Accurate Keyphrase Extraction from Scientific Papers by Mining Linguistic Information

Mounia Haddoud [1, 2], Aïcha Mokhtari [2], Thierry Lecroq[1] and Saïd Abdeddaïm[1]
[1] LITIS, Université de Rouen, 76821 Mont-Saint-Aignan Cedex, France.
[2] RIIMA, USTHB, BP 32, El-Alia, Bab-Ezzouar, 16111 Algiers, Algeria.

Abstract
In this paper we investigate the impact of candidate terms filtering using linguistic information on the accuracy of automatic keyphrase extraction from scientific papers. According to linguistic knowledge, the noun phrases are most likely to be keyphrases. However the definition of a noun phrase can vary from a system to another. We have identified five POS tag sequence definitions of a noun phrase in keyphrase extraction literature and proposed a new definition. We estimated experimentally the accuracy of a keyphrase extraction system using different noun phrase filters in order to determine which noun phrase definition yields to the best results.

Conference Topic
Text mining and information extraction

Introduction
A keyphrase is a sequence of words that describes the content of a document. Applications of automatic keyphrase extraction include digital libraries management, content-based tag recommendation, document retrieval, summarization, document clustering and query expansion. This paper deals with keyphrase extraction from abstracts of scientific papers.

There is two ways of associating keyphrases to articles: keyphrase extraction or keyphrase assignment. The first approach chooses phrases appearing in the text, while the second one, also known as subject indexing and text categorization or classification, assigns keyphrases from terminological databases. Despite the huge literature dealing with keyphrase extraction the accuracy of these methods remains low when compared to keyphrase assignment methods. This is due to the fact that keyphrases are not predefined and must be discovered only in the basis of their distribution in the text. The crucial points about designing an automatic keyphrase extraction system are: 1) filtering all the terms (phrases or *n*-grams) of a document in order to identify candidate terms that could be keyphrases, 2) selecting the properties that could distinguish keyphrases from other terms. These properties (called features) are combined, using machine learning in order to give a synthetic score for each document term. This score is used to rank all the candidate terms, the *k*-top ranked terms are output as the keyphrases we look for.

Both filtering and learning using the features can significantly affect the accuracy of the keyphrase extraction method. Features can be classified according to their nature: statistical, structural and linguistic features. The most used *statistical features* are term length i.e. the number of words it contains, term frequency (TF) in the document, inverse document frequency (IDF) which depends on the number of corpus documents that contain the term, TFIDF which combines TF and IDF, the first position in the document and the co-occurrence frequency of the term with other document terms. *Structural features* that are provided by HTML or XML documents (like apparition in title, document header, hypertext link etc.) can help keyphrase identification. Part-of-speech tags (POS tags) and noun phrases chunks are *linguistic features* that try to capture the linguistic properties of keyphrases.

In this paper we investigate the impact of candidate terms filtering using linguistic information on the accuracy of automatic keyphrase extraction. According to linguistic knowledge, the noun phrases are most likely to be keyphrases. However the definition of a noun phrase can vary from a system to another. In his pioneer work Turney (Turney, 1997) proposed

to keep as candidates the noun phrases corresponding to a POS tag sequence which satisfy the regular expression: `(NN|NNS|NNP|NNPS|JJ)*(NN|NNS|NNP|NNPS|VBG)`. Hulth (Hulth, 2003) considers the 56 POS tag sequences most frequently occurring among keyphrases in the training data and uses them as a criterion for selecting candidate terms. Additionally the POS tag sequence of each term is used as a feature for learning keyphrases. Nguyen et al. (Nguyen & Kan, 2007) consider candidate POS tag sequences of the form `NBAR = (NN|NNS|NNP|NNPS|JJ|JJR|JJS)*(NN|NNS|NNP|NNPS)` and also consider the POS tag sequence of the candidate as a single feature in their set of features. This expression was improved adding the pattern `NBAR IN NBAR` in papers that have followed (Kim & Kan, 2009, Kim, Baldwin, & Kan, 2010). Krapivin et al. (Krapivin, Marchese, Yadrantsau, & Liang, 2008) use the POS tag of each token in a phrase as features in order to classify it as keyphrase or not. Liu et al. (Liu, Li, Zheng, & Sun, 2009) filter candidate terms that correspond to `(JJ)*(NN|NNS|NNP)+`. Pal et al. (Pal, Banka, Mitra, & Das, 2011) keep the noun phrases which satisfy the regular expression used by Turney (Turney, 1997) and use the tags as features.

