A Comparative Study of Features for Keyphrase Extraction in Scientific Literature

Katja Hofmann, Manos Tsagkias, Edgar Meij and Maarten de Rijke
ISLA, University of Amsterdam
{K.Hofmann, E.Tsagkias, Edgar.Meij}@uva.nl, mdr@science.uva.nl

ABSTRACT

Keyphrases are short phrases that reflect the main topic of a document. Because manually annotating documents with keyphrases is a time-consuming process, several automatic approaches have been developed. Typically, candidate phrases are extracted using features such as position or frequency in the document text. Many different features have been suggested, and have been used individually or in combination. However, it is not clear which of these features are most informative for this task.

We address this issue in the context of keyphrase extraction from scientific literature. We introduce a new corpus that consists of full-text journal articles and is substantially larger than data sets used in previous work. In addition, the rich collection and document structure available at the publishing stage is explicitly annotated. We suggest new features based on this structure and compare them to existing features, analyzing how the different features capture different aspects the keyphrase extraction task.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: H.3.1 Content Analysis and Indexing; H.3.7 Digital Libraries; I.2 [**Artificial Intelligence**]: I.2.7 Natural Language Processing

General Terms

Algorithms, Experimentation

Keywords

Keyphrase extraction, Scientific literature search

1. INTRODUCTION

Keyphrases are short phrases that indicate the main topic of a document [10]. Initially, curator-assigned keyphrases were used to facilitate information access [14, 21]. Today, keyphrases are particularly important for exploratory search—to quickly get an overview of the contents of a collection, and for discovering information objects in the case of under-specified information needs. In the context of scientific literature search, keyphrases appear to be one of the clues that researchers use to make relevance decisions, and have been found beneficial for exploring digital libraries [1, 11]. Methods similar to keyphrase extraction have been applied to detect key concepts, for example to improve the precision of retrieval with long search queries [2].

As manual assignment of keyphrases is a tedious process, methods that automatically suggest keyphrases have been developed [13, 27, 30]. Keyphrase assignment to documents can be closed or open. The former draws keyphrases from a controlled vocabulary and assigned keyphrases do not necessarily occur in the document. In this paper, we focus on *open keyphrase extraction* where any phrase contained in the document is a potential candidate. Common approaches include selecting phrases on the basis of features such as the position or frequency of occurrence within the document. Such features can be combined, e.g., using classification or regression.

Although a number of features have been used for keyphrase extraction in the past, we know little about what makes a good keyphrase. Different studies make different (implicit) assumptions on what makes a good keyphrase, and it is difficult to objectively compare approaches based on different assumptions. For example, Shah et al. [23] assume that words representative of a document co-occur frequently with many other words, while Tomokiyo and Hurst [24] assume that keyphrases are those phrases that best distinguish a document from a background corpus. Features reflecting these intuitions have been developed, but not compared.

We address this problem by systematically evaluating and comparing features for keyphrase extraction on a new document collection that will be released with the publication of this paper, along with the features we have extracted. The collection consists of scientific journal articles with author-annotated keyphrases and is substantially larger than the collections previously considered.

Besides size, our document collection differs from previous ones in that it preserves the clean collection and document structure available at the publishing stage, which we utilize to derive new features that capture information contained in this structure. We explore whether this structural information captures additional characteristics of keyphrases and can be used to improve keyphrase extraction. Our models of these structural features are formulated in a probabilistic way which allows natural interpretations of results.

We follow a two-step approach to keyphrase extraction: (i) extract candidate phrases from the document text and (ii) rank the candidate list according to features assumed to reflect the phrases' likelihood of being assigned as a keyphrase. We consistently view step two as a ranking problem and discuss implications of this view.

To summarize, we address the following research questions:

- How do existing features and feature combinations perform on the new, substantially larger data collection?
- How do our models based on *collection structure* compare to existing features and can they be used to improve keyphrase extraction performance?
- How do our models based on *document structure* compare to

existing features and can they be used to improve keyphrase extraction performance?

The remainder of this paper is organized as follows. In Section 2 we give an overview of existing keyphrase extraction approaches, focusing on the features used for this task. We describe our document collection in Section 3, detail our approach in Section 4 and discuss the features we use in Section 5. We present and analyze our results in Section 6 and end with a concluding section.

2. RELATED WORK

Various approaches to keyphrase extraction have been explored in the past, which can be divided into unsupervised methods and methods that apply supervised learning. Unsupervised methods filter or rank candidate phrases according to a scoring function. Approaches using supervised learning train a machine learning algorithm to predict whether a phrase is a keyphrase or not. Both types of approaches can be viewed and evaluated within our view of keyphrase extraction as a ranking problem. Below, we introduce other approaches and describe the features and data sets used. We do not consider methods that use external knowledge sources such as thesauri or the web [8, 17, 27].

2.1 Supervised Methods

Turney [26] defines automatic keyphrase extraction as "*the automatic selection of important, topical phrases from within the body of a document*." He formulates the task as a supervised learning problem [25, 26]. Based on labeled example phrases, a machine learning algorithm learns a function that classifies phrases as positive (i.e., is a keyphrase) or negative (is not a keyphrase) examples. The author uses a set of 9 features related to position, frequency, and syntactical information. Performance is evaluated on 5 manually annotated corpora including journal articles, web pages, and email, and varying sizes of 20–141 documents.

Witten et al. [30] develop the *Kea* keyphrasing system that uses a Naive Bayes (NB) classifier. The authors find two features to perform well on this task: a combined $TF.IDF$ score, and the normalized position of the first occurrence of a phrase within the document. The system is evaluated on two of the corpora used by Turney [26]. Despite the reduced size of the feature set, performance is found to be statistically equivalent to that of Turney [26]. We compare our results with a recent version of *Kea* in Section 6.

Hulth [13] evaluates the contribution of linguistic features and compares three ways of extracting phrases (n-grams, part-of-speech (PoS) patterns, and noun phrases), and considers different features (TF, IDF, position of first occurrence, PoS sequence). The document collection used for training and evaluation (2000 scientific abstracts) appears to be the largest used for this task to date.

Few approaches have made use of structural information. Wang and Peng [28] introduce markup features to extract keyphrases from web pages. In addition to TF and IDF, paragraph frequency (the number of paragraphs a phrase occurs in) and title frequency (the number of times a phrase occurs in the page title or headings) are obtained from the HTML markup. Nguyen and Kan [19] use similar features for keyphrase extraction from scientific publications. They use baseline features used previously [30] and domain-specific morphological features. They also use two structural features: *in title* and a *section occurrence vector*, which consists of the term count for 14 generic sections, such as *abstract* and *introduction*. The complete feature set leads to significant improvements in classifier accuracy over the *Kea* baseline.

Below (cf. Section 4), we consider and evaluate all of the features discussed so far, except for morphological features. Based on the results obtained by Hulth [13], we believe that such this information is already captured by our candidate extraction approach which will be detailed in Section 4.

2.2 Unsupervised Methods

Unsupervised methods for keyphrase extraction have been developed analytically or based on the analysis of sample data. Typically, candidate words or phrases are extracted from a document using clues such as PoS [16, 23], punctuation marks, stop words, term counts, or combinations of these criteria [7, 24]. The candidates are assigned a score, ranked by this score, and the top-ranked phrases are selected as keyphrases.

KP-Miner assigns scores to phrases based $TFIDF$, position of first occurrence, and a boosting term that increases the chance of longer phrases being selected (as TF is found to otherwise favor single terms) [7]. The system is evaluated on the collection used by Turney [26] and significantly improves upon the performance of *Kea*, which is remarkable given that the system uses a linear combination of essentially the same features.

