Keyphrases Extraction from Scientific Documents: Improving Machine Learning Approaches with Natural Language Processing

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

DISI, University of Trento, Italy

Abstract. In this paper we use Natural Language Processing techniques to improve different machine learning approaches (Support Vector Machines (SVM), Local SVM, Random Forests) to the problem of automatic keyphrases extraction from scientific papers. For the evaluation we propose a large and high-quality dataset: 2000 ACM papers from the Computer Science domain. We evaluate by comparison with expert-assigned keyphrases. Evaluation shows promising results that outperform state-of-the-art Bayesian learning system KEA improving the average F-Measure from 22% (KEA) to 30% (Random Forest) on the same dataset without the use of controlled vocabularies. Finally, we report a detailed analysis of the effect of the individual NLP features and data set size on the overall quality of extracted keyphrases.

1 Introduction

The rapid growth of information on the web has made autonomous digital libraries possible and popular. Autonomy here means automatic information harvesting, processing, classification and representation and it brings several information extraction challenges.

A specific challenge lies in the domain of scholarly papers [1] collected in autonomous digital libraries[1]. Library spider crawls the web for scientific papers. Having downloaded the paper, crawler converts it into a text format. Then relevant metadata (like title, authors, citations) are extracted and finally documents are classified, identified and ranked. Metadata helps to categorize papers, simplifying and improving users' searches, but often metadata is not available explicitly. Information extraction from scholarly papers contains two broad classes of tasks: i) recognition of structural information which is present inside the paper body (like authors, venues, title); ii) 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. In particular, we analyse the effect of the use of Natural Language Processing (NLP) techniques and the use of specific NLP-based

[1] `http://citeseer.ittc.ku.edu`, `http://scholar.google.com`, `http://rexa.info`

heuristics to improve current Machine Learning (ML) approaches to keyphrases extraction. Machine learning methods are successfully used to support automatic information extraction tasks for short news, mails, web pages [2], and for the problem of keyphrases extraction [3–5]. 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 [6] describing our initial concepts and presenting initial results obtained using standard SVM approaches. In this paper:

1. We extend the use of state-of-the-art NLP tools [7] for the extraction, definition and use of linguistic-based features such as part of speech and syntactic relations extracted by dependency parsers [8].
2. We apply the proposed NLP-based approach to different ML methods: traditional SVM, innovative Local SVM and Random Forests.
3. We define and publish a large and high-quality dataset of 2000 documents with expert-assigned keyphrases in the 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.

An extended version of this paper is also available as a technical report[2].

2 Related Work

The state-of-the-art system for keyphrases extraction is KEA [9]. It uses Naïve Bayes classificator and few heuristics. The best results reported by KEA team show about 18% of Precision [9] in extracting keyphrases from generic web pages. Use of domain specific vocabularies may improve the result up to 28.3% for Recall and 26.1% for Precision [10].

Turney suggested another approach which uses GenEx algorithm [11]. The GenEx algorithm is based on a combination of parameterized rules and genetic algorithms. The approach provides nearly the same precision and recall as KEA. In a more recent work [2], 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 problematic for Digital Libraries with millions of documents. In this approach the author measures the results by the ratio of average number of correctly found phrases vs. total number of extracted phrases.

Recent works by A. Hulth et al. considered domain [5] and linguistic [4] knowledge to search for relevant keyphrases. In particular, [5] used a thesaurus

[2] `http://disi.unitn.it/~krapivin/2010keyphrases-krapivin-etal.pdf`

to capture domain knowledge. But the recall value reported in this work is very low, namely 4-6%. The approach proposed in [4] 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 fairly good results (F-Measure up to 33.9%). However, it is hard to compare their results with others because of 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 about the application of linguistic knowledge to the specific problem is reported in [12]. The authors used WordNet and "lexical chains" based on synonyms and antonyms. Then they applied decision trees as an 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. Other ML technique, least square SVM [3] 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.

