Article

JIS

Journal of Information Science
2014, Vol. 40(4) 488–500

DOI: 10.1177/0165551514530210
jis.sagepub.com
SAGE

Accurate keyphrase extraction by discriminating overlapping phrases

Mounia Haddoud
USTHB, Algeria and Université de Rouen, France

Saïd Abdeddaïm
Université de Rouen, France

Abstract
In this paper we define the document phrase maximality index (DPM-index), a new measure to discriminate overlapping keyphrase candidates in a text document. As an application we developed a supervised learning system that uses 18 statistical features, among them the DPM-index and five other new features. We experimentally compared our results with those of 21 keyphrase extraction methods on SemEval-2010/Task-5 scientific articles corpus. When all the systems extract 10 keyphrases per document, our method enhances by 13% the *F*-score of the best system. In particular, the DPM-index feature increases the *F*-score of our keyphrase extraction system by a rate of 9%. This makes the DPM-index contribution comparable to that of the well-known TFIDF measure on such a system.

Keywords
Information extraction; keyphrase extraction; scientific digital libraries; text mining

1. Introduction

Given a text document, the *keyphrase extraction problem* consists of finding terms (phrases or *n*-grams) that best describe the document. Documents are of different types: scientific papers (most of bibliographical references deal with this document type), news articles [1], meeting transcripts [2], web pages [3], emails [3], social networks [4], etc. Applications of automatic keyphrase extraction include digital libraries management [5], web content-based advertising [6], content-based tag recommendation [7], document and web page retrieval [8], summarization [1], document clustering [9], query expansion [10] and automated sentiment analysis.

In any text extraction problem we are faced with the choice between improving the specificity of an *n*-gram by extending its size *n*, which reduces its frequency, or improving its statistical significance by decreasing its size, which reduces the specificity to the considered document. This choice between overlapping *n*-grams of different sizes needs a salience criterion in order to extract significant terms, which are also specific to the document we are considering. For example, the candidate 3-gram *multiple sequence alignment* contains two 2-grams, *multiple sequence* and *sequence alignment*. These three *n*-grams could have close TFIDF values – how does one choose the real keyphrases among them? Intuitively, *multiple sequence* is a wrong candidate, but *sequence alignment* or *multiple sequence alignment* or both could be in the main topics of the text document. The comparison of the frequencies of these *n*-grams in the document can help a learning system to decide which of them are keyphrases. This is achieved in this paper by the DPM-index metric, which gives a way to control both the size and the frequency of an *n*-gram by preserving both its significance and its specificity.

Recent works in keyphrase extraction area focused mainly on using features based on external knowledge information or document structural information. Our general goal in this paper is to return to the fundamental knowledge

Corresponding author:
Saïd Abdeddaïm, LITIS, Université de Rouen, 76821 Mont-Saint-Aignan Cedex, France.
Email: said.abdeddaim@univ-rouen.fr

extraction question addressed by the pioneer works in the area, which is: how to extract keyphrases from any type of text document without *a priori* knowledge on keyphrases. For this purpose we developed a supervised learning system that uses 18 statistical features, among them DPM-index and five other new measures. After selecting keyphrase candidates using linguistic information (Part-of-Speech, POS, tags), the system offers the possibility to evaluate the efficiency of the contribution of each statistical feature.

We experimentally compared our results with those of 21 keyphrase extraction methods on SemEval-2010/Task-5 scientific articles corpus. When all the systems extract 10 keyphrases per document, our method increases by a rate of 13% the *F*-score of the best system. In particular, the DPM-index feature increases the accuracy of our keyphrase extraction system by a rate of 9%. We also show on this data that combining the 18 features using any supervised learning paradigm (boosting, bagging and regression) yields the best performance.

This paper is organized as follows. In Section 2 we give a synthetic state-of-the-art of keyphrase extraction methods based on the features they use to solve the problem. The preprocessing step of our extraction system is summarized in Section 3. The DPM-index measure and other features we use are described in Section 4. In Section 5 we present the utilized classifier. We analyse and compare our results in Section 6 and end with a conclusion.

2. Related works

Designing a keyphrase extraction system consists of selecting the properties that could distinguish keyphrases from other terms. These properties, called *features*, can be classified according to their origin: *term-level features, document-level features, corpus-level features* and *external knowledge-based features*. Keyphrase extraction methods are then traditionally classified according to the features they use into two categories: document-dependent and corpus-dependent. These two categories become four if we take into account that they use external knowledge sources or not.

Document-dependent approaches [11–17] are based on term- and document-level features. These methods are unsupervised and generally use the co-occurrence frequencies of the terms in the considered document. In addition to term and document features, *corpus dependent approaches* (the majority of cited references) generally use the number of corpus documents that contain a term as an indicator of its specificity. Terms that appear frequently in the document but rarely in the remainder of the corpus are more likely to be keyphrases. *External knowledge based approaches* use external knowledge sources such as terminological databases: GRISP, MeSH, etc. [5, 18–21], linguistic ressources (WordNet, etc. [22, 23]), article databanks (Wikipedia, etc. [7, 8, 14, 24–28]), search engine query logs [6, 8] and search engine results (rank scores and/or text-snippets from Google, Bing, MSN etc. [24, 29–32]). Terminological databases were used early by *text categorization* methods [33] that are essentially based on such information to assign *a priori* known keyphrases to a document.

Features can also be classified according to their nature: statistical, linguistic and structural features. The most used *statistical features* are term length, that is, the number of words it contains [34, 3], term frequency (TF) in the document [34, 3], inverse document frequency (IDF), which depends on the number of corpus documents that contain the term, TFIDF [35, 36], which combines TF and IDF, the first position in the document [3, 34–36] and the co-occurrence frequency of the term with other document terms [11, 12]. Part-of-speech tags and noun phrases chunks [37–39] are *linguistic features* that try to capture the linguistic properties of keyphrases. *Structural features* that are provided by HTML or XML documents (like occurrence in title, document header, hypertext link, etc.) also help keyphrase identification [6, 20, 40–43].

The selected features are combined 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. The way this score is computed depends on the method. Some use an unsupervised approach when others use supervised learning on the corpus. Some *unsupervised* approaches are simply based on a score formula that combines the feature values like the baseline TFIDF or other expressions [44–46]. Other approaches use more sophisticated unsupervised algorithms [1, 2, 11–17, 23, 26, 47–51], generally a clustering method or a graph-based ranking approach [52] inspired by the well-known PageRank algorithm [53]. *Supervised* learning is achieved with machine learning approaches like bagged C4.5 [3], naive Bayes [35, 36], neural networks [54], SVM [40, 55] maximum entropy [6] and so on.