Our observation of the real keyphrases in the used training data suggests us a very large POS tag definition of noun phrases which satisfies the regular expression: `(NN|NNS|NNP|NNPS|JJ|VBN|NN IN|NNS IN)*(NN|NNS|NNP|NNPS|VBG)`. Unlike the other definitions, our noun phrases can contain a verb at the past participle (tag VBN) such as *multi-agent distributed system* (JJ VBN NN) or *unified framework* (VBN NN). Furthermore, our definition imposes that a preposition (tag IN) can only be used after a noun in a keyphrase, as in *quality of service* (NN IN NN) or in *clusters of topics* (NNS IN NNS).

In a previous work (Haddoud & Abdeddaïm, 2014) we developed a supervised learning system which uses 18 statistical features. This system offers the possibility to evaluate the efficiency of keyphrase candidate selection based on linguistic information. Precisely, we will estimate experimentally the accuracy of keyphrase extraction using different noun phrase filters in order to determine which noun phrase definition yields to the best results.

Keyphrase extraction system

Given a document and an integer *k*, the *keyphrase extraction problem* consists in finding *k* terms (phrases or *n*-grams) that best describe the document. Designing a keyphrase extraction system consists in selecting the features that could distinguish keyphrases from other terms.

In a previous work (Haddoud & Abdeddaïm, 2014) we developed a supervised learning system which uses 18 statistical features. Among these features, the document phrase maximality index (DPM-index), a new measure to discriminate overlapping keyphrase candidates, improves the accuracy of our keyphrase extraction system by 9%.

When a keyphrase is an n-gram that contains more than one compound (word), it is frequent that one of them is a specific word to the document. We have defined the feature TFIDFRatio as the ratio between the TFIDF of a term t and the maximum value of the TFIDF of a compound of t. This indicator tends to be small when the term has a compound with high TFIDF. Most of keyphrase extraction systems consider only the first position of a candidate term as a feature. The position of the first occurrence of a term is known to be a very useful feature for keyphrase prediction, however we want to take benefit of the distribution of all its positions in the document. We conjecture that keyphrase positions in the document are clustered differently than other term positions. Thus we propose to use as features the mean of these positions and their 2-means. After trying approximately 30 features used in the literature, we retained 12 features that works well for keyphrase extraction in our experiments. Among these features 4 are rarely used in the literature: SFS, GDC (an adaptation of a widely used measure in terminology extraction), MLE and KLD. We added the 6 proposed features to them obtaining the 18 features we utilize in our system. The Table 3 reviews all the features used in the system, our 6 features are numbered form 13 to 18.

Table 1. Notations used in the paper.

