (2009) 295–310
24. Blanzieri, E., Melgani, F.: Nearest neighbor classification of remote sensing images with the maximal margin principle. IEEE Transactions on Geoscience and Remote Sensing **46**(6) (2008) 1804–1811
25. Segata, N., Blanzieri, E.: Fast and Scalable Local Kernel Machines. Technical Report DISI-09-072, University of Trento (2009)
26. Segata, N.: FaLKM-lib v1.0: a Library for Fast Local Kernel Machines. Technical report, University of Trento (2009) `http://disi.unitn.it/~segata/FaLKM-lib/`.
27. Tipping, M.E.: Sparse bayesian learning and the relevance vector machine. Journal of Machine Learning Research **1** (2001) 211–244
28. Williams, C.K.I., Barber, D.: Bayesian classification with gaussian processes. IEEE Trans. Pattern Anal. Mach. Intell. **20**(12) (1998) 1342–1351
29. Chang, C., Lin, C.: LIBSVM: a library for support vector machines. (2001)