3. Candidate term identification

Identifying the candidate terms of each document needs to preprocess the texts. The way this classical step is done can significantly affect the accuracy of the keyphrase extraction method. Thus it is important to give some details on how we have implemented it. The preprocessing stages are as follows:

- *Tokenization and POS-tagging.* The first step of this stage includes sentence boundary detection, tokenization and POS tagging. We use the Stanford POS tagger [56] for these purposes. For each sentence, we generate all possible *n*-grams (subsequences of up to *n* tokens) with $n \leq 4$. *n*-Grams should not contain POS tagged tokens as punctuation, marks, brackets, numbers and special characters.
- *Stemming.* Term variation can affect its frequency, which is an important indicator of whether it is a keyphrase or not. The solution consists of replacing each word by its stem. A stem is generally what remains when we remove a suffix from a word. There are different stemmers [57]; we use the iterated Lovins stemmer [58], which is more aggressive than other stemmers. Aggressive stemming gives better results for keyphrase extraction method [3].
- *Document indexing.* All positions of a stemmed term are conserved. As the occurrences of a stemmed term can appear in different forms (different *n*-grams), we decided that the *n*-gram of a term is the most frequent one in the document. The same problem arises for POS tags. We have considered two cases: if the majority of POS tags correspond to noun phrases (as defined next) we choose the most frequent noun phrase POS tag, otherwise a POS tag which is not a noun phrase is chosen. At the end of this stage the document is transformed to a set of terms, where each term *t* in a document *d* is defined as a tuple: (*n-gram*, *stem*, *tag*, *d*, {*positions*}).
- *Linguistic filtering.* According to linguistic knowledge, the noun phrases are most likely to be keyphrases. Once all the terms are discovered, we only keep those which are considered as noun phrases. Our observation of the real keyphrases in the used train data suggest a very large POS tag definition of noun phrases which satisfy the regular expression: (NN | NNS | NNP | NNPS | JJ | VBC | VBN | NN IN | NNS IN)* (NN | NNS | NNP | NNPS | VBG)
- *Corpus indexing.* To compute corpus dependent features we need to index all the documents of the train data. After corpus indexing, each document term becomes a tuple: (*n-gram*, *stem*, *tag*, *d*, {*positions*}, {*documents*}) where {*documents*} represents the set of document paths of the training data containing this term, that is, a term which has the same stem.

4. Feature selection

Recent works in keyphrase extraction area focused mainly on using features based on external knowledge information or document structural information. Our general goal in this paper is to return to the fundamental knowledge extraction question addressed by the pioneer works of Turney et al. [34] and Frank et al. [35, 36], which is to extract keyphrases from any type of text document using statistical features without *a priori* knowledge on keyphrases, the opposite of text categorization approaches [33].

Some features used in the literature have different definitions that have different impacts on keyphrase extraction accuracy. In order to define precisely each used feature we propose to use the notations given in Table 1.

4.1. DPM-index

State-of-the-art keyphrase extraction features do not deal with the fact that a candidate term may contain or be contained in another candidate. This relationship between *n*-grams can be expressed by a feature that characterizes real keyphrases among different overlapping subterms of the same *n*-gram. For example, the candidate 3-gram *multiple sequence alignment* contains two 2-grams, *multiple sequence* and *sequence alignment*. These three *n*-grams could have close TFIDF values; how does one choose the real keyphrases among them? Intuitively, *multiple sequence* is a wrong candidate, but *sequence alignment* or *multiple sequence alignment* or both could be in the main topics of the paper.

The comparison of the frequencies of these *n*-grams in the considered document can help a learning system to decide which of them are keyphrases. The term *multiple sequence* by itself is not a keyphrase because it is not used outside *multiple sequence alignment*. This means that the frequency of *multiple sequence* is equal to the frequency of *multiple sequence alignment*. So we can suppose that, if a term *t* is contained in a superterm *s* and f(*t, d*) = f(*s, d*), i.e. f(*s, d*)/f(*t, d*) = 1, then *t* cannot be a keyphrase.

Now deciding whether *t* = *sequence alignment* is a keyphrase or not will depend on its frequency compared with the frequency of *s* = *multiple sequence alignment*. Let us, for example, consider a document *d* where *multiple sequence alignment* occurs 10 times. If *sequence alignment* occurs 100 times (f(*s, d*)/f(*t, d*) = 0.1) it is more likely to be a keyphrase than if it occurs 11 times (f(*s, d*)/f(*t, d*) = 0.91). The more *sequence alignment* is used outside *multiple sequence alignment*, the more it is likely to be in the main topics of the considered document. It is unlikely that, when the frequency of *sequence alignment* is close to the frequency of *multiple sequence alignment*, it means that the document deals specifically with the question of *multiple sequence alignment*. Suppose now that we have three documents d_1, d_2 and d_3 where *t* = *sequence alignment* is contained in the superterms:

Table 1. Notations used in this 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 nth occurrence of t in d (in number of words preceding it)
$\mathrm{npos}_n(t, d)$	The nth 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 nth 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)	$\mathrm{TFIDF}(t,d,D) = \mathrm{TF}(t,d) \times \mathrm{IDF}(t,D)$

- s_1 = *multiple sequence alignment*;
- s_2 = *pairwise sequence alignment*;
- x = *w sequence alignment*, where w denotes any other word different from *multiple* and *pairwise*.

The frequencies of t and its superterms are given in the Table 2. One can observe that, the more the maximum of the ratio f(s, d)/f(t, d) is close to 1, the more t is unlikely to be a keyphrase by itself.

We define the document phrase maximality index of a term t as:

$$\text{DPM–index}(t,d) = 1 - \max_{s \in \sup(t,d)} \left(\frac{\mathrm{f}(s,d)}{\mathrm{f}(t,d)} \right) \tag{1}$$

As we defined it, DPM-index(t, d) tends to 1 when t is not included in a superterm that has a similar frequency in the document d. In contrast, it tends to 0 when t is almost always a part of a bigger term in d, which indicates that it is unlikely to be a pertinent keyphrase for this document. By means of the DPM-index measure, we can immediately see in the example of Table 2 that the document d_1 does not deal with the general topic of t = *sequence alignment*, which has a DPM-index of 0.09, but is specific to s_1 = *multiple sequence alignment*. The opposite arises in the document d_3 where its DPM-index is 0.91. The document d_2 represents an intermediate situation where the DPM-index of 0.45 indicates that *sequence alignment* is more likely to be in the main topics of d_2 than in those of d_1.