Our work contributes along three dimensions: 1) using a large high-quality freely available dataset which can set up a ground for further comparison; 2) empowering machine learning methods with NLP techniques and heuristics; 3) applying ensemble learning method (namely RF) that have never been applied to the specific keyphrases extraction problem before; and 4) providing a detailed investigation of the results like features importance, varying dataset size.

3 Dataset Description, Characterization and Linguistic Processing

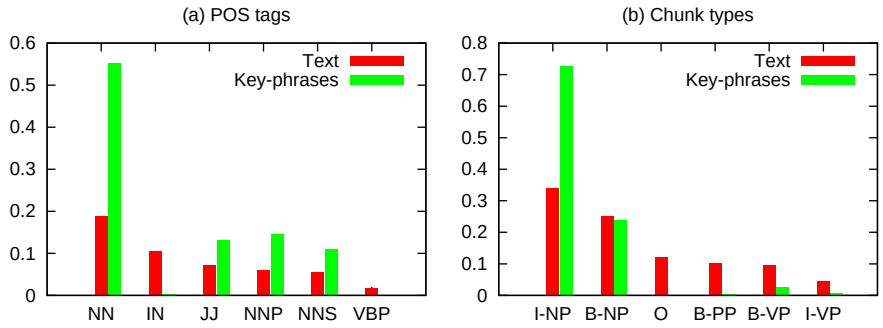
3.1 Dataset Description and Characterization

The dataset presented 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[3] 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 possible 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.

Our experimental dataset consists of 2000 documents with keyphrases assigned by ACM authors and verified by ACM editors. Keyphrases fall into two categories: i) author assigned: located inside each document in the header sections after the prefix "Keywords:"; ii) editor assigned: manually assigned by human experts in a particular domain. 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

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

Fig. 1. Distributions for normal text and keyphrases.



more challenging case of *completely* implicit extraction [4]. In our dataset, each document has on average about 3 unique human assigned keyphrases inside.

3.2 Linguistic Analysis of Keyphrases

We performed an NLP analysis of the keyphrases to study their linguistic properties. This forms the base for heuristics choice and definition of the features used in the machine learning step. This improves the quality of generated keyphrase candidates while simultaneously reducing their quantity.

We analysed 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. 1 shows POS tag and chunk type distributions, respectively for the most common POS tags, such as nouns: NN, NNP, NNS; prepositions: IN, adjectives: JJ and verbs: VBP; and 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 the intuition that most 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. This is also confirmed by additional analysis we have done with MaltParser to explore differences between dependencies of keyphrases and the ones of normal text.

3.3 Text Processing

Before any extraction task, text needs to be pre-processed [13]. Pre-processing includes sentence boundary detection, tokenization, POS tagging, chunking, pars-

[4] to tackle such a challenging implicit extraction one could move from syntactic to semantic relations between words in order to access (implicitly) related keyphrases.

ing, stemming and recognizing separate blocks inside the article, such as Title, Abstract, Section Headers, Reference Section, Body.

We used OpenNLP suite [7] 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. 1a, 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.

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

In addition, we apply MaltParser to extract dependencies which we use as additional features for machine learning. Finally, we use S-removal stemmer (from KEA) to avoid well-known issues related to the presence of the same words written in different forms.

4 Enhancing Machine Learning with Natural Language Processing

4.1 Features Selection

Proper feature space selection is a crucial step in information extraction. Table 2 details the feature set we propose. Features 1, 2 and 3 are common and widely used in most information extraction systems [14]. Less traditional features include feature 4 – quantity of tokens in a phrase [1] and 5 – part of a text [3]. The features numbered in Table 2 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, respectively, to the first, second and third token of the candidate. 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 dependant and a labelled arc joining them. Dependencies help us to capture the relations between tokens, the position and the

Table 2. 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

role of the keyphrase in the sentence. Specifically 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 the sentence. Feature 13 refers to the distance between the first keyphrase token and its dependant if it exists. As a distance we take the difference between token indices. Feature 14 refers to the distance between the first token and its head.

Differently from many text classification approaches based on the "bag-of-words" model, which encodes the text in a binary vector showing which words are present and causes very high dimensionality of the data, we work here with only 20 features.