Symbol	*Description*
d	The document, $\|d\|$ the number of words included in
D	The document collection or corpus, $\|D\|$ the number of documents in D
T	The set of all the terms selected from the corpus documents after the preprocessing step, $\|T\|$ its size
T_d	The set of all the terms selected from the document d after the preprocessing step, $\|T_d\|$ its size
t	A term of T, $\|t\|$ the number of words included in
s	A sentence, $\|s\|$ the number of words included in
S_d	The sentences of d, $\|S_d\|$ its size
$S_d(t)$	The sentences of d containing t, $\|S_d(t)\|$ its size
head(d, r)	The head part of the document d of size $r\|d\|$, with $0 < r < 1$
f(t, d)	The frequency of t in the document d
f(t, D)	The frequency of t in the corpus D
df(t, D)	Number of documents of D where t appears (document frequency)
p(t, d)	An estimation of the probability of t given d: $\mathrm{p}(t,d) = \mathrm{f}(t,d)/\sum_{t' \in T_d} \mathrm{f}(t',d)$
p(t, D)	An estimation of the probability of t given D: $\mathrm{p}(t,D) = \mathrm{f}(t,D)/\sum_{t' \in T} \mathrm{f}(t',D)$
$\mathrm{pos_n}(t, d)$	The position of the n-th occurrence of t in d (in number of words preceding it)
$\mathrm{npos_n}(t, d)$	The n-th normalized position: $\mathrm{npos_n}(t,d) = \mathrm{pos_n}(t,d)/\|d\|$
$\mathrm{sent_n}(t, d)$	The number of the sentence containing the n-th occurrence of t in the document d
comp(t)	The compounds of t, i.e., the words of the n-gram t
sub(t)	The subterms of t, i.e., all the m-grams that are contained in (substrings of) the n-gram t, with $m \leq n$
sup(t, d)	The superterms of t in the document d, i.e., all the selected terms s of the document d containing t excepting t
sup(t, D)	The superterms of t in the corpus D, i.e., all the selected terms s of the corpus D containing t, but not equal to t
TF(t, d)	The normalized term frequency of a term in a document d: $\mathrm{TF}(t,d) = \mathrm{f}(t,d)/\|d\|$
IDF(t, D)	The inverse document frequency of t in the corpus D: $\mathrm{IDF}(t,D) = \log(\|D\|/\mathrm{df}(t,D))$
TFIDF(t, d, D)	$\mathrm{TFIDF}(t,d,D) = \mathrm{TF}(t,d) \times \mathrm{IDF}(t,D)$

Table 2. Features used in our system.

#	*Feature*	*Description*
1	$\mathrm{Len}(t) = \|t\|$	Length n of the n-gram t in words
2	$\mathrm{TF}(t,d) = \mathrm{f}(t,d)/\|d\|$	Term normalized frequency
3	$\mathrm{IDF}(t,d,D) = \log(\|D\|/\mathrm{df}(t,D))$	Inverse document frequency
4	$\mathrm{logTFIDF}(t,d,D) = \mathrm{logTF}(t,d) \times \max(0, \log((\|D\| - \mathrm{df}(t,D))/\mathrm{df}(t,D))$	Variant of TFIDF
5	$\mathrm{FP}(t,d) = \mathrm{npos_0}(t,d) = \mathrm{pos_0}(t,d)/\|d\|$	First position
6	$\mathrm{FS}(t,d) = \mathrm{sent_0}(t,d)/\|S_d\|$	First sentence
7	$\mathrm{HF}(t,d,r) = \mathrm{f}(t, \mathrm{head}(d,r))/r\|d\| \quad (r = 0.25)$	Head frequency. The frequency of t in the first quarter part of d
8	$\mathrm{ASL}(t,d) = \sum_{s \in S_d(t)}(\|s\|/\|S_d(t)\|) / \sum_{s \in d}(\|s\|/\|S_d\|)$	Average sentence length
9	$\mathrm{SFS}(t,d) = \sum_{s \in \mathrm{sub}(t)} \mathrm{f}(s,d)/\|d\|$	Substrings frequencies sum
10	$\mathrm{GDC}(t,d) = \|t\|\log(\mathrm{f}(t,d))\mathrm{f}(t,d)/\sum_{c \in \mathrm{comp}(t)} \mathrm{f}(c,d)$	Generalized Dice coefficient
11	$\mathrm{MLE}(t,d) = p(t,d)$	Maximum likelihood estimate
12	$\mathrm{KLD}(t,d,D) = \mathrm{p}(t,d)\log(\mathrm{p}(t,d)/\mathrm{p}(t,D))$	Kullback-Leibler divergence
13	$\text{DPM-index}(t,d) = 1 - \max_{s \in \mathrm{sup}(t,d)}(\mathrm{f}(s,d)/\mathrm{f}(t,d))$	Document phrase maximality index
14	$\text{DPM-TFIDF}(t,d,D) = \text{DPM-index}(t,d) \times \mathrm{TFIDF}(t,d,D)$	DPM-index cross TFIDF
15	$\mathrm{TFIDFRatio}(t,d,D) = \mathrm{TFIDF}(t,d,D) / \max_{c \in \mathrm{comp}(t)}(\mathrm{TFIDF}(c,d,D))$	TFIDF ratio of the term and its main compound
16-18	Position mean and 2-means	k-means of the normalized positions (k = 1,2)

