# Keyword Extraction using Word Co-occurrence

Christian Wartena
Novay
Enschede, The Netherlands
Christian.Wartena@novay.nl

Rogier Brussee
University of Applied Sciences Utrecht
Utrecht, The Netherlands
Rogier.Brussee@hu.nl

Wout Slakhorst
Novay
Enschede, The Netherlands
Wout.Slakhorst@novay.nl

***Abstract*—A common strategy to assign keywords to documents is to select the most appropriate words from the document text. One of the most important criteria for a word to be selected as keyword is its relevance for the text. The *tf.idf* score of a term is a widely used relevance measure. While easy to compute and giving quite satisfactory results, this measure does not take (semantic) relations between words into account. In this paper we study some alternative relevance measures that do use relations between words. They are computed by defining co-occurrence distributions for words and comparing these distributions with the document and the corpus distribution. We then evaluate keyword extraction algorithms defined by selecting different relevance measures. For two corpora of abstracts with manually assigned keywords, we compare manually extracted keywords with different automatically extracted ones. The results show that using word co-occurrence information can improve precision and recall over *tf.idf*.**

## I. Introduction

Keywords provide a concise and precise high-level summarization of a document. They therefore constitute an important feature for document retrieval, classification, topic search and other tasks even if full text search is available.

Both the importance and cost of manual annotations have led to considerable interest in automatic keyword extraction. The basic idea is to select words from a text that gives a good impression of its content. Many keyword selection criteria have been formulated using either properties of the word in the text and collections of texts or using external resources like thesauri. In most approaches however, the main feature is the relevance of the word in the text as expressed by the classic *tf.idf*–value.

The *tf.idf*-measure combines two aspects of a word: the importance of a word for a document and its discriminative power within the whole collection. These two aspects match well with the general intuition for suitability of words as good keywords. However, the *tf.idf* measure heuristics assume that words are completely independent. We will show in the following that there is room for improvement if we also take correlations between words into account. For the discriminative power of a keyword, *tf.idf* uses the number of documents in which a word is used. The smaller the number, the more distinguishing the word is. However, we can be more precise. A word that occurs in a number of documents on the *same topic* has more discriminative power than a word occurring in the same number of documents but scattered over different topics. In the following we will introduce co-occurrence distributions of words that encodes information about related words. We then show that these distributions can be used to measure the importance and discriminative power of a term.

The paper is organized as follows: In section II we give an overview of the state of the art in keyword extraction. In section III we introduce co-occurrence distributions and show how these distributions can help to include co-occurrence information in the definition of relevance of a (key)word for a text. Section IV shortly presents the preprocessing that is done to extract candidate terms from the texts. In section V we evaluate three possible relevance measures for keyword extraction on two datasets.

## II. Related Work

Extracting keywords from a text is closely related to ranking words in the text by their relevance for the text. To first approximation, the best keywords are the most relevant words in the text. Determining the right weight structure for words in a text is a central area of research since the late 1960's ([1]). In 1972 Spärck Jones (reprinted as [2]) proposed a weighting for specificity of a term based on $1 + log(\#\text{documents}/\#\text{term occurrences})$. This term weighting, which has become known as *tf.idf*, is subsequently refined in [3], studied in the light of latent semantic analysis by [4], given a detailed statistical analysis by [5], and a probabilistic interpretation by [6]. An information theoretic explanation of *tf.idf* is given by [7].

Keywords are not simply the most specific or most distinguishing words of a text, as keywords are (at least partially) intended for human readers with their own. Other features besides frequencies counted in a corpus of text may therefore also play a role in keyword selection. If keyword extraction is treated as a supervised machine learning problem the integration of different types of features is straightforward. This approach to keyword extraction was proposed by [8] and [9] and subsequently followed by many others. In [8] 4 different features are used: term frequency, collection frequency, relative position of the (first occurrence of) the word in the text, and number of times a term is used as keyword. In subsequent work the value of other features is studied. In [10] and [11] the average *tf.idf* value of surrounding words and the number of phrases modifying and modified by the given key phrase is used as an additional feature.