Several graph-based approaches have been proposed based on the assumption that good keyphrases are more "salient", or more "central" to a document. These intuitions are modeled using word co-occurrences in sentences [16, 23, 31]. Matsuo and Ishizuka [16] use co-occurrences of frequent phrases. Sentences within a document are clustered and phrases are extracted from these sentences. Candidate phrases are ranked using the χ^2 test of independence between expected and observed co-occurrences of frequent phrases. The approach is evaluated on 20 documents and performance is comparable to $TFIDF$.

Shah et al. [23] build on a similar intuition to extract (single term) keywords from scientific journal articles. Per document section, keywords are extracted by calculating the centrality of each noun, which is defined by the proportion of a word co-occuring with other words in the same sentence, following [20]. The centrality score is normalized per document, and terms that exceed a threshold are taken to be keywords. The authors also analyze the distribution of keywords in different document sections; while abstracts typically have the highest density of keywords, the remainder of an article contains a much larger number of keywords; the kinds of keywords that can be extracted differ between sections.

Tomokiyo and Hurst [24] propose a language modeling framework for keyphrase extraction. The "informativeness" of a phrase for a document is determined by the KL-Divergence between a foreground model (the document) and a background model (the collection). The "phraseness" of a sequence of words is determined by the KL-Divergence between a unigram and an n-gram model. A linear combination of these two scores is suggested and supported through a qualitative evaluation of news articles. The approach has not yet been evaluated in the context of keyphrase extraction, but on the related task of back-of-the-book index generation good result were achieved using KL-Divergence, the χ^2 test, and $TFIDF$ [5].

Below, we use some of the features derived from the unsupervised methods listed above. We include *keyphrase length*, based on the observation that single terms may otherwise be preferred [7]. We include one co-occurrence measure and choose the centrality score from [23], adjusted for phrases. We implement KL-Divergence corresponding to the measure of informativeness in [24] and χ^2 as proposed in [5], and also explore other probabilistic measures. We do not consider phraseness scores, as this aspect of candidate extraction is covered in our candidate extraction step.

Finally, keyphrase extraction is loosely related to automatic hypertext link generation, specifically to the identification of candidate anchor texts. Recent work explores features such as *link probability*, *relatedness*, *disambiguation confidence*, *generality*, and *location and spread* [18]; while many of these are specific to link

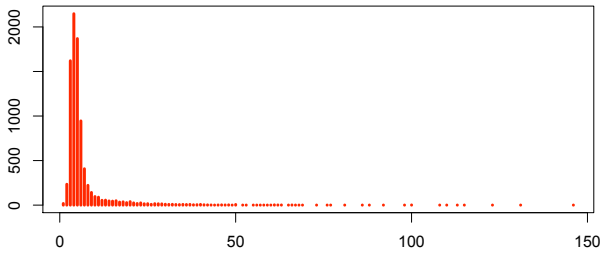

Figure 1: Number of documents vs. number of keyphrases. Most documents have between 3 and 6 keyphrases.

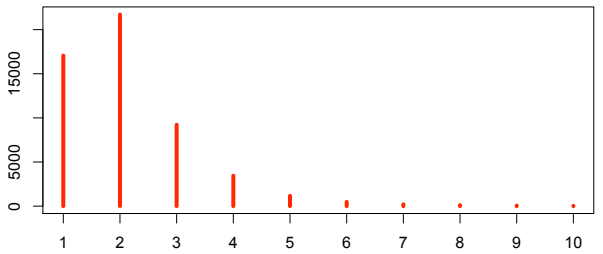

Figure 2: Number of keyphrases vs. keyphrase length. Typical keyphrases consist of 1 to 3 words.

generation, the latter two make sense for keyphrase extraction and we also use them.

3. DOCUMENT COLLECTION

We run our experiments on a collection of scientific journal articles provided by Elsevier.[1] The collection consists of 14,724 articles from 26 journals in the Food Informatics and Computer Science domains published between 1995 and 2005. This collection will be made available to the research community with publication of this paper.[2] The rich collection and document structure available at the publishing level is preserved in this data set. *Collection structure* refers to metadata that places documents in the context in which they are published, e.g., there are natural groupings by journal and issue, and an ordering by publication date. *Document structure* is provided in the form of XML markup according to a publicly available DTD.[3] Document annotations serve different functions, e.g., there are elements indicating article abstracts, individual sections, lists and list items, italicized terms, individual elements of mathematical formulas, references and citations, and many more. Overall, there are more than 100 XML codes, although not each document is marked up with all elements.

The collection spans a time-frame during which an increasing number of scientific journals started to make articles available online and this development is reflected in the collection. A growing part of articles was being digitized, starting, e.g., with publication records only, then including abstracts, and more recently full-text documents. Of the documents, 81% are available with abstracts, 77% include references, and 48% are available as full text.

The documents' authors have annotated 8,479 (58%) of the documents with between 1 and 146 keyphrases (mean: 6.33, mode: 4, see Figure 1). More than 75% of the documents with keyphrases have between 3 and 6 keyphrases. Keyphrases were are between 1 and 142 terms long. Very long keyphrases result from a small number of outliers where definitions are included as keyphrases. We ignore such outliers, automatically removing keyphrases of length greater than 10 terms. The remaining keyphrases have an average of 2.13 terms (Figure 2). In the documents we analyzed there are 53,651 keyphrases; of these, 38,222 (71.24%) also occur in the document content and can (theoretically) be extracted.

For our experiments we use a subset of the collection, consisting of the 5,504 documents for which both the document's full text and manually annotated keyphrases are available. We make this selection, as opposed to also including articles for which only keyphrases and abstract are provided, to avoid any possible bias.

4. APPROACH

Keyphrase extraction typically follows a two-step approach. First, candidate phrases are selected from the document text (1), then the best candidates are selected (2). Our main focus is on the second step. We view the task of identifying keyphrases from a set of candidate phrases as a problem of ranking candidate phrases according to their probability of being selected as a keyphrase. This probability can be expressed as $p(t = K|D)$, where K denotes the event that a phrase t drawn from document D is assigned as a keyphrase to this document (cf. Table 3 for an overview of our notation).

We explore different ways of estimation, where we call individual scores for estimation *features*. We detail the groups of features used in our work in Section 5 below. Before this, we describe our preprocessing steps (Section 4.1), our method of selecting candidate phrases from documents (Section 4.2), the method for combining multiple features into one ranking (Section 4.3), and evaluation measures (Section 4.4).

4.1 Experimental Setup and Preprocessing

As we will experiment with both ranking keyphrases using individual features and combinations of features using supervised learning, we divide the collection into development, training and test set to avoid overestimating the performance of learned combinations. The evaluation and comparison of individual features can be seen as manual feature selection. As is common in automatic feature selection we perform this step on the training set [9].

In our experimental setup we model the scenario where we have a number of existing journal issues available for analysis and training, and want to predict keyphrases for a set of unseen documents published in subsequent issues. Therefore we split the collection taking publication date and journal issue into account.

	size	number of documents
Development	20 %	1120
Training	60 %	3277
Test	20 %	1107

Table 1: Overview of development, training and test set used in our experiments. The collection was split by journal and issue as close as possible to the target percentage.

We take the first 20% of the issues per journal as *development* set, the next 60% as *training* set, and the last 20% as the *test* set (Table 1). We use a relatively large training set because our main focus is the comparison of a large set of individual features and we want to avoid spurious effects that may result from sparse data.

All documents in the collection are pre-processed and indexed identically, as follows. Extraction of phrase positions and frequencies was implemented on top of Lucene.[4] The XML documents were parsed and the textual content of each XML element was in-

[1] http://www.elsevier.com
[2] URL removed for anonymity.
[3] http://www.sciencedirect.info/techsupport/xmlsgml/dtd45/art452.dtd.txt
[4] http://lucene.apache.org/

dexed as a separate field (with stemming, but without stopword removal). In this way elements can be searched, and we can also retrieve the full text content of a specified element. We use "the full text of a document" to refer to the combined content of a document's title, abstract and section fields; otherwise, we refer to individual elements by name.