These examples about three potential overlapping noun phrase candidates *multiple sequence alignment, multiple sequence* and *sequence alignment* can arise in any extraction system where we have to choose between extending the size n of an n-gram and reducing its frequency, or at the opposite, to reduce n and augment its frequency. The DPM-index gives a way to control both the size and the frequency of an n-gram by preserving its significance and its specificity to the document d.

The DPM-index of a term in a document resembles at first sight the C-value of a term in a corpus [59] which is:

$$\text{C-value}(t,D) = \log|t| \left(\mathrm{f}(t,D) - \frac{\sum_{s \in \sup(t,D)} \mathrm{f}(s,D)}{|\sup(t,D)|} \right) \tag{2}$$

Table 2. The DPM-index of the term t = sequence alignment in three documents.

Document	$f(t,d)$	$f(s_1,d)$	$\frac{f(s_1,d)}{f(t,d)}$	$f(s_2,d)$	$\frac{f(s_2,d)}{f(t,d)}$	$f(x,d)$	$\frac{f(x,d)}{f(t,d)}$	DPM–index(t,d)
d_1	11	10	0.91	1	0.09	0	0	**0.09**
d_2	11	6	0.55	5	0.45	0	0	**0.45**
d_3	11	1	0.09	1	0.09	1 $(\forall x)$	0.09	**0.91**

The C-value is known to be an essential criterion for *terminology extraction*. It could be adapted to keyphrase extraction in a document by replacing the document frequency in the corpus by the frequency in the document d. We denote this adaptation by docC-value:

$$\text{docC-value}(t,d) = \log|t|\left(f(t,d) - \frac{\sum_{s\in \sup(t,d)} f(s,d)}{|\sup(t,d)|}\right) \tag{3}$$

The DPM-index differs completely from docC-value because it uses the maximum of the superterm frequencies, rather than their mean, as is the case in the docC-value. The mean is not efficient for keyphrase extraction as it does not point out that a term t is almost always contained in a particular superterm. For the example of Table 2, in both d_1 and d_2, the means of the frequencies of the superterms of t are the same and equal 5.5, when their maximum is 10 in d_1 and 6 in d_2. This leads to a DPM-index of 0.09 for d_1 and 0.45 for d_2, while their docC-values are identical. Thus deciding whether t is a keyphrase or not does not depend on the mean of the superterm frequencies but on their maximum. This was confirmed by our experiments; both C-value and docC-value do not improve keyphrase extraction accuracy.

4.2. Other proposed features

4.2.1. TFIDF ratio. When a keyphrase is an n-gram that contains more than one compound, it is frequent that one of them is a specific word to the document. In You et al. [46] this compound is called core word and is defined simply as a word with a frequency greater than a certain threshold. In our case we do not use a fixed threshold and, rather than using TF to characterize this compound, we use TFIDF. We define the feature TFIDFRatio(*t, d, D*) as the ratio between the TFIDF of a term t and the maximum value of the TFIDF of a compound of t (see the formula in Table 3). This indicator tends to be small when the term has a compound with high TFIDF.

4.2.2. Position mean and 2-means. 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 benefit from 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 using as features the mean of these positions and their 2-means. We recall that the k-means μ_m $(m = 1, \ldots, k)$ of the normalized positions $x_i = \text{npos}_i(t,d)$ $(i = 1, \ldots, n)$ of a term t in the document d are such that there exists a partition of these positions in k clusters E_m $(m = 1, \ldots, k)$ where $\sum_m \sum_{x_i \in E_m} (x_i - \mu_m)^2$ is minimal. In the general case, the k-means problem is often solved by a heuristic as it is known to be NP-hard in general, even when k = 2 or when the dimensionality is 2 [60]. However, in our case the n positions of a term are defined in one dimension for which there is an $O(kn^2)$ time complexity dynamic programming algorithm to guarantee optimality [60].

4.2.3. DPM-TFIDF. In our experiments we noticed that combining the DPM-index with TFIDF (see the formula in Table 3) improves the accuracy of this later when we use it as an unsupervised score for automatic keyphrase extraction.

4.3. Other used features

After trying approximately 30 features used in the literature, we retained 12 features that work well for keyphrase extraction in our experiments. Among these features four are rarely used in the literature:

(1) $\text{SFS}(t,d)$ – the substring frequencies sum [43];
(2) $\text{GDC}(t,d)$ – the Generalized Dice coefficient [61, 20], which is an adaptation of a widely used measure [62] in terminology extraction [63];

Table 3. Features used in our system.

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

(3) $\text{MLE}(t, d)$ – the maximum likelihood estimate [64];
(4) $\text{KLD}(t, d, D)$ – the Kullback–Leibler divergence [64].

We added the six 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 six features are numbered from 13 to 18.

5. Classification

Our keyphrase extraction system utilizes a classifier trained using a supervised machine learning algorithm. Owing to the difficulty of the keyphrase extraction problem, instead of using directly the classifier outputs, one rather utilizes the probability of being classified as a keyphrase [36]. 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 the learning algorithm. We also tested other learning algorithms, for instance bagged C4.5 decision trees, random forests and LogitBoost, but in every case logistic regression gave good results. We used the Weka implementation of these methods [65].

The accuracy of our system greatly depends on the way the 18 features are transformed into input attributes for the logistic regression algorithm. After we tested different data mining methods, we found that the best results are obtained after discretization and binarization of the input features. In order to explain this we need to recall some details on logistic regression.

A classifier is able to estimate the probability $P(Y = C|X)$ of an instance represented by its attribute vector X to belong to the class value $Y = C$. This estimation consists of training the learning algorithm on a set of instances (X_i, Y_i) where the class Y_i of each instance is known. After the training step, the obtained model can predict $P(Y = C|X)$ for any given X. In our case C is equal to 1 if the term is a keyphrase and 0 otherwise, the attribute vector X represents the features of a candidate term and $P(Y = 1|X)$ predicts the probability that a candidate term is a keyphrase.

Logistic regression tries to estimate $P(Y = 1|X)$ considering this assumption:

$$\log \frac{P(Y_i = 1|X_i)}{1 - P(Y_i = 1|X_i)} = \alpha_0 + \sum_{j=1}^{m} \alpha_j x_{ij} = z \tag{4}$$

where $X_\text{i} = (x_{i1}, x_{i2}, \ldots, x_{im})$ is the vector of m attribute values of the training instance i. This means that, for the logistic regression model, we have:

$$P(Y_i = 1|X_i) = \frac{1}{1 + e^{-z}} \quad (5)$$

The values of the parameters α_j $(j = 1, \ldots, m)$ that best fit with the training data are estimated by an optimization algorithm, a stochastic gradient descent algorithm for example. Given a term t_i, we are interested in estimating $p_i = P(Y_i = 1|X_i)$ the probability that t_i is a keyphrase. First, we consider that the features are given to the algorithm without preprocessing, which means that the $X_i = (x_{i1}, x_{i2}, \ldots, x_{im})$ are exactly the $m = 18$ feature values of t_i. In this case p_i depends on a linear product $\alpha_j x_{ij}$ of each feature value x_{ij} and its parameter α_j according to equation (4). However if we consider that the feature j is TFIDF for example, whatever α_j is, it is unlikely that the logit function (4) is linear in $x_{ij} = \text{TFIDF}(t_i, d, D)$.