Our keyphrase extraction system utilizes a classifier trained using a supervised machine learning algorithm. Due to the difficulty of the keyphrase extraction problem, instead of using

directly the classifier outputs, one rather utilizes the probability to be classified as a keyphrase (Witten, Paynter, Frank, Gutwin, & Nevill-Manning, 1999). These probabilities are used as scores to generate a ranked list of keyphrases. According to a fixed parameter *k*, the system outputs the *k*-top score terms as predicted keyphrases. We use logistic regression as learning algorithm. We also tried other learning algorithms, for instance bagged C4.5 decision trees, random forests and LogitBoost but in every case logistic regression gives better results. We used the Weka implementation of these methods (Hall et al.., 2009).

Experiments

In order to evaluate our system with different noun phrase filters, we used the SemEval-2010 data for the task 5: Automatic Keyphrase Extraction from Scientific Articles (Kim, Medelyan, Kan, & Baldwin, 2010). These data consist in 244 scientific conference and workshop papers from ACM Digital Library. Papers were selected from four research areas: Distributed Systems, Information Search and Retrieval, Learning and Social and Behavioural Sciences. For each paper at most 15 keyphrases were manually assigned by both paper authors and readers. From this corpus 144 papers are provided for training and the evaluation is done on 100 articles. The main advantage of using SemEval-2010/Task-5 corpus is that we can compare our results to those obtained by 19 teams that participated to the challenge. Furthermore, two recent papers (Newman, Koilada, Lau, & Baldwin, 2012, You, Fontaine, & Barthès, 2013) also used these data.

We followed the procedure given in the challenge for the evaluation of our system. In this task the methods were compared using three exact match evaluation metrics. An exact match evaluation metric measures how well the automatically generated keyphrases match *exactly* the manually assigned ones. More flexible metrics could be used (Kim, Baldwin, & Kan, 2010), however exact match is stricter and enables us to compare our results with those of the twenty one teams. Specifically, the three metrics used are: the *precision* which represents the proportion of the extracted keyphrases that match the manually assigned ones, the *recall* which is the proportion of the keyphrases manually assigned that are extracted by the keyphrase extraction system and the F1-Score is defined as: 2 . *precision* . *recall* / (*precision*+*recall*).

In order to measure the contribution of each possible noun phrase filter to the overall system performance, we represent in Table 3 the results obtained by our method when using each filter.

We considered the following filters:

- Filter 1: keeps the noun phrases corresponding to our proposed POS tag sequence `(NN|NNS|NNP|NNPS|JJ|VBN|NN IN|NNS IN)*(NN|NNS|NNP|NNPS|VBG)`
- Filter 2: `(NN|NNS|NNP|NNPS|JJ)*(NN|NNS|NNP|NNPS|VBG)` (Turney, 1997, Pal, Banka, Mitra, & Das, 2011)
- Filter 3: `NBAR IN NBAR` where `NBAR = (NN|NNS|NNP|NNPS|JJ|JJR|JJS)* (NN|NNS|NNP|NNPS)` (Kim & Kan, 2009, Kim, Baldwin, & Kan, 2010)
- Filter 4: `(JJ)*(NN|NNS|NNP)+` (Liu, Li, Zheng, & Sun, 2009)
- Filter 5: `(NN|NNS|NNP|NNPS|JJ|JJR|JJS)*(NN|NNS|NNP|NNPS)` (Nguyen & Kan, 2007)
- None: keep all phrases (no linguistic filtering)

Table 3. Performances of our system with different noun phrase filters according to precision (P), recall (R) and F1-Score (F).