We apply sentence-splitting and PoS-tagging using an off-the-shelf software package.[5] Stemming is performed using the implementation of the snowball stemmer included with Lucene.

4.2 Candidate Selection

The first step in keyphrase extraction is to select candidate phrases from the document text. We compare existing methods in order to identify a method that achieves a good trade-off between precision and recall. Ideally, we want to obtain a large set of candidate phrases to maximize recall. However, a large number of candidate phrases results in a large processing overhead, as features for all candidates have to be calculated.

To select candidate phrases for a given document we take the full text content of the document, pre-processed as described above (see Section 4.1), and apply one of the following strategies:

- All n-grams: for each sentence, we generate all possible subsequences of up to n words.
- Filtered n-grams: we generate all n-grams, as above, and accept those that follow certain PoS patterns [11].
- PoS patterns: we extract all PoS patterns that occur as keyphrases in the first 10% of the corpus and use this list of patterns to filter the n-grams generated from each sentence [13].

We tune and evaluate candidate selection strategies using our development set. We split the development set in half, and tune the PoS patterns on the first half, corresponding to the first 10% of the collection. The approaches are then tested on the second half, corresponding to the second 10% of the collection (Table 2).

Method	candidates	correct	recall	prec.
All 3-grams	4,735,793	2,208	0.7449	0.0006
All 5-grams	9,882,514	2,284	0.7705	0.0003
All 10-grams	21,196,347	2,293	0.7732	0.0001
Filtered 3-grams*	1,437,186	2,186	0.7385	0.0018
Filtered 5-grams	2,825,797	2,256	0.7622	0.0010
Filtered 10-grams	5,896,621	2,265	0.7649	0.0005
PoS patterns	1,206,078	2,166	0.7296	0.0021

Table 2: Performance of different candidate selection methods on the development set. The method used in subsequent experiments is marked with *.

As expected, we achieve the highest recall using all n-grams but at very low precision. The highest recall that can be achieved on the data set is 77.32%: the missing keyphrases are not contained in the document text and simply cannot be assigned using an extraction-based approach. These typically include morphological variations that are not collapsed through stemming, or words that are broad descriptions of a document topic, too broad to occur in running text and more comparable to generic categories.

In comparison with previous work, our recall score after candidate selection is slightly lower [11, 13] and precision is substantially lower [13]. Our low precision stems from the fact that we select candidate phrases from full texts, not just abstracts. *Filtered 3-grams* is the candidate selection method of choice as it combines high recall and a reasonable number of candidate phrases. .

[5] http://alias-i.com/lingpipe

After applying candidate selection, we obtain a list of candidate phrases per document. The next step is to extract features for each phrase, which will be detailed in Section 5.

4.3 Initial Feature Combinations

While the main focus of our paper is to systematically compare individual features, we find features that perform lower than expected and for which we hypothesize that they may be particularly useful in combinations. For an initial exploration of these hypotheses we evaluate a small number of feature combinations.

In the past, good results have been achieved with combinations of a few relatively simple features, such as $TF.IDF$ and position of first occurrence [11]. The *Kea* system, which generates keyphrases based on these features (the current version also uses keyphrase length), is typically used as the baseline in keyphrase extraction [7, 19]. In our combination experiments we aim to find out wether combinations of features can improve over this baseline, and which combinations are promising.

For our classification experiments we make use the Weka toolkit [29]. For comparison with the Kea system we run our experiments using the same classifier setup as used in Kea, namely Regression by Discretization in combination with the Naive Bayes classifer. Naive Bayes assumes features to be independent and classifies instances based on the probability of the feature values given each class. The regression component allows output of probability estimates which we use to rank the resulting keyphrases.

4.4 Evaluation

Our choice of evaluation measures is based on our view of keyphrase extraction as a ranking problem. Previously, keyphrase extraction has been evaluated using the number of correctly identified keyphrases and measures typical for classification, such as precision and recall (and sometimes f-score). These measures are based on evaluating sets, where there is no ordering in the returned positive and negative instances. A problem with these measures is that there typically is a threshold that needs to be determined, either beforehand or through tuning, to control how many keyphrases an approach to return. Depending on the application, different thresholds may be appropriate, and for comparing methods, an arbitrary threshold needs to be chosen. For these reasons we complement these set-oriented evaluation measures with measures that take ranking into account. We propose the use of evaluation measures for ranked lists as they are typically used in IR:

- Mean reciprocal rank (MRR) is the inverse of the rank of the first correctly returned keyphrase, averaged over all test documents.
- Precision at N (P@N) is the portion of correctly identified keyphrases returned within the top N results.
- Mean average precision (MAP) is the average precision at N, where N takes on the ranks at which correct keyphrases are returned, averaged over all documents.

As some of the new features we propose cannot be generated for all documents, in particular those that use XML markup that is not used in all documents, we also report *coverage*: the portion of documents for which keyphrase suggestions were generated. In case coverage is under 100% we average the remaining evaluation measures only over these covered documents.

We evaluate the top 100 results of the ranked lists of keyphrase suggestions against the author-annotated keyphrases supplied with the documents. For individual features, we rank phrases by that feature in ascending *and* descending order and evaluate the top results of both lists. For combinations we sort by the output score of the learner, where higher scores are ranked at the top. The set of

all annotated keyphrases is used as recall base, so the upper bound on recall is around 75%, depending on the keyphrases that occur in the document and are selected by the candidate selection method. We evaluate using exact phrase matching, without stemming.

5. FEATURES

In this section we detail the concepts and intuitions behind the features we consider for ranking keyphrases for a given document. As explained in Section 1, an analysis of the performance of individual features can help explain what makes a good keyphrase, and can inform the development of effective approaches for keyphrase extraction.

We have described how the collection was preprocessed and indexed, and a list of candidate phrases was selected from the document's full text. For each phrase, we extract the features detailed below. The phrases for each document are then ranked and the resulting result lists are evaluated. Results are presented in Section 6.

We explore features that take into account positions, co-occurrences, and probabilistic properties of phrases on the level of the collection, journal, document, section, and even individual XML element. Below, we detail the intuitions behind each group of features. Each section below stands for a group of features; a feature can combine several concepts, for example when we model structure in a probabilistic way. Notation and formal definitions can be found in Table 3. We do not include lexical features such as PoS-pattern, suffix pattern, or presence of adjectives, because, based on the results obtained by Hulth [13], we assume that these are captured by the way we extract candidate keyphrases (cf. Section 4).

5.1 Position and Length

Position of first occurrence (POS_1) and length in words have been found to be useful indicators for keyphrase extraction early on, and are used in most current keyphrase extraction methods. In addition, the spread between first and last occurrence of phrases have been found beneficial in the related task of link generation [18]. For comparison with prior work we implement POS and SPR on the document's full text and abstract.

5.2 Centrality

Shah et al. [23] describe a scoring method for keyphrase extraction based on the assumption that words central to the main topic of an article have strong associations with many other words. Associations between words are computed based on how often words co-occur in the same sentence. For comparison we re-implement this centrality score for document abstracts and sections (CPa, s, cf. Table 3). We apply a slight modification as we are interested in scoring phrases instead of individual words. We generate co-occurrence statistics for all nouns contained in a phrase, and then use as features the minimum, maximum, and average scores.