In order to deal with the nonlinearity of feature effects on keyphrase probability we propose to discretize [66] each feature and binarize the intervals resulting from discretization. Precisely the logit function becomes:

$$\log \frac{P(Y_i = 1|X_i)}{1 - P(Y_i = 1|X_i)} = \alpha_0 + \sum_{j=1}^{m} \sum_{k=1}^{n_j} \alpha_{jk} I(x_{ij} \in \mathbf{I}_{jk}) \quad (6)$$

where $\mathbf{I}_{jk}$ is the kth interval resulting from the discretization of the feature j, n_j the number of those intervals and $I(B) = 1$ if B is true, 0 otherwise. By replacing $\alpha_j x_{ij}$ by $\sum_{j=1}^{m} \sum_{k=1}^{n_j} \alpha_{jk} I(x_{ij} \in \mathbf{I}_{jk})$ in the logit function we no longer suppose the linearity of feature effects on keyphrase probability. Each α_j is substituted by the parameters α_{jk} $(k = 1, \ldots, n_j)$ that are estimated by the logistic regression. Thus, the logit function of equation (6) depends on any function of each feature j that is approximated by α_{jk} in each interval $\mathbf{I}_{jk}$.

6. Experiments

6.1. Corpus

In order to evaluate our system, we used the SemEval-2010 data for task 5: Automatic Keyphrase Extraction from Scientific Articles [67]. These data consist of 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 Behavioral Sciences.

For each paper at most 15 keyphrases were manually assigned by both paper authors and readers. From this corpus 144 papers were provided for training and the evaluation was done on 100 articles. The main advantage of using SemEval-2010/Task-5 corpus is that we can compare our results with those obtained by 19 teams that participated in the challenge. Furthermore, two recent papers [51, 46] 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 [68]; however, exact match is stricter and enables us to compare our results with those of the 21 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 F_1-score, which is defined as $2 \times precision \times recall/(precision + recall)$.

6.2. Results

We used SemEval-2010/Task-5 data for two experiments. First we followed the procedure of the challenge in order to compare our results with those of others systems. In this procedure 144 documents are used for training and 100 documents for the evaluation. Second, we grouped the 244 documents and used them for a 5-fold cross-validation experiment in order to confirm the results.

6.2.1. Challenge procedure. The performances of each system are given over the numbers of keyphrase candidates: top 5, 10 and 15. Table 4 shows the performances of each system ranked by the F_1-Score over the top 15 keyphrases. Our system ranks first over the three keyphrase candidates and for the three metrics used. For 10 keyphrases, our system increases by a rate of 13% (29.4/26.0% − 1) the F_1-score of HUMB [20]. Notice that, opposite to our system, HUMB uses structural features and different external knowledge features in order to improve its performance. These knowledge

Table 4. Our system compared with 21 systems according to precision (*P*), recall (*R*) and F_1-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%
[46]	—	—	—	—	—	—	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%
SZTERGAK	34.2%	11.7%	17.4%	28.5%	19.4%	23.1%	24.8%	25.4%	25.1%
ICL	34.4%	11.7%	17.5%	29.2%	19.9	23.7%	24.6%	25.2%	24.9%
SEERLAB	39.0%	13.3%	19.8%	29.7%	20.3%	24.1%	24.1%	24.6%	24.3%
KX-FBK	34.2%	11.7%	17.4%	27.0%	18.4%	21.9%	23.6%	24.2%	23.9%
DERIUNLP	27.4%	9.4%	13.9%	23.0%	15.7%	18.7%	22.0%	22.5%	22.3%
Maui	35.0%	11.9%	17.8%	25.2%	17.2%	20.4%	20.3%	20.8%	20.6%
DFKI	29.2%	10.0%	14.9%	23.3%	15.9%	18.9%	20.3%	20.7%	20.5%
DP-Seg [51]	33.3%	10.5%	16.0%	23.7%	16.9%	19.7%	19.2%	21.3%	20.2%
BUAP	13.6%	4.6%	6.9%	17.6%	12.0%	14.3%	19.0%	19.4%	19.2%
SJTULTLAB	30.2%	10.3%	15.4%	22.7%	15.5%	18.4%	18.4%	18.8%	18.6%
UNICE	27.4%	9.4%	13.9%	22.4%	15.3%	18.2%	18.3%	18.8%	18.5%
UNPMC	18.0%	6.1%	9.2%	19.0%	13.0%	15.4%	18.1%	18.6%	18.3%
U_CSE	28.4%	9.7%	14.5%	21.5%	14.7%	17.4%	17.8%	18.2%	18.0%
LIKEY	29.2%	10.0%	14.9%	21.1%	14.4%	17.1%	16.3%	16.7%	16.5%
UvT	24.8%	8.5%	12.6%	18.6%	12.7%	15.1%	14.6%	14.9%	14.8%
POLYU	15.6%	5.3%	7.9%	14.6%	10.0%	11.8%	13.9%	14.2%	14.0%
UKP	9.4%	3.2%	4.8%	5.9%	4.0%	4.8%	5.3%	5.4%	5.3%

Table 5. Our system performances after removing a feature or a family of features compared with HUMB and WINGNUS.

Features	Top 5 candidates			Top 10 candidates			Top 15 candidates		
	P	*R*	*F*	*P*	*R*	*F*	*P*	*R*	*F*
All Features	44.8%	15.3%	22.8%	**36.2%**	**24.7%**	**29.4%**	**28.3%**	**28.9%**	**28.6%**
All – TFIDF ratio	43.2%	14.7%	22.0%	35.5%	24.2%	28.8%	27.7%	28.3%	28.0%
All – DPM-TFIDF[a]	**45.2%**	**15.4%**	**23.0%**	35.2%	24.0%	28.5%	27.7%	28.3%	28.0%
All – k-means	44.6%	14.2%	22.7%	35.9%	24.5%	29.1%	27.4%	28.0%	27.7%
HUMB	*39.0%*	*13.3%*	*19.8%*	*32.0%*	*21.8%*	*26.0%*	*27.2%*	*27.8%*	*27.5%*
All – DPM-index[b]	39.0%	13.3%	19.8%	33.2%	22.6%	26.9%	26.5%	27.1%	26.8%
All – Corpus level features[c]	43.2%	14.7%	22.0%	32.8%	22.4%	26.6%	26.0%	26.6%	26.3%
All – Our features	38.6%	13.2%	19.6%	32.0%	21.8%	25.9%	25.8%	26.4%	26.1%
WINGNUS	*40.2%*	*13.7%*	*20.5%*	*30.5%*	*20.8%*	*24.7%*	*24.9%*	*25.5%*	*25.2%*

[a]DPM-TFIDF is replaced by TFIDF.