4.2 Machine Learning Methods Used for Comparison

Random Forest (RF) [15] is based on the core idea of building many decision tree classifiers and voting for the best result. RF extends the "bagging" approach [16] with random selection of features sets. The RF algorithm does not need costly cross-validation procedure and it is considered scalable and fairly fast. RF is one of the state-of-the-art approaches for classification tasks.

Support Vector Machines (SVMs) [17] are classifiers with sound foundations in statistical learning theory [18] considered, with RF, the state-of-the-art classification method. The reasons of SVM popularity include the possibility of customizable input space mapping (with kernels), handling of noisy data and robustness to the curse of dimensionality. When the dimensionality is high, a linear classifier is often the best choice, while with a reduced number of features a non-linear approach works best. We adopt SVM with the highly non-linear Gaussian (RBF) kernel, because in our case the dimensionality of data is low and linear separation in the input space gives poor results.

FaLK-SVM [19] is a kernel method based on Local SVM [20] 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 [21]. In FaLK-SVM[5] a set of local SVMs is trained on redundant neighbourhoods in the training set selecting at testing time the most suitable model for each query point. The global separation function is subdivided in solutions of local optimization problems that can be handled very efficiently. This way, all points in the local neighbourhoods can be considered without any computational limit on the total number of support vectors which is the major problem in applying SVM on large and very large datasets.

KEA [9] represents the state-of-the-art for key-phrase extraction tasks and is based on the bag-of-words concept. The Bayes theorem is used to estimate a probability of a phrase to be a keyphrase using the frequencies of the training set. 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 considers them to be keyphrases. Naïve Bayes learning is widely used for other text-oriented tasks like spam filtering.

5 Experimental Evaluation

In this section we detail 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 [18, 1]. We divided the whole dataset of 2000 documents into 3 major sets: training set (TR), validation set (VS) and testing set (TS) respectively with 1400, 200 and 400 documents each. To find out the best dataset size we further divided the training set into 7 subsets of 200 documents each.

5.1 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. By trial and error we defined the following strategy to lessen training tries: (1) take 3 different K parameters: default, half and double of default; (2) stop the algorithm as soon as an increase in the number of trees does not improve significantly the solution; (3) the depth of tree usually should not overcome the quantity of selected features, and should not be much smaller than them. We used the fast open source implementation[6] compatible with WEKA [13].

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 [23].

FaLK-SVM: as well as the SVM parameters (C^-, C^+ and σ) we have to set the neighbourhood size k used for the local learning approach. Model selection is

[5] FaLKM-lib [22] source is available at `http://disi.unitn.it/~segata/FaLKM-lib`

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

Table 3. SVM, FaLK-SVM, RF and KEA results. 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%

thus performed as described for SVM, but using a four-dimensional grid-search. It is important to mention that FaLK-SVM and SVM have the parameters fully automatically tuned.

KEA: KEA has only one tuning parameter q which is the threshold; experimentally we have found that $q = 5$ produces the best F-Measure. Table 3 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. Since the difference between best three methods is not very big (RF outperforms SVM by ca. 4%) it is important to understand which are the most important factors: particular features or peculiarities of the dataset. These has led us to explore both dimensions in the next set of experiments.

5.2 Experiment 2. Training Set Size Analysis

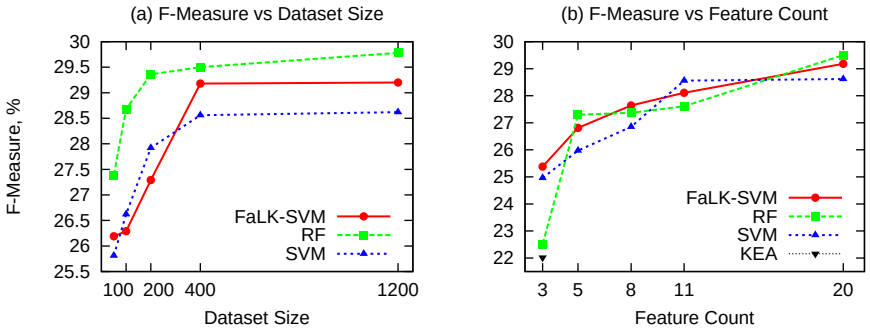
An increase in training set size may bring an improvement of prediction quality. However, training on a large amount of data is computationally expensive. Thus it is relevant to estimate which dataset size is enough to get the best prediction performance. To study this, we carried out experiments at increasing training set sizes as summarized in Figure 2a. 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 computationally expensive without a relevant increase in prediction quality.