5.3 Probabilistic Features

A feature that has been found to perform well for keyphrase extraction is $TFIDF$, the product of normalized term frequency and inverse document frequency. By explicating the relation between keyphrase extraction and information retrieval (IR), we can justify this finding analytically and derive a new set of probabilistic features based on retrieval models. In IR, the task is to retrieve documents that are relevant to a query. Given a query, retrieval algorithms typically return a ranked list of documents, where the documents that are considered most likely to be relevant to the query are ranked at the top. Probabilistic generative retrieval models express this likelihood of being relevant as the probability that a document "generates" a given query, i.e., the probability that a random sample drawn from the document model produces the query. This probability can be expressed as $P(D|t)$. As it is difficult to reliably estimate this probability directly, this term is typically re-written as

$$P(D|t) = \frac{P(t|D)P(D)}{P(t)} \quad (1)$$

using Bayes' Theorem. For ranking documents for a given query, the normalizing term $P(t)$ can be dropped, as it is independent of the query, resulting in $P(D|t) \propto P(t|D)P(D)$. This function can be estimated in different ways, under different assumptions, and a close relation to $TFIDF$ ranking has been identified [12, 22].

In keyphrase extraction, we have a given document and the task is to rank phrases by their representativeness for this document, i.e., the term we are interested in is $P(t|D)$, the probability of drawing a phrase from a given document, which we estimate using the maximum likelihood estimate (MLE).

Apart from only considering document features, we can integrate information about the collection. The intuition that words or phrases that occur in few documents in the collection are more informative than more frequent terms is expressed through the inverse document frequency (IDF). Church and Gale [3] find that informative terms are not distributed equally over all documents. More informative terms are found in relatively fewer documents than could be assumed given a term's frequency in the document collection. They model this finding as *residual IDF* ($RIDF$), the difference between the expected and actually observed IDF.

Expanding on these ideas, we consider the entropy H which models the uncertainty associated with a possible outcome of a random variable. Relative entropy ($RelH(t, R_1, R_2)$) is a measure of the information gain when observing t. Observing a phrase that is very likely to occur in R_1, but rarely occurs in R_2 results in a high information gain, as to which corpus the phrase was drawn from. Similarly to $RelH$, the KL-divergence, $KLD(t, R_1, R_2)$ measures the loss of information if we assume that a phrase was drawn from model R_1 while the true distribution is R_2.

In a task related to keyphrase extraction—extracting back-of-the-book indexes—using the χ^2 test of independence as a scoring component was found to achieve good results [5]. Therefore we include this scoring method in our comparison, following the implementation of the authors. The test statistic measures the degree to which the observed frequency of a phrase deviates from it's expectation given the collection. A higher value corresponds to a stronger association between a phrase-document pair.

In our implementation of probabilistic features we consider phrases as atomic values, i.e., we do not consider the individual words that a phrase is composed of, and the count of terms in a document is the count of candidate phrases generated.

5.4 Collection Structure

As discussed in Section 3, our collection is structured by journal, journal issue, and publication date. We make use of this structure in our experimental design (Section 4.4), but beyond this, the collection structure may provide useful information for keyphrase extraction. E.g., articles within the same journal may discuss similar topics, which we could identify by comparing term distributions between the journal and the collection as a whole.

We focus on the level of journal as a natural grouping within the collection. Although the journals represent two relatively closely related research areas (Computer Science and Food Informatics), there may be differences in document style and format, or in how keyphrases are used. Conventions on the level of journal may, for example, be influenced by the editors, or by the particular research community primarily contributing to a journal.

We incorporate collection structure by considering the journal of an article as the reference corpus J for the probabilistic features

Symbol	Description
t	Phrase
D	Document
C	Document collection
J	Subset of documents in C that were published in journal J
$R \in \{D, J, C\}$	Reference corpus
E	Element type, e.g., t = title, a = abstract, s = sections; other element types are referred to by name. When no element is specified we mean full text.
$S_n(D)$	Section n of document D
$n_E(t, R)$	Term count, i.e., the number of times t occurs in element E in R
$df_E(t, R)$	Document frequency, i.e., the number of documents in R where t occurs in element E
$s_E(t_1, ..., t_n)$	Number of sentences in element E in which phrases $t_1, \ldots, t_n$ co-occur

Feature	Description
$LEN = \|t\|$	Length of phrase t in words [11]
$POS_n(t, D) = \frac{pos_n(t)}{\|D\|}$	Position of the nth occurrence of t in D, normalized by the length of D, not to be confused with PoS (part-of-speech) [11]
$SPR(t, D) = d(POS_{max}(t), POS_1(t))$	Spread, i.e., distance between the last and first occurrences of t in D [29]
$CP_E(t) = \sum_{t'} \frac{\|s_E(t, t')\|}{\|s_E(t)\|}$	Phrase centrality of phrase t, based on word centrality as defined in [23]
$TF_E(t, R) = \frac{n_E(t, R)}{\|R\|}$	Term frequency of phrase t in all elements E in R, normalized by the length of all elements E in R
$IDF_E(t, R) = \log\left(\frac{R}{df_E(t, R)}\right)$	Inverse document frequency of phrase t in elements of type E [11]
$TFIDF_E(t, R) = TF_E(t, R) IDF_E(t, R)$	common statistical measure in IR, models the importance of phrase t for elements of type E in document d [11]
$RIDF_E(t, R) = \log\left(\frac{R}{df_E(t, R)}\right) + \log\left(1 - e^{-\frac{TF_E(t, R)}{R}}\right)$	Residual IDF, i.e., the difference between expected and observed IDF in elements of type E
$MLE_E(t, R) = P_E(t\|R) = \frac{n_E(t, R)}{\sum_{t'} n_E(t', R)}$	Maximum likelihood estimate, i.e., the relative importance of t in elements of type E reference corpus R
$RelMLE_E(t, R_1, R_2) = \frac{MLE_E(t\|R_1)}{MLE_E(t\|R_2)}$	Relative MLE_E between reference corpora R_1 and R_2
$H_E(t, R) = P_E(t\|R) \log P_E(t\|R)$	Entropy of t with respect to R
$RelH_E(t, R_1, R_2) = \frac{H_E(t\|R_1)}{H_E(t\|R_2)}$	Relative entropy H_E between reference corpora R_1 and R_2
$KLD_E(t, R_1, R_2) = P_E(t\|R_1) \log \frac{P_E(t\|R_1)}{P_E(t\|R_2)}$	Kullback-Leibler divergence, the relative entropy between R_1 and R_2 when we assume R_1 [5, 24]
$RelKLD_E(t, R_1, R_2, R_3) = \frac{KLD_E(t, R_1, R_2)}{KLD_E(t, R_1, R_3)}$	Relative KLD_E between reference corpora R_1 and R_2
$\chi^2_E(t, R)$	χ^2 test of independence between occurrence of a candidate phrase t in E and a reference corpus R [5]

Table 3: Notation and features used in this paper; features labeled with a subscript stand for multiple features.

that measure deviation from a background corpus: $MLE(t, D, J)$ gives the ratio of the expected frequency of observing a phrase in the document as opposed to the journal, identifying phrases that are relatively more frequent in the document. $RelH(t, D, J)$ captures the relative reduction in entropy, $KLD(t, D, J)$ quantifies the loss when we assume model D as opposed to J as t is observed. $RelKLD(t, D, J, C)$ gives the ratio of this loss when considering the journal as opposed to the whole collection as the background corpus, and $\chi^2(t, J)$ tests independence between a phrase and a term with expectations obtained from the journal.

We compare probabilistic features implemented on the journal level to those based on the whole collection. In addition, we evaluate these rankings on subsets of the corpus corresponding to five journals that are most frequently represented in the collection.

5.5 Document Structure

Many document types, or genres, exhibit a characteristic style and form. E.g., news articles typically have a headline summarizing the article, an indication of the news source, location and date, etc. Both authors and readers are aware of these conventions and use them to effectively process the document content. Similarly, scientific papers are subject to constraints in form and style, that have developed over time, and are, for example, enforced through the review process, which results in a certain degree of standardization [4]. Experienced readers of scientific articles have been found to form a mental model of the typical structure of a research article and to use this model for selective reading [6].

In contrast to corpora previously used for keyphrase extraction, our corpus preserves the clean document structure available at the publishing stage of scientific articles. We use this structure in two ways. First, we model the content of markup elements to identify whether some of these elements are useful for keyphrase extraction, and which elements are the most informative. Second, we focus on section structure and augment the markup with position and clues in the section headers to identify main section types.