[b]DPM-index is removed and DPM-TFIDF is replaced by TFIDF.

[c]IDF, logTFIDF, KLD, DPM-TFIDF and TFIDFRatio are removed.

bases (GROBID/TEI, GRISP and HAL) are specific to scientific papers. Then the most important feature of these results is that, by using only statistical features on linguistically filtered terms, our system outperforms the others without loss of generality.

In order to measure the contribution of each feature to the overall system performance, we represent in Table 5 the results obtained by our method when removing each feature from the learning attributes. According to these experiments, we can see that the DPM-index is the most important of our features; removing it significantly decreases the performance of our system. When DPM-index is used, the F_1-score of the keyphrase extraction system increases by 9% (for the 10 top keyphrases), which makes its contribution comparable to that of all corpus-level features including the popular TFIDF. Note also that using our six proposed features increases the F_1-score by a rate of 13.5% for the 10 top keyphrases.

Table 6. Our system performances with different learning methods compared with HUMB and WINGNUS

Learning method	Top 5 candidates			Top 10 candidates			Top 15 candidates		
	P	*R*	*F*	*P*	*R*	*F*	*P*	*R*	*F*
Logistic regression	**44.8%**	**15.3%**	**22.8%**	**36.2%**	**24.7%**	**29.4%**	**28.3%**	**28.9%**	**28.6%**
LogitBoost	43.2%	14.7%	22.0%	34.8%	23.7%	28.2%	27.7%	28.4%	28.0%
Bagged C4.5	40.8%	13.9%	20.8%	33.2%	22.6%	26.9%	27.2%	27.8%	27.5%
HUMB	*39.0%*	*13.3%*	*19.8%*	*32.0%*	*21.8%*	*26.0%*	*27.2%*	*27.8%*	*27.5%*
Random Forest	43.0%	14.7%	21.9%	32.6%	22.2%	26.4%	26.0%	26.6%	26.3%
WINGNUS	*40.2%*	*13.7%*	*20.5%*	*30.5%*	*20.8%*	*24.7%*	*24.9%*	*25.5%*	*25.2%*

Table 7. Five-fold cross-validation performances (mean and standard deviation) after removing a feature or a family of features

Features	Top 5 candidates			Top 10 candidates			Top 15 candidates		
	P	*R*	*F*	*P*	*R*	*F*	*P*	*R*	*F*
All features	46.7 ± 2.1%	15.5 ± 1.0%	23.3 ± 1.4%	35.9 ± 1.4%	23.8 ± 1.3%	28.6 ± 1.4%	28.5 ± 0.8%	28.4 ± 1.5%	28.4 ± 1.1%
All – TFIDF ratio	45.5 ± 3.1%	15.1 ± 1.4%	22.7 ± 1.9%	35.4 ± 1.3%	23.5 ± 1.4%	28.2 ± 1.4%	28.4 ± 0.9%	28.3 ± 1.5%	28.3 ± 1.2%
All – DPM-TFIDF	45.6 ± 2.1%	15.1 ± 1.0%	22.7 ± 1.4%	35.2 ± 1.1%	23.4 ± 1.3%	28.1 ± 1.3%	28.1 ± 0.7%	27.9 ± 1.2%	28.0 ± 0.9%
All – k-means	45.6 ± 1.8%	15.1 ± 1.0%	22.7 ± 1.3%	35.5 ± 1.5%	23.5 ± 1.5%	28.3 ± 1.5%	28.0 ± 1.3%	27.9 ± 1.9%	27.9 ± 1.6%
All – DPM-index	43.9 ± 2.3%	14.6 ± 1.1%	21.9 ± 1.5%	32.7 ± 0.5%	21.7 ± 0.8%	26.1 ± 0.7%	26.4 ± 0.6%	26.3 ± 1.1%	26.4 ± 0.8%
All – Corpus level feat.	44.3 ± 3.7%	14.7 ± 1.6%	22.1 ± 2.2%	33.3 ± 1.3%	22.1 ± 1.4%	26.6 ± 1.4%	26.5 ± 0.9%	26.4 ± 1.5%	26.4 ± 1.2%
All – Our features	41.6 ± 1.8%	13.8 ± 1.0%	20.7 ± 1.3%	31.3 ± 0.7%	20.8 ± 1.0%	25.0 ± 0.9%	25.6 ± 1.5%	25.5 ± 2.0%	25.6 ± 1.7%

Table 8. 5-fold cross-validation performances (mean and standard deviation) with different learning methods

Learning Method	Top 5 candidates			Top 10 candidates			Top 15 candidates		
	P	*R*	*F*	*P*	*R*	*F*	*P*	*R*	*F*
Logistic regression	46.7 ± 2.1%	15.5 ± 1.0%	23.3 ± 1.4%	35.9 ± 1.4%	23.8 ± 1.3%	28.6 ± 1.4%	28.5 ± 0.8%	28.4 ± 1.5%	28.4 ± 1.1%
LogitBoost	45.5 ± 1.8%	15.1 ± 1.0%	22.7 ± 1.3%	36.1 ± 1.7%	24.0 ± 1.7%	28.8 ± 1.8%	28.6 ± 1.2%	28.5 ± 1.9%	28.5 ± 1.5%
Bagged C4.5	44.2 ± 2.3%	14.7 ± 1.0%	22.0 ± 1.4%	35.7 ± 1.5%	23.7 ± 1.3%	28.4 ± 1.3%	27.9 ± 0.9%	27.7 ± 1.3%	27.8 ± 1.1%
Random Forest	46.6 ± 1.7%	15.5 ± 0.9%	23.2 ± 1.2%	35.2 ± 1.0%	23.3 ± 1.1%	28.0 ± 1.1%	27.4 ± 1.0%	27.3 ± 1.5%	27.3 ± 1.2%

We also tested our system using different machine learning algorithms. The goal was to find the best classifier. As one can observe in Table 6, combining the 18 features using any learning paradigm (boosting, bagging and regression) yields the best performance, confirming that our results do not depend on a specific learning approach.