Filters	*Top 5 candidates*			*Top 10 candidates*			*Top 15 candidates*		
	P	*R*	*F*	*P*	*R*	*F*	*P*	*R*	*F*
Filter 1	44.8%	15.3%	22.8%	**36.2%**	**24.7%**	**29.4%**	**28.3%**	**28.9%**	**28.6%**
Filter 2	**45.0%**	**15.4%**	**22.9%**	33.5%	22.9%	27.2%	26.7%	27.3%	27.0%
Filter 3	42.2%	14.4%	21.5%	33.6%	22.9%	27.3%	26.3%	26.9%	26.6%
Filter 4	43.2%	14.7%	22.0%	32.7%	22.3%	26.5%	25.8%	26.4%	26.1%
None	38.4%	13.1%	19.5%	30.8%	21.0%	25.0%	24.2%	24.8%	24.5%
Filter 5	37.8%	12.9%	19.2%	29.6%	20.2%	24.0%	23.8%	24.4%	24.1%

The performances of the system with each noun phrase filter are given over the numbers of keyphrase candidates: top 5, 10 and 15. The Table 3 shows the performances ranked by the F1-Score over the top 15 keyphrases. According to these experiments, we can see that the proposed filter 1 gives the best results. For the top 5 candidates, filter 2 gives slightly better results but the difference of 0.1 % in the F1-Score is not significant. Note that one can always be restrictive in the definition of the noun phrase filter in order to improve the prediction of the top 5 candidates, however the quality of prediction will decrease significantly when we aim to retrieve more correct keyphrases. The Table 3 shows also that filter 5 imposes so many restrictions that it underperforms the extraction without filtering the candidate terms. When no linguistic filter is used, the learning method does better than filter 5.

The Table 4 shows the performances of our system with the proposed noun phrase definition (filter 1) compared to the 4 other best systems. The 4 systems are the best among 21 systems that include the 19 that participated to the challenge and the two published recently. Our system ranks first over the three numbers of keyphrase candidates and for the three metrics used. For 10 keyphrases, our system yields a 13% improvement compared to HUMB (Lopez & Romary, 2010) in F1-Score. Notice that at the opposite of our system, HUMB uses structural features and different external knowledge features in order to improve its performances. These knowledge bases (GROBID/TEI, GRISP and HAL) are specific to scientific papers. Then the most important regarding to these results is that, by using only statistical features on linguistically filtered terms, our system outperforms the others without loss of generality.

Table 4. Our system compared to the 4 best systems according to precision (P), recall (R) and F1-Score (F).

System	*Top 5 candidates*			*Top 10 candidates*			*Top 15 candidates*		
	P	*R*	*F*	*P*	*R*	*F*	*P*	*R*	*F*
Our system	**44.8%**	**15.3%**	**22.8%**	**36.2%**	**24.7%**	**29.4%**	**28.3%**	**28.9%**	**28.6%**
HUMB	39.0%	13.3%	19.8%	32.0%	21.8%	26.0%	27.2%	27.8%	27.5%
You et al.[1]	-	-	-	-	-	-	26.2%	26.8%	27.5%
WINGNUS	40.2%	13.7%	20.5%	30.5%	20.8%	24.7%	24.9%	25.5%	25.2%
KP-Miner	36.0%	12.3%	18.3%	28.6%	19.5%	23.2%	24.9%	25.5%	25.2%

Conclusion

This paper presents a noun phrase filter for keyphrase extraction. We showed experimentally that this filter improved by 16.7% the ability of our system to extract correct keyphrases. The F1-Score of the keyphrase extraction increases from 24.5% to 28.6% for the top 15 keyphrases. The results show also significative improvement over other filters which we think makes it more flexible and adaptable to other types of text mining problems.