5.5.1 XML markup

We assume that the types of markup elements in our document collection have semantics that are similar across documents. Overall, some types of elements may be more likely to contain content representative of the document content than others. E.g., a reader may be more likely to find information about the document topic in the title than in the author's contact information. In this case the XML markup may serve different purposes. It can explicate struc-

ture corresponding to the conventions mentioned above (marking the document title, abstract, sections and section headings, lists, figure captions, etc.). But the specific markup format has been developed by a publishing house for use during the electronic publishing process and may not necessarily be relevant for readers. The markup ranges from coarse (e.g., section) to very fine granularity (e.g., individual symbols within a mathematical formula, individual cells of a table). Thus, some markup may be helpful for identifying important phrases in the document, while others are not.

We model XML markup probabilistically, using the probabilistic features described above. We include every XML element type that was found to contain at least one keyphrase at least once in our development set. We compare these features to identify element types that are most likely to contain keyphrases, to determine how we can best capture this information.

5.5.2 *Sections*

A particularly important structural element of scientific articles are sections and a lot of research is concerned with modeling the discourse structure created through the use of section types [15]. Given the different functions of section types (introduction, results presentation, etc.), the content of some sections may be more representative of a document's topic than others. This hypothesis is supported by Shah et al. [23] who analyzed occurrences of MeSH[6] terms by section type in 104 articles of a biomedical journal. The authors found a relatively higher concentration of MeSH terms in abstracts and methods sections, and also found qualitative differences between the different sections.

To identify generic section types we make use of two types of cues: (i) position and (ii) characteristic words in section titles [15]. First, we identify top-level sections based on section numbering. Position is then inferred from the ordering of the top-level sections in the document, and we include features for the first N and last N sections (in our case we set N to 10, a number chosen to exceed our estimate of generic section types identifiable based on position).

Type	cue words	count
Introduction	introduction	954
Background	background, related work	114
Method	method	373
Result	result	415
Discussion	discussion	410
Conclusion	conclusion, concluding, summary	651

Table 4: Generic section types, cue words, and frequency of occurrence on the development set. For 98% of the documents at least one generic section type can be identified using cue words.

Characteristic words in section headings were obtained from the most frequent section titles of documents in the development set. Frequent section headings were grouped by functions; cue words were manually extracted. All top-level sections containing a cue word were assigned to the corresponding type; Table 4 summarizes the results. For each generic section type we generate probabilistic scores as described before and draw comparisons between section types and with features based on the full document text.

6. RESULTS AND DISCUSSION

In this section we present results to address our research questions. First we evaluate how features that have been found to perform well in previous work perform on our new collection. We consider position and phrase length (6.1), centrality (6.2), and probabilistic models, including our novel ones (6.3). Because previous work has performed evaluation either on full text documents or on abstracts only, we include both for comparison. Next, we turn to the features related to *document structure*, namely in the XML markup (6.4), and section structure (6.5). Concerning *collection structure*, we report on performance of models that use the journal as background corpus, and we also compare the best-performing features when evaluated on articles of individual journals (6.6). Finally, we report on a small number of feature combinations, and compare them with previous approaches to demonstrate that our proposed features indeed improve keyphrase extraction performance (6.7) over other established methods.

Because we explore a very large number of features it is impractical to report results for all. Thus, we only report on and compare results for selected features, in particular those that perform well or those that are interesting otherwise. The full list of results will be made available online as a supplement to this paper[7].

We analyze the performance of individual features on the training data set as described in Section 4. Selected feature combinations are trained on the training set and performance is reported on the test set. In all result tables we mark best scores per column in bold face. Triangles next to rankings of individual feature indicate whether the ranking was ascending ($^{\triangle}$, i.e. lower scores correspond to better keyphrases) or descending ($^{\triangledown}$, higher scores indicate better keyphrases). Recall that we evaluate 100 keyphrases each time, so a different sorting yields different results.

feature	coverage	P@10	recall	MRR	MAP
$LEN^{\triangle}$	100%	0.0574	0.1766	0.1688	0.0033
$POS_{1,abstract}{}^{\triangle}$	100%	0.0265	0.0906	0.0967	0.0011
$POS_{1,abstract}{}^{\triangledown}$	100%	0.0099	0.2300	0.0442	0.0008
$POS_{1,fulltext}{}^{\triangle}$	100%	**0.0821**	0.3975	**0.2237**	0.0043
$POS_{n,abstract}{}^{\triangledown}$	100%	0.0306	0.3186	0.1164	0.0021
$POS_{n,fulltext}{}^{\triangledown}$	100%	0.0215	0.2166	0.0826	0.0013
$SPR_{abstract}{}^{\triangledown}$	100%	0.0719	**0.4069**	0.2232	**0.0045**
$SPR_{fulltext}{}^{\triangledown}$	100%	0.0738	0.3593	0.2197	0.0041

Table 5: Performance of several individual features in abstract and full text which performed well in earlier work.

6.1 Position and Length

Four simple features that have been found to perform well in previous work are POS_1, LEN, SPR, and POS_n. In Table 5 we show the most interesting results of these individual features using the abstract and full-text representation of the documents. From this table we observe that there are distinct performance differences between full-text and abstract. On early precision, $POS_{1,fulltext}$ performs best, while the same measure calculated on the abstract performs much worse. In fact, while all other measures show a clear tendency as to whether higher or lower scores are associated with good keyphrase candidates, $POS_{1,abstract}$ shows higher precision values when phrases are ranked in ascending order, while in the reverse order recall is higher. This result corresponds to important concepts being introduced either at the beginning, or at the end of the abstract. SPR, a feature adapted from the related task of link generation, is found to perform well, with $SPR_{abstract}$ achieving highest recall and MAP scores in this group of features and similar scores on the full text document. LEN is not expected to achieve meaningful results, and stays well below the remaining features. However, it may be useful in combination with other features to steer preferences for longer or shorter keyphrases.

[6]Medical Subject Headings—a controlled vocabulary of indexing terms.

[7]Url removed to preserve anonymity.

feature	coverage	P@10	recall	MRR	MAP
$CP_{abstract,max}{}^{\nabla}$	100%	**0.0245**	**0.1263**	**0.0920**	**0.0013**
$CP_{abstract,avg}{}^{\nabla}$	100%	0.0214	0.1061	0.0832	0.0012
$CP_{abstract,min}{}^{\nabla}$	100%	0.0214	0.1065	0.0833	0.0012

Table 6: Performance of rankings by centrality score on document abstracts. Results for full text documents stayed far below those for abstracts.

Overall, we find relatively low precision of up to 8%, which translates to less than one correct keyphrase returned in the top-10. Recall in the top-100 list of these features is reasonably high with about 41% of the ground truth keyphrases returned on average. MAP is low, suggesting that the correct instances are spread out within the result list instead of being concentrated towards the top of the list.

6.2 Centrality

An approach to keyphrase extraction based on the concept of centrality has been used in the past [23], but was not compared to other methods. When evaluated on the current collection, it achieves very low scores when used individually for ranking candidate phrases (Table 6). From this table we also note that its performance stays well below the ones we saw in the previous section. The reason for this may be attributed to the fact that the current implementation only takes nouns into account. Indeed, upon manual inspection of the generated keyphrases, we find that the identified nouns appear to be very central to the topic of the document. The low performance results from the fact that all phrases sharing the same nouns get the same score and are returned in "random" order. This indicates that the centrality score does capture some useful information but should be combined with other features.

6.3 Probabilistic Features

$TFIDF$ is the most commonly used feature for keyphrase extraction, and our results confirm that this is a good choice (Table 7). Out of all probabilistic models it achieves highest scores, with $TFIDF_{fulltext}$ performing slightly higher on most measures; KLD performs similarly well. Of the features we have considered so far, the probabilistic ones achieve the highest precision, with only recall being exceeded by the positional feature $SPR_{abstract}$.