6.2.2. Five-fold cross-validation. We used 5-fold cross-validation in order to confirm the stability and the robustness of the features. The 244 documents of the entire corpus are partitioned randomly into five sets of 48 documents (four documents remain). Testing is performed on each set and the remaining is used for training. The results are averaged over all runs. Table 7 presents the 5-fold cross-validation results obtained by our method when removing each feature from the learning attributes. These results confirm that DPM-index improves keyphrase extraction (by 9.6%). Our three other features impact less the F_1-score when they are used individually, but together with DPM-index they significantly improve the performance. Table 8 gives the results we obtained with different machine learning algorithms. In the 5-fold cross-validation experiment LogitBoost and logistic regression give similar results on average.

7. Conclusion

This paper presents a new measure the DPM-index that discriminates the overlapping *n*-grams in a document. We developed a new supervised learning keyphrase extraction system that combines 18 statistical features, among them the

DPM-index. We showed experimentally that this latter makes a contribution comparable to corpus-level features for the extraction of 10 keyphrases. The results of this system demonstrate an improvement over other systems without using any external knowledge or document structural features, which makes it more flexible and adaptable to other types of corpora.

Acknowledgements

Professor Aïcha Mokhtari (USTHB) and Professor Thierry Lecroq (Université de Rouen) are gratefully acknowledged for their help and support.

Funding

M.H. was a recipient of a PhD fellowship from the PROFAS French–Algerian cooperation programme.

References

[1] Zha H. Generic summarization and keyphrase extraction using mutual reinforcement principle and sentence clustering. In: *Proceedings of the 25th annual international ACM SIGIR conference on research and development in information retrieval, SIGIR 2002*, Tampere, Finland, 11–15 August 2002. New York: ACM, 2002, pp. 113–120.

[2] Liu F, Pennell D, Liu F and Liu Y. Unsupervised approaches for automatic keyword extraction using meeting transcripts. In: *Proceedings of the 2009 conference of the North American Chapter of the Association of Computational Linguistics, NAACL 2009*, Boulder, CO, 31 May to 5 June 2009. Stroudsburg, PA: ACL, 2009, pp. 620–628.

[3] Turney PD. Learning algorithms for keyphrase extraction. *Information Retrieval* 2000; 2(4): 303–336.

[4] Li Z, Zhou D, Juan YF and Han J. Keyword extraction for social snippets. In: *Proceedings of the 19th international conference on World Wide Web, WWW 2010*, Raleigh, NC, 26–30 April 2010. New York: ACM, 2010, pp. 1143–1144.

[5] Medelyan O and Witten IH. Measuring inter-indexer consistency using a thesaurus. In: *Proceedings of the 6th ACM/IEEE joint conference on digital libraries, JCDL 2006*, Chapel Hill, NC, 11–15 June 2006. New York: ACM, 2006, pp. 274–275.

[6] Yih W, Goodman J and Carvalho VR. Finding advertising keywords on web pages. In: *Proceedings of the 15th international conference on World Wide Web, WWW 2006*, Edinburgh, 23–26 May 2006. New York: ACM, 2006, pp. 213–222.

[7] Medelyan O, Frank E and Witten IH. Human-competitive tagging using automatic keyphrase extraction. In: *Proceedings of the 2009 conference on empirical methods in natural language processing, EMNLP 2009*, Singapore, 6–7 August 2009, Stroudsburg, PA: ACL, 2009, pp. 1318–1327.

[8] Qiu M, Li Y and Jiang J. Query-oriented keyphrase extraction. In: *Information retrieval technology – proceedings of the 18th Asia Information Retrieval Societies conference, AIRS 2012*, Tianjin, China, 17–19 December 2012. Lecture Notes in Computer Science, Vol. 7675. Berlin: Springer, 2012, pp. 64–75.

[9] Kang SS. Keyword-based document clustering. In: *Proceedings of the 6th international workshop on information retrieval with Asian languages*, Sappro, Japan, 7 July 2003. Stroudsburg, PA: ACL, 2003, pp. 132–137.

[10] Song M, Song IY, Allen RB and Obradovic Z. Keyphrase extraction-based query expansion in digital libraries. In: *Proceedings of the 6th ACM/IEEE joint conference on digital libraries, JCDL 2006*, Chapel Hill, NC, 11–15 June 2006. New York: ACM, 2006, pp. 202–209.

[11] Mihalcea R and Tarau P. TextRank: Bringing order into text. In: *Proceedings of the 2004 conference on empirical methods in natural language processing, EMNLP 2004*, Barcelona, 25–26 July 2004. Stroudsburg, PA: ACL, 2004, pp. 404–411.

[12] Matsuo Y and Ishizuka M. Keyword extraction from a single document using word co-occurrence statistical information. *International Journal on Artificial Intelligence Tools* 2004; 13(1): 157–169.

[13] Palshikar GK. Keyword extraction from a single document using centrality measures. In: *Proceedings of the 2nd international conference on pattern recognition and machine intelligence, PReMI 2007*, Kolkata, 18–22 December 2007. Lecture Notes in Computer Science, Vol. 4815. Berlin: Springer; 2007, pp. 503–510.

[14] Liu Z, Li P, Zheng Y and Sun M. Clustering to find exemplar terms for keyphrase extraction. In: *Proceedings of the 2009 conference on empirical methods in natural language processing, EMNLP 2009*, Singapore, 6–7 August 2009. Stroudsburg, PA: ACL, 2009, pp. 257–266.

[15] Rose S, Engel D, Cramer N and Cowley W. Automatic keyword extraction from individual documents. In: Berry MW and Kogan J (eds) *Text mining: Applications and theory*. Chichester: Wiley, 2010, pp. 1–20.

[16] Liu J, Wang C, Liu Z and Yao W. Advertising keywords extraction from web pages. In: *Proceedings of the 7th international conference on web information systems and mining, WISM 2010*, Sanya, China, 23–24 October 2010. Lecture Notes in Computer Science, Vol. 6318. Berlin: Springer; 2010, pp. 336–343.

[17] Wei Y. An iterative approach to keywords extraction. In: *Proceedings of the 3rd international conference, on advances in swarm intelligence, part II, ICSI 2012*, Shenzhen, China, 17–20 June 2012. Lecture Notes in Computer Science, Vol. 7332. Berlin: Springer, 2012, pp. 93–99.

[18] Hulth A, Karlgren J, Jonsson A, Boström H and Asker L. Automatic keyword extraction using domain knowledge. In: *Proceedings of the 2nd international conference on computational linguistics and intelligent text processing, CICLing 2001*, Mexico-City, 18–24 February 2001. Lecture Notes in Computer Science, Vol. 2004. Berlin: Springer, 2001, pp. 472–482.