5.3 Experiment 3. NLP Features Analysis

In this experiments we analyse 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: i) Arcs (11 features left) ii) Head POS Tags (8 features left) iii) POS tags (5 features left) iv) TFxIDF and relative position (3 features left).

Figure 2b summarizes the results (KEA has just 3 features so there is a just one point on plot). We see that in case of Random Forest using only the first three features decreases F-Measure essentially to KEA results. This is very interesting, because bayesian learning of KEA consideres just 3 features (we made a try to

Fig. 2. F-Measure behaviour with dataset size and feature count growth. Triangle in b) shows 3 features of KEA.



increase of features quantity in Naïve Bayes approach, but with no significant success). 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 2b 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 show clearly to have reached a "plateau", thus other relevant features may be found.

6 Conclusions

In this paper we applied NLP-based knowledge to several different ML methods 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-powered 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% (KEA) to 30% without the use of controlled vocabularies;
ii) Feature removal leads to stable decrease of F-Measure for all considered machine learning methods.
iii) Larger training set improves F-Measure and it reaches a "plateau" with training set size around 400 documents.
iv) Random Forests is a good tradeoff between quality of keyphrases extraction and computational speed.

The proposed hybrid ML+NLP approach may also be valid with different and more specific data like news, emails, abstracts and web pages.

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. 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)
3. Wang, J., Peng, H.: Keyphrases extraction from web document by the least squares support vector machine. In: ICWI. (2005)
4. Hulth, A.: Improved automatic keyword extraction given more linguistic knowledge. In: EMLNP. (2003) 216–223
5. Hulth, A., Karlgren, J., Jonsson, A., Bostrom, H., Asker, L.: Automatic Keyword Extraction Using Domain Knowledge. Springer (2004)
6. Krapivin, M., Marchese, M., Yadrantsau, A., Liang, Y.: Automated key-phrases extraction using domain and linguistic knowledge. In: ICDIM. (2008)
7. Morton, T.: Using Semantic Relations to Improve Information Retrieval. PhD thesis, University of Pennsylvania (2005)
8. 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
9. Witten, I., Paynte, G., Frank, E., Gutwin, C., Nevill-Manning, C.: Kea: Practical automatic keyphrase extraction. In: DL. (1999)
10. Medelyan, O., Witten, I.: Thesaurus based automatic keyphrase indexing. In: JCDL. (2006)
11. Braams, J.: Learning to extract keyphrases from text. Technial report **NRC**(ERB-1057) (February 1999)
12. Ercan, C., Cicekli, I.: Using lexical chains for keyword extraction. Information Processing and Management **43**(6) (2007) 1705–1714
13. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann, San Francisco (2005)
14. Turney, P.: Coherent keyphrase extraction via web mining. In: IJCAI. (2003) 434–439
15. Statistics, L.B., Breiman, L.: Random forests. In: Machine Learning. (2001) 5–32
16. Breiman, L.: Bagging predictors. Machine Learning **24**(2) (1996) 123–140
17. Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning **20**(3) (1995) 273–297
18. Osuna, E., Freund, R., Girosi, F.: Support Vector Machines: Training and Applications. In: CVPR. (1997) 130
19. 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
20. 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
21. Segata, N., Blanzieri, E.: Fast and Scalable Local Kernel Machines. Technical Report DISI-09-072, University of Trento (2009)
22. 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/`.
23. Chang, C., Lin, C.: LIBSVM: a library for support vector machines. (2001)