We further observe that performance is low on $RelH$. Interestingly, $RIDF$ performs substantially better than IDF. Between abstract and full text there is no clear winner. The abstract works better for MLE, IDF, and $RelH$, but these features perform low in general. On the best-performing probabilistic models $TFIDF$ and KLD, their performance is very similar. It appears that the features with large difference, i.e. IDF, $RIDF$, etc. are more susceptible to large differences in language use. KLD and $TFIDF$ tend to be more robust to varied language use, but that also means that they are less able to pick up finer variations when necessary.

6.4 XML Markup

For XML markup features we report $TFIDF$ on the features with the highest scores and high coverage (Table 8). Coverage is relatively low for these features, as many XML markup codes are only used in some articles. As expected, the best-performing elements are, the $TFIDF$ scores for abstract, sections, and title. However, there are many other elements that achieve high performance, such as the bibliography ($ce : bibliography - sec$), captions ($ce : caption$), and table headings ($thead$). Elements starting with "$sb :$" are sub-elements of the bibliography. As such, a high $TFIDF$ of a phrase in cited article title, books, etc. is a good indicator for keyphrases.

feature	cov.	P@10	recall	MRR	MAP
$IDF_{abstract}{}^{\nabla}$	100%	0.0316	0.1233	0.1596	0.0021
$IDF_{fulltext}{}^{\nabla}$	100%	0.0161	0.0547	0.0681	0.0008
$TFIDF_{abstract}{}^{\nabla}$	100%	0.0957	0.3578	**0.3515**	0.0067
$TFIDF_{fulltext}{}^{\nabla}$	100%	**0.1001**	**0.4051**	0.3366	**0.0074**
$RIDF_{abstract}{}^{\nabla}$	100%	0.0399	0.2036	0.1179	0.0026
$RIDF_{fulltext}{}^{\nabla}$	100%	0.0629	0.2757	0.2161	0.0048
$MLE_{abstract}(t,D)^{\nabla}$	100%	0.0781	0.3703	0.2249	0.0051
$MLE_{fulltext}(t,D)^{\nabla}$	100%	0.0590	0.3331	0.1620	0.0036
$RelH_{abstract}(t,D,C)^{\nabla}$	100%	0.0166	0.3189	0.1000	0.0018
$RelH_{fulltext}(t,D,C)^{\nabla}$	100%	0.0018	0.0065	0.0088	0.0001
$KLD_{abstract}(t,D,C)^{\nabla}$	100%	0.0941	0.3541	0.3289	0.0063
$KLD_{fulltext}(t,D,C)^{\nabla}$	100%	0.0948	0.4010	0.3109	0.0069
$\chi^2_{abstract}(t,D,C)^{\nabla}$	100%	0.0601	0.3393	0.2629	0.0042
$\chi^2_{fulltext}(t,D,C)^{\nabla}$	100%	0.0704	0.2384	0.2827	0.0047

Table 7: Performance of features based on probabilistic models for abstracts and full text. $TFIDF$ performs best overall, KLD achieves similar scores.

feature	coverage	P@10	recall	MRR	MAP
$TFIDF_{ce:abstract-sec}{}^{\Delta}$	100%	**0.0957**	0.3578	**0.3515**	0.0067
$TFIDF_{ce:bibliography-sec}{}^{\Delta}$	100%	0.0761	0.3319	0.2890	0.0055
$TFIDF_{ce:caption}{}^{\Delta}$	91%	0.0603	0.1777	0.2321	0.0094
$TFIDF_{ce:sections}{}^{\Delta}$	100%	0.0955	**0.3768**	0.3272	0.0071
$TFIDF_{ce:simple-para}{}^{\Delta}$	100%	0.0935	0.3582	0.3317	0.0067
$TFIDF_{ce:title}{}^{\Delta}$	100%	0.0762	0.2411	0.2078	**0.0267**
$TFIDF_{ce:table}{}^{\Delta}$	70%	0.0512	0.1575	0.1728	0.0056
$TFIDF_{thead}{}^{\Delta}$	50%	0.0320	0.0618	0.1246	0.0149
$TFIDF_{sb:book}{}^{\Delta}$	56%	0.0147	0.0517	0.0839	0.0104
$TFIDF_{sb:edited-book}{}^{\Delta}$	56%	0.0197	0.0619	0.1244	0.0150
$TFIDF_{sb:maintitle}{}^{\Delta}$	98%	0.0839	0.2984	0.3084	0.0093
$TFIDF_{sb:title}{}^{\Delta}$	98%	0.0845	0.3002	0.3076	0.0091

Table 8: Performance of $TFIDF$ of XML markup features. Best performance is achieved with abstract, title, and sections.

6.5 Section Structure

Generic section types show an interesting pattern (Table 9). "Introduction" seems to be the most general type and is found in almost 80% of the test documents and keyphrase extraction performance is also good. Background has very low coverage and low performance. 'Method", "result" and "conclusion" sections show medium coverage, and good performance. By far the best is the "discussion" section, which even performs better than when generating scores on the full-text. These results indicate that, when we can identify section types, such information can be very useful for keyphrase extraction.

For sections of type "discussion" we include additional probabilistic features. Here we find a similar performance, for example in IDF and MLE to the ones we observed for abstracts (cf. Section 6.3). This suggests that performance of these features is indeed related to the topical coherence of the text samples.

As expected, the sections based on position have higher coverage than section types identified based on section. Beyond section 3 coverage drops, as there are fewer documents with more than three sections. Performance of the features based on the first and last sections is good. We assume that these sections correspond to "introduction" and "discussion" / "conclusion" (given the scores, discussion is more likely).

In comparison with [23] we see similarities and differences which may be attributed to the field of the documents in question. They found the largest number of keywords on average in the methods and introduction sections, and the highest concentration (keyword over section length) in the abstract and introduction. We also get

feature	coverage	P@10	recall	MRR	MAP
$TFIDF_1{}^{\nabla}$	83%	0.0941	0.3098	0.3590	0.0062
$TFIDF_2{}^{\nabla}$	83%	0.0701	0.2310	0.2645	0.0048
$TFIDF_3{}^{\nabla}$	83%	0.0696	0.2217	0.2650	0.0051
$TFIDF_n{}^{\nabla}$	83%	0.0968	0.3123	0.3445	0.0066
$TFIDF_{n-1}{}^{\nabla}$	83%	0.0740	0.2508	0.2700	0.0052
$TFIDF_{n-2}{}^{\nabla}$	83%	0.0689	0.2391	0.2517	0.0047
$TFIDF_{INTR}{}^{\nabla}$	79%	0.0956	0.3118	0.3619	0.0063
$TFIDF_{BACK}{}^{\nabla}$	9%	0.0612	0.2652	0.2357	0.0034
$TFIDF_{METHOD}{}^{\nabla}$	33%	0.0960	0.2304	0.3310	0.0072
$TFIDF_{RESULT}{}^{\nabla}$	36%	0.0944	0.2282	0.3481	0.0077
$TFIDF_{DISC}{}^{\nabla}$	40%	0.1205	0.3112	0.4212	0.0092
$TFIDF_{CONCL}{}^{\nabla}$	47%	0.0728	0.2933	0.2748	0.0042
$TF_{DISC}{}^{\nabla}$	40%	0.0827	0.3124	0.2378	0.0052
$IDF_{DISC}{}^{\nabla}$	40%	0.0391	0.0917	0.1689	0.0024
$RIDF_{DISC}{}^{\nabla}$	40%	0.0737	0.2499	0.2190	0.0056
$MLE_{DISC}{}^{\nabla}$	40%	0.0828	0.3125	0.2378	0.0052
$KLD_{DISC}{}^{\nabla}$	40%	0.1170	0.3096	0.3971	0.0086
$\chi^2_{DISC}{}^{\nabla}$	40%	0.0822	0.1829	0.3414	0.0058

Table 9: Probabilistic features for generic section types. The discussion section, as well as first and last sections perform best.

good performance on the introduction, but find the discussion section to perform best overall.