[19] Gazendam L, Wartena C and Brussee R. Thesaurus based term ranking for keyword extraction. In: *Database and expert systems applications, international workshops, DEXA 2010*, Bilbao, Spain, 30 August to 3 September 2010. New York: IEEE, 2010, pp. 49–53.

[20] Lopez P and Romary L. HUMB: Automatic key term extraction from scientific articles in GROBID. In: *Proceedings of the 5th international workshop on semantic evaluation, SemEval 2010*, Uppsala, Sweden, 15–16 July 2010. Stroudsburg, PA: ACL, 2010, pp. 248–251.

[21] Bong SY and Hwang KB. Keyphrase extraction in biomedical publications using mesh and intraphrase word co-occurrence information. In: *Proceedings of the ACM 15th international workshop on Data and text mining in biomedical informatics, DTMBio 2011*, New York, 24–28 October 2011. New York: ACM, 2011, pp. 63–66.

[22] Ercan G and Cicekli I. Using lexical chains for keyword extraction. *Information Processing and Management* 2007; 43(6): 1705–1714.

[23] Wang J, Liu J and Wang C. Keyword extraction based on PageRank. In: *Proceedings of the 11th Pacific–Asia conference on advances in knowledge discovery and data mining, PAKDD 2007*, Nanjing, China, 22–25 May 2007. Lecture Notes in Computer Science, Vol. 4426. Berlin: Springer, 2007, pp. 857–864.

[24] Eichler K and Neumann G. DFKI KeyWE: Ranking keyphrases extracted from scientific articles. In: *Proceedings of the 5th international workshop on semantic evaluation, SemEval 2010*, Uppsala, Sweden, 15–16 July 2010. Stroudsburg, PA: ACL, 2010, pp. 150–153.

[25] Berend G and Farkas R. SZTERGAK: Feature engineering for keyphrase extraction. In: *Proceedings of the 5th international workshop on semantic evaluation, SemEval 2010*, Uppsala, Sweden, 15–16 July 2010. Stroudsburg, PA: ACL, 2010, pp. 186–189.

[26] Zhang W, Wang D, Xue GR and Zha H. Advertising keywords recommendation for short-text web pages using Wikipedia. *ACM Transactions on Intelligent Systems and Technology* 2012; 3(2): 36.

[27] Ercan G and Cicekli I. Keyphrase extraction through query performance prediction. *Journal of Information Science* 2012; 38(5): 476–488.

[28] Joorabchi A and Mahdi AE. Automatic keyphrase annotation of scientific documents using Wikipedia and genetic algorithms. *Journal of Information Science* 2013; 39(3): 410–426.

[29] Turney PD. Coherent keyphrase extraction via web mining. In: *Proceedings of the 18h international joint conference on artificial intelligence, IJCAI 2003*, Acapulco, Mexico, 9–15 August 2003. San Francisco, CA: Morgan Kaufmann, 2003, pp. 434–442.

[30] Mori J, Matsuo Y and Ishizuka M. Finding user semantics on the Web using word co-occurrence information. In: *Proceedings of the international workshop on personalization on the Semantic Web, PerSWeb 2005*, Edinburgh, 24–30 July 2005, posters, pp. 77–86.

[31] Joshi A and Motwani R. Keyword generation for search engine advertising. In: *Workshops proceedings of the 6th IEEE international conference on data mining, ICDM 2006 workshops*, Hong Kong, China, 18–22 December 2006. New York: IEEE Computer Society, 2006, pp. 490–496.

[32] Chen DY, Li X, Liu J and Chen X. Ranking-constrained keyword sequence extraction from web documents. In: *Proceedings of the 20th Australasian database conference, ADC 2009*, Wellington, New Zealand, 20–23 January, 2009. CRPIT 2009, Vol. 92. Australia: ACS, 2009, pp. 161–169.

[33] Aggarwal CC and Zhai C. A survey of text classification algorithms. In: Aggarwal CC and Zhai C (eds) *Mining text data*. Berlin: Springer, 2012, pp. 163–222.

[34] Turney PD. *Learning to extract keyphrases from text*. National Research Council, Institute for Information Technology, 1999, ERB-1057.

[35] Frank E, Paynter GW, Witten IH, Gutwin C and Nevill-Manning CG. Domain-specific keyphrase extraction. In: *Proceedings of the 16th international joint conference on artificial intelligence, IJCAI 1999*, Stockholm, 31 July to 6 August 1999. San Francisco, CA: Morgan Kaufmann; 1999, pp. 668–673.

[36] Witten IH, Paynter GW, Frank E, Gutwin C and Nevill-Manning CG. KEA: Practical automatic keyphrase extraction. In: *Proceedings of the 4th ACM conference on digital libraries, DL 1999*, Berkeley, CA, 11–14 August 1999. New York: ACM, 1999, pp. 254–255.

[37] Hulth A. Improved automatic keyword extraction given more linguistic knowledge. In: *Proceedings of the 2003 conference on empirical methods in natural language processing, EMNLP 2003*, Sapporo, Japan, 11–12 July 2003. Stroudsburg, PA: ACL, 2003, pp. 216–223.

[38] Krapivin M, Marchese M, Yadrantsau A and Liang Y. Unsupervised key-phrases extraction from scientific papers using domain and linguistic knowledge. In: *Proceedings of the 3rd international conference on digital information management, ICDIM 2008*, London, 13–16 November 2008. New York: IEEE, 2008, pp. 105–112.