[1] (You, Fontaine, & Barthès, 2013)

References

Haddoud, M., & Abdeddaïm, S. (2014). Accurate keyphrase extraction by discriminating overlapping phrases. J. Information Science, 40(4), 488–500.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The WEKA data mining software: an update. SIGKDD Explorations, 11(1), 10-18.

Hulth, A. (2003). Improved automatic keyword extraction given more linguistic knowledge. In Proceedings of the 2003 conference on empirical methods in natural language processing, Sapporo, Japan, July 11-12, 2003 (pp. 216–223). ACL.

Kim, S. N., Baldwin, T., & Kan, M.-Y. (2010). Evaluating N-gram based evaluation metrics for automatic keyphrase extraction. In Proceedings of the 23rd international conference on computational linguistics, Beijing, China, august 23-27, 2010 (p. 572-580).

Kim, S. N., & Kan, M.-Y. (2009). Re-examining automatic keyphrase extraction approaches in scientific articles. In Proceedings of the ACL 2009 workshop on multiword expressions, Singapore, 6 august 2009 (pp. 9–16). ACL.

Kim, S. N., Medelyan, O., Kan, M.-Y., & Baldwin, T. (2010). SemEval-2010 Task 5: Automatic keyphrase extraction from scientific articles. In Proceedings of the 5th international workshop on semantic evaluation, Uppsala, Sweden, July 15-16, 2010 (pp. 21–26). ACL.

Krapivin, M., Marchese, M., Yadrantsau, A., & Liang, Y. (2008). Unsupervised key-phrases extraction from scientific papers using domain and linguistic knowledge. In Proceedings of the 3rd international conference on digital information management, London, UK, November 13-16, 2008 (pp. 105–112). IEEE.

Liu, Z., Li, P., Zheng, Y., & Sun, M. (2009). Clustering to find exemplar terms for keyphrase extraction. In Proceedings of the 2009 conference on empirical methods in natural language processing, Singapore, August 6-7, 2009 (p. 257-266). ACL.

Lopez, P., & Romary, L. (2010). HUMB: Automatic key term extraction from scientific articles in GROBID. In Proceedings of the 5th international workshop on semantic evaluation, Uppsala, Sweden, July 15-16, 2010 (pp. 248–251). ACL.

Newman, D., Koilada, N., Lau, J. H., & Baldwin, T. (2012). Bayesian text segmentation for index term identification and keyphrase extraction. In Proceedings of the 24th international conference on computational linguistics, Mumbai, India, December 8-15, 2012 (p. 2077-2092).

Nguyen, T. D., & Kan, M.-Y. (2007). Keyphrase extraction in scientific publications. In Proceedings of the 10th international conference on Asian digital libraries, ICADL 2007, Hanoi, Vietnam, December 10-13, 2007 (Vol. 4822, p. 317-326). Springer.

Pal, T., Banka, H., Mitra, P., & Das, B. (2011). Linguistic knowledge based supervised keyphrase extraction. In Proceedings of national conference on future trends in information & communication technology & applications, Bhubaneswar, India, September 10-11, 2011.

Turney, P. D. (1997). Extraction of keyphrases from text: Evaluation of four algorithms (Tech. Rep. No. ERB-1051). National Research Council. Institute for Information Technology.

Witten, I. H., Paynter, G. W., Frank, E., Gutwin, C., & Nevill-Manning, C. G. (1999). KEA: practical automatic keyphrase extraction. In Proceedings of the 4th ACM conference on digital libraries, Berkeley, California, USA, August 11-14, 1999 (pp. 254–255). ACM.

You, W., Fontaine, D., & Barthès, J.-P. A. (2013). An automatic keyphrase extraction system for scientific documents. Knowledge and Information Systems, 34(3), 691-724.