6.6 Collection Structure

feature	cov.	P@10	recall	MRR	MAP
$IDF_{abstract}(t,J)^{\nabla}$	100%	0.0391	0.1561	0.1881	0.0025
$IDF_{fulltext}(t,J)^{\nabla}$	100%	0.0206	0.0779	0.0793	0.0010
$RIDF_{abstract}(t,J)^{\nabla}$	100%	0.0440	0.2330	0.1290	0.0029
$RIDF_{fulltext}(t,J)^{\nabla}$	100%	0.0668	0.3060	0.2259	0.0053
$TFIDF_{abstract}(t,J)^{\nabla}$	100%	0.0943	0.3564	**0.3357**	0.0060
$TFIDF_{fulltext}(t,J)^{\nabla}$	100%	**0.0957**	0.3885	0.3117	**0.0070**
$MLE_{abstract}(t,J)^{\nabla}$	100%	0.0040	0.0671	0.0190	0.0002
$MLE_{fulltext}(t,J)^{\nabla}$	100%	0.0027	0.0407	0.0111	0.0001
$RelMLE_{abstract}(t,D,J)^{\nabla}$	100%	0.0469	0.3176	0.2136	0.0032
$RelMLE_{fulltext}(t,D,J)^{\nabla}$	100%	0.0293	0.0870	0.1320	0.0018
$RelMLE_{abstract}(t,J,C)^{\nabla}$	100%	0.0367	0.2546	0.1343	0.0024
$RelMLE_{fulltext}(t,J,C)^{\nabla}$	100%	0.0307	0.1371	0.1071	0.0018
$KLD_{abstract}(t,D,J)^{\nabla}$	100%	0.0914	0.3538	0.3128	0.0057
$KLD_{fulltext}(t,D,J)^{\nabla}$	100%	0.0927	**0.3895**	0.2978	0.0066
$RelKLD_{abstract}(t,D,J,C)^{\nabla}$	100%	0.0045	0.2492	0.0281	0.0008
$RelKLD_{fulltext}(t,D,J,C)^{\nabla}$	100%	0.0002	0.0063	0.0016	0.0000
$\chi^2_{abstract}(t,J)^{\nabla}$	100%	0.0693	0.3269	0.2775	0.0045
$\chi^2_{fulltext}(t,J)^{\nabla}$	100%	0.0781	0.2791	0.2921	0.0052

Table 10: Probabilistic features based on journal. Like on the collection as a whole $TFIDF$ and KLD perform best.

To utilize collection structure we have evaluated the probabilistic models at the journal level. Some features use the journal as background corpus, whilst others exploit the difference between journal and collection. Results are summarized in Table 10. We find that many features perform better using the journal than when using the whole collection, in particular IDF, $RIDF$, MLE and $RelMLE$, and χ^2. This suggests that for these features to work well, the document collection as a whole may be too general or too noisy. The best-performing individual features are KLD and $TFIDF$, as with the overall collection.

To further investigate the relation between documents, journals, and the collection as a whole, we also evaluated on the 5 journals for which the most articles are available in our collection; the results of which are shown in Table 11. We report the 5 best performing features per journal, determined by performance on P@10.

We see that there are large performance differences between individual journals. On the most frequent journal the highest P@10

feature	coverage	P@10	recall	MRR	MAP
Food and Chemical Toxicology (1199 articles in 92 issues)					
$TFIDF_{fulltext}(t,J)^{\nabla}$	100%	**0.1874**	0.3411	0.5788	0.0170
$TFIDF_{abstract}(t,D)^{\nabla}$	100%	0.1869	**0.3566**	**0.5850**	**0.0172**
$KLD_{fulltext}(t,D,J)^{\nabla}$	100%	0.1799	0.3416	0.5695	0.0161
$KLD_{fulltext}(t,D,C)^{\nabla}$	100%	0.1755	0.3499	0.5723	0.0160
$RIDF_{fulltext}(t,J)^{\nabla}$	100%	0.1584	0.3210	0.4744	0.0156
Computer Networks (982 articles in 115 issues)					
$TFIDF_{abstract}{}^{\nabla}$	100%	**0.0740**	0.3715	**0.2830**	**0.0043**
$KLD_{abstract}(t,D,C)^{\nabla}$	100%	0.0718	0.3676	0.2749	0.0042
$TFIDF_{abstract}(t,J)^{\nabla}$	100%	0.0701	0.3585	0.2744	0.0041
$POS_{1,fulltext}{}^{\Delta}$	100%	0.0692	**0.3925**	0.1935	0.0031
$TFIDF_{fulltext}(t,D)^{\nabla}$	100%	0.0668	0.3833	0.2248	0.0041
International Journal of Medical Informatics (704 articles in 66 issues)					
$TFIDF_{fulltext}(t,D)$	100%	**0.0740**	**0.4061**	**0.2747**	**0.0046**
$KLD_{fulltext}(t,D,C)^{\nabla}$	100%	0.0710	0.4031	0.2409	0.0042
$KLD_{fulltext}(t,D,J)^{\nabla}$	100%	0.0678	0.3821	0.2344	0.0039
$POS_{1,fulltext}{}^{\Delta}$	100%	0.0649	0.3744	0.1930	0.0032
$TFIDF_{fulltext}(t,J)$	100%	0.0646	0.3716	0.2348	0.0038
Information Sciences (669 articles in 90 issues)					
$TFIDF_{abstract}(t,J)^{\nabla}$	100%	**0.0757**	0.4212	0.2580	0.0042
$TFIDF_{abstract}(t,D)^{\nabla}$	100%	0.0750	**0.4222**	**0.2776**	**0.0044**
$KLD_{abstract}(t,D,J)^{\nabla}$	100%	0.0726	0.4189	0.2347	0.0039
$KLD_{abstract}(t,D,C)^{\nabla}$	100%	0.0712	0.4206	0.2482	0.0040
$\chi^2_{abstract}(t,J)$	100%	0.0674	0.3952	0.2726	0.0041
Cognition (535 articles in 101 issues)					
$POS_{1,fulltext}{}^{\Delta}$	100%	**0.1067**	**0.5435**	**0.3091**	**0.0055**
$KLD_{abstract}(t,D,C)^{\nabla}$	100%	0.0835	0.4530	0.2811	0.0044
$TFIDF_{abstract}(t,D)^{\nabla}$	100%	0.0829	0.4568	0.2968	0.0046
$KLD_{abstract}(t,D,J)^{\nabla}$	100%	0.0811	0.4458	0.2646	0.0041
$TFIDF_{abstract}(t,J)^{\nabla}$	100%	0.0805	0.4490	0.2837	0.0043

Table 11: Performance of the top-5 features (selected by P@10) of the 5 journals with the most articles.

value of 0.1874 is almost twice that of that achieved on the overall collection. For the same journal, MRR goes up to 0.5788, corresponding to the first correct result being found at rank 2 on average. The highest recall values range from 34% up to 54%. For all journals variants, KLD and $TFIDF$ are among the best-performing features, but for some journals the collection-based versions perform better. Other high-performing features are $POS_{1,fulltext}$, and in one case each $RIDF_{fulltext}(t,J)$ and $\chi^2 abstract(t,J)$. We think that the $TFIDF$ and KLD are generally robust, but that other features may be useful for tuning in to the language use of a particular journal.

6.7 Combining Features

In our analysis we have identified features for which we expect to improve keyphrase extraction performance when used in combination with other features. We perform a small number of experiments to follow up on these intuitions. A more systematic exploration of feature combinations is beyond the scope of this paper.