[39] Krapivin M, Autayeu A, Marchese M, Blanzieri E and Segata N. Keyphrases extraction from scientific documents: improving machine learning approaches with natural language processing. In: *Proceedings of the 12th international conference on Asia–Pacific digital libraries, ICADL 2010*, Gold Coast, Australia, 21–25 June 2010. Berlin: Springer, 2010, pp. 102–111.
[40] Wang J and Peng H. Keyphrases extraction from web document by the least squares support vector machine. In: *Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence, WI 2005*, Compiegne, France, 19–22 September 2005. New York: IEEE, 2005, pp. 293–296.
[41] Nguyen TD and Kan MY. Keyphrase extraction in scientific publications. In: *Proceedings of the 10th international conference on Asian digital libraries, ICADL 2007*, Hanoi, Vietnam, 10–13 December 2007. Lecture Notes in Computer Science, Vol. 4822. Berlin: Springer, 2007, pp. 317–326.
[42] Hofmann K, Tsagkias M, Meij E and de Rijke M. The impact of document structure on keyphrase extraction. In: *Proceedings of the 18th ACM conference on information and knowledge management, CIKM 2009*, Hong Kong, China, November 2009. New York: ACM, 2009, pp. 1725–1728.
[43] Nguyen TD and Luong MT. WINGNUS: Keyphrase extraction utilizing document logical structure. In: *Proceedings of the 5th international workshop on semantic evaluation, SemEval 2010*. Uppsala, Sweden: Stroudsburg, PA: ACL, 2010, pp. 166–169.
[44] Kim SN, Baldwin T and Kan MY. An unsupervised approach to domain-specific term extraction. In: *Proceedings of the Australasian language technology association workshop, ALTA 2009*, Sydney, 3–4 December 2009. Stroudsburg, PA: ACL, 2009, pp. 94–98.
[45] El-Beltagy SR and Rafea A. KP-Miner: Participation in SemEval-2. In: *Proceedings of the 5th international workshop on semantic evaluation, SemEval 2010*, Uppsala, Sweden, 15–16 July 2010. Stroudsburg, PA: ACL, 2010, pp. 190–193.
[46] You W, Fontaine D and Barthès JPA. An automatic keyphrase extraction system for scientific documents. *Knowledge and Information Systems* 2013; 34(3): 691–724.
[47] Tomokiyo T and Hurst M. A language model approach to keyphrase extraction. In: *Proceedings of the ACL 2003 workshop on multiword expressions, MWE 2003*, Sapporo, Japan, 7–12 July 2003. Stroudsburg, PA: ACL, 2003, pp. 33–40.
[48] Wan X and Xiao J. CollabRank: Towards a collaborative approach to single-document keyphrase extraction. In: *Proceedings of the 22nd international conference on computational linguistics, COLING 2008*, Manchester, 18–22 August 2008, pp. 969–976.
[49] Wan X and Xiao J. Single document keyphrase extraction using neighborhood knowledge. In: *Proceedings of the 23rd AAAI conference on artificial intelligence, AAAI 2008*, Chicago, IL, 13–17 July 2008. Cambridge, MA: AAAI Press, 2008, pp. 855–860.
[50] Wan X and Xiao J. Exploiting neighborhood knowledge for single document summarization and keyphrase extraction. *ACM Transactions on Information and System Security* 2010; 28(2): 1–34.
[51] Newman D, Koilada N, Lau JH and Baldwin T. Bayesian text segmentation for index term identification and keyphrase extraction. In: *Proceedings of the 24th international conference on computational linguistics, COLING 2012*, Mumbai, 8–15 December 2012, pp. 2077–2092.
[52] Hasan KS and Ng V. Conundrums in unsupervised keyphrase extraction: Making sense of the state-of-the-art. In: *Proceedings of the 23rd international conference on computational linguistics, COLING 2010*, Beijing, 23–27 August 2010, Posters, pp. 365–373.
[53] Brin S and Page L. The anatomy of a large-scale hypertextual web search engine. *Computer Networks* 1998; 30(1–7): 107–117.
[54] Jo T. Neural based approach to keyword extraction from documents. In: *Proceedings of the 2003 international conference on computational science and its applications, part I, ICCSA 2003*, Montreal, 18–21 May 2003. Lecture Notes in Computer Science, Vol. 2667. Berlin: Springer, 2003, pp. 456–461.
[55] Jiang X, Hu Y and Li H. A ranking approach to keyphrase extraction. In: *Proceedings of the 32nd annual international ACM SIGIR conference on research and development in information retrieval, SIGIR 2009*, Boston, MA, 19–23 July 2009. New York: ACM, 2009, pp. 756–757.
[56] Toutanova K, Klein D, Manning CD and Singer Y. Feature-rich part-of-speech tagging with a cyclic dependency network. In: *Proceedings of the 2003 conference of the North American Chapter of the Association for Computational Linguistics on human language technology, HLT-NAACL 2003,* Vol. 1, Edmonton, 27 May to 1 June 2003, pp. 173–180.
[57] Smirnov I. Overview of stemming algorithms, unpublished, 2008, http://the-smirnovs.org/info/stemming.pdf
[58] Lovins JB. Development of a stemming algorithm. *Mechanical Translation and Computational Linguistics* 1968; 11: 22–31.
[59] Frantzi K, Ananiadou S and Mima H. Automatic recognition of multi-word terms: The C-value/NC-value method. *International Journal on Digital Libraries* 2000; 3(2): 115–130.
[60] Wang H and Song M. Ckmeans.1d.dp: Optimal *k*-means clustering in one dimension by dynamic programming. *The R Journal* 2011; 3: 29–33.
[61] Kim SN and Kan MY. Re-examining automatic keyphrase extraction approaches in scientific articles. In: *Proceedings of the ACL 2009 workshop on multiword expressions, MWE 2009*, Singapore, 6 August 2009. Stroudsburg, PA: ACL, 2009, pp. 9–16.
[62] Dice LR. Measures of the amount of ecologic association between species. *Ecology* 1945; 26(3): 297–302.
[63] Park Y, Byrd RJ and Boguraev BK. Automatic glossary extraction: beyond terminology identification. In: *Proceedings of the 19th international conference on computational linguistics, COLING 2002, Vol. 1*, Taipei, Taiwan, 24 August–September 2002, pp. 1–7.

[64] Hofmann K, Tsagkias M, Meij E and de Rijke M. A Comparative study of features for keyphrase extraction in scientific literature, 2009, http://edgar.meij.pro/comparative-study-features-keyphrase-extraction/

[65] Hall M, Frank E, Holmes G, Pfahringer B, Reutemann P and Witten IH. The WEKA data mining software: an update. *SIGKDD Explorations* 2009; 11(1): 10–18.

[66] Fayyad UM and Irani KB. Multi-interval discretization of continuous-valued attributes for classification learning. In: *Proceedings of the 13th international joint conference on artificial intelligence, IJCAI 1993*, Chambéry, France, 28 August to 3 September 1993. San Francisco, CA: Morgan Kaufmann, 1993, pp. 1022–1029.

[67] Kim SN, Medelyan O, Kan MY and Baldwin T. SemEval-2010 Task 5: Automatic keyphrase extraction from scientific articles. In: *Proceedings of the 5th international workshop on semantic evaluation, SemEval 2010*, Uppsala, Sweden, 15–16 July 2010. Stroudsburg, PA: ACL, 2010, pp. 21–26.

[68] Kim SN, Baldwin T and Kan MY. Evaluating N-gram based evaluation metrics for automatic keyphrase extraction. In: *Proceedings of the 23rd international conference on computational linguistics, COLING 2010*, Beijing, 23–27 August 2010, pp. 572–580.