For comparison we run the *Kea* system—which is typically used as baseline system —and compare results with the most recent version [11]. This system uses three features: LEN, POS_1, and $TFIDF$, all extracted from full text documents. It uses a machine learning algorithm called *Regression-by-Discretization* which wraps around a Naive Bayes classifier. As Kea cannot handle XML files we converted the documents' title, abstract, and sections to plain text. The settings were adjusted to not use a controlled vocabulary, and no lower threshold for phrase occurrence was used. We also re-implemented the approach taken by *Kea* to reflect our candidate phrases and features, which is marked $Comb1$ below. Performance of both is shown in Table 12.

feature	coverage	P@10	recall	MRR	MAP
KEA system	100%	0.1058	0.4695	0.3588	0.0069
$Comb1$	100%	0.1144	0.4929	0.3643	0.0074
$Comb1 + SPR_{abstract}$	100%	0.0918	0.4306	0.3343	0.0060
$Comb1 + CP_{abstract}$	100%	**0.1232**	0.4999	**0.4033**	**0.0081**
$Comb1 + probabilities$	100%	0.1041	0.4352	0.3652	0.0067
$Comb1 + sections$	100%	0.1098	0.4883	0.3589	0.0071
$Comb1 + xml$	100%	0.1069	**0.5071**	0.3346	0.0068
$Comb1 + journal$	100%	0.1046	0.4613	0.3487	0.0067

Table 12: Performance of initial feature combinations.

We observe that our re-implementation performs better than Kea, which can be attributed to the candidate selection strategy. Indeed, we find that Kea produces more negative instances, which makes the learning task more difficult. Spread does not add to the performance of the baseline feature set. The reason may be that it is highly correlated with POS_1, which is already included in the features set. It is known that the Naive Bayes classifier does not perform well on feature combinations with high correlations, as it is based on the assumption of feature independence.

A substantial improvement is achieved using the centrality score CP. We had hypothesized that its low individual performance is due to the fact that it does not distinguish between different phrases containing the same central nouns. From the combinations we see that this is indeed the case, and that it can work well in combinations with other features. The combination with XML features results in the highest recall. This suggests that including features based on different elements can boost correct keyphrases up on the result list. The combinations with section and journal features do not improve over the baseline combination. Overall, these combinations indicate that keyphrase extraction can be improved by using a diverse set of features.

7. CONCLUSION AND FUTURE WORK

In this paper we have systematically compared features for keyphrase extraction on a large corpus of scientific journal articles with rich semantic annotations. We have proposed the use of structural features that utilize the rich markup of the documents and have compared them to a standard baseline and a large number of features that have been suggested in prior work.

Our main finding is that differentiation, both between different elements of the document and parts of the collection, can contribute useful information for keyphrase extraction. On the level of *document structure* we find that probabilistic models for generic section types perform well, in particular the discussion section appears to be informative for this task. On the level of *collection structure* we find large differences in performance on different journals, and for different journals different features show best performance.

We explore a small number of feature combinations using a Naive Bayes classifier to demonstrate that combinations of different groups of features can outperform a standard baseline. An interesting direction for future work is to develop further combinations using more advanced learning or optimization approaches that specifically address the characteristics of this task.

8. REFERENCES

[1] T. D. Anderson. Studying human judgments of relevance: interactions in context. In *IIiX '06*, 2006.

[2] M. Bendersky and B. W. Croft. Discovering key concepts in verbose queries. In *SIGIR '08*, 2008.

[3] K. W. Church and W. A. Gale. Inverse document frequency (idf): A measure of deviations from poisson. In *Proc. Third Workshop on Very Large Corpora*, 1995.

[4] G. Crookes. Towards a validated analysis of scientific text structure. *Applied Linguistics*, 7(1):57–70, 1986.

[5] A. Csomai and R. Mihalcea. Investigations in unsupervised back-of-the-book indexing. In *FAIRSC '07*, 2007.

[6] A. Dillon. Readers' models of text structures: the case of academic articles. *Intern. J. of Man-Machine Studies*, 35(6):913–925, 1991.

[7] S. El-Beltagy and A. Rafea. KP-Miner: A keyphrase extraction system for english and arabic documents. *Information Systems*, 34(1): 132–144, 2009.

[8] E. Frank, G. W. Paynter, I. H. Witten, C. Gutwin, and C. G. Nevill-Manning. Domain-specific keyphrase extraction. In *IJCAI'99*, 1999.

[9] X. Geng, T.-Y. Liu, T. Qin, and H. Li. Feature selection for ranking. In *SIGIR '07*, 2007.

[10] I. Gil-Leiva and A. Alonso-Arroyo. Keywords given by authors of scientific articles in database descriptors. *JASIST*, 58(8):1175–1187, 2007.

[11] C. Gutwin. Improving browsing in digital libraries with keyphrase indexes. *Decision Support Systems*, 27(1-2):81–104, 1999.

[12] D. Hiemstra. A probabilistic justification for using tf.idf term weighting in information retrieval. *Intern. J. on Digital Libraries*, 3(2):131–139, 2000.

[13] A. Hulth. *Combining Machine Learning and Natural Language Processing for Automatic Keyword Extraction*. PhD thesis, Stockholm University, 2004.

[14] T. Joyce and R. M. Needham. The thesaurus approach to information retrieval. *American Documentation*, 9(3):192–197, 1958.

[15] N. Kando. Text-level structure of research papers: Implications for text-based information processing systems. In *Proc. British Comp. Soc. Ann. Coll. Information Retrieval Research*, 1997.

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

[17] O. Medelyan and I. H. Witten. Domain-independent automatic keyphrase indexing with small training sets. *JASIST*, 59(7):1026–1040, 2008.

[18] D. Milne and I. H. Witten. Learning to link with wikipedia. In *CIKM '08*, 2008.

[19] T. Nguyen and M.-Y. Kan. Keyphrase extraction in scientific publications. *Asian Digital Libraries. Looking Back 10 Years and Forging New Frontiers*, pages 317–326, 2007.

[20] C. Perez-Iratxeta, H. S. Keer, P. Bork, and M. A. Andrade. Computing fuzzy associations for the analysis of biological literature. *Biotechniques*, 32(6), 2002.

[21] N. Roberts. The pre-history of the information retrieval thesaurus. *J. of Documentation*, 40:271–285(15), 1984.

[22] T. Roelleke and J. Wang. Tf-idf uncovered: a study of theories and probabilities. In *SIGIR '08*, 2008.

[23] P. K. Shah, C. Perez-Iratxeta, P. Bork, and M. A. Andrade. Information extraction from full text scientific articles: where are the keywords? *BMC Bioinformatics*, 4(1), 2003.

[24] T. Tomokiyo and M. Hurst. A language model approach to keyphrase extraction. In *Proc. ACL 2003 Workshop on Multiword expressions*, 2003.

[25] P. D. Turney. Learning to extract keyphrases from text. Technical Report ERB-1057, Nat. Research Council, Inst. Inform. Techn., 1999.

[26] P. D. Turney. Learning algorithms for keyphrase extraction. *Information Retrieval*, 2:303–336, 2000.

[27] P. D. Turney. Coherent keyphrase extraction via web mining. In *IJCAI '03*, 2003.

[28] J. Wang and H. Peng. Keyphrases extraction from web document by the least squares support vector machine. In *Web Intelligence '05*, 2005.

[29] I. H. Witten and E. Frank. *Data mining: practical machine learning tools and techniques*. Morgan Kaufmann Publishers Inc., 2005.

[30] I. H. Witten, G. W. Paynter, E. Frank, C. Gutwin, and C. G. Nevill-Manning. KEA: practical automatic keyphrase extraction. In *DL '99*, 1999.

[31] H. Zha. Generic summarization and keyphrase extraction using mutual reinforcement principle and sentence clustering. In *SIGIR '02*, 2002.