Mihalcea and Tarau [12] use the sentence structure of the text in a way somewhat similar to the co-occurrence methods of this paper. For each document a graph of terms is build in which the link strength is determined by the probability that the terms occurs in the same sentence. In the same spirit [13] compute for each word the distribution of words co-occurring in the same sentences and compare this distribution with the general term distribution to detect terms with special behavior. The last two techniques are designed to work with relatively long texts. In the following we will however concentrate mainly on (very) short abstracts.

In this paper we will not follow the machine learning approach, but concentrate on a single measure for the suitability of a keyword for a document. The reason for this is twofold. In the first place there are many situation in which keyword extraction could be useful but in which no training data are available. In the second place, the measures we propose in this paper can be used in a machine learning approach and can potentially improve results in this setting as well.

## III. Distributions of words co-occurring with (key)words

We will use co-occurrence of words as the primary way of quantifying semantic relations between words. According to the distributional hypothesis ([14], [15]) semantically similar words occur in similar contexts, i.e. they co-occur with the same other words. Therefore rather than using the immediate co-occurrence of two terms as a measure for their semantic similarity we will compare the co-occurrences of the terms with all other terms. We formalize this intuition by defining a so called co-occurrence distribution of each word which is simply the weighted average of the word distributions of all documents in which the word occurs. We then operationalize the "semantic similarity" of two terms by computing similarity measure(s) for their co-occurrence distributions. The co-occurrence distribution of a word can also be compared with the word distribution of a text. This gives us a measure to determine how typical a word is for a text.

### A. Basic Distributions

We simplify a document to a bag of words. Thus, consider a set of $n$ term occurrences $\mathcal{W}$ each being an instance of a term $t$ in $\mathcal{T} = \{t_1, \dots t_m\}$, and each occurring in a source document $d$ in a collection $\mathcal{C} = \{d_1, \dots d_M\}$. Let $n(d,t)$ be the number of occurrences of term $t$ in $d$, $n(t) = \sum_d n(d,t)$ the number of occurrences $t$, and $N(d) = \sum_t n(d,t)$ the number of term occurrences in $d$. We define probability distributions

$$\begin{aligned} Q(d) &= N(d)/n && \text{on } \mathcal{C} \\ q(t) &= n(t)/n && \text{on } \mathcal{T} \end{aligned}$$

that measure the probability to randomly select a term or a source document. In addition we have the conditional probability distributions

$$\begin{aligned} Q(d|t) = Q_t(d) &= n(d,t)/n(t) && \text{on } \mathcal{C} \\ q(t|d) = q_d(t) &= n(d,t)/N(d) && \text{on } \mathcal{T} \end{aligned}$$

The notation $Q_t$ or $Q_t(d)$ for the *source distribution of* $t$ emphasizes that it is the distribution of source documents of a fixed term $t$, whereas the notation $Q(d|t)$ emphasizes the interpretation as the conditional probability that a term occurrence has source $d$ given that the term is $t$. Likewise the notations $q_d$ and $q_d(t)$ for the *term distribution of* $d$ emphasize that it is the distribution of terms in a fixed document $d$, whereas $q(t|d)$ emphasizes the interpretation as the probability of an occurrence of term $t$ given that the source is $d$. Other probability distributions on $\mathcal{C}$ and $\mathcal{T}$ will be denoted by $P$ and $p$ with various sub and superscripts .

### B. Distribution of Co-occurring Terms

Consider a Markov chain on $\mathcal{T} \cup \mathcal{C}$ having only transitions $\mathcal{C} \to \mathcal{T}$ with transition probabilities $Q(d|t)$ and transitions $\mathcal{T} \to \mathcal{C}$ with transition probabilities $q(t|d)$. It allows us to propagate probability distributions from terms to document and vice versa.

Given a term distribution $p(t)$, the one step Markov chain evolution gives us a document distribution $P_p(d)$. This is the probability to find a term occurrence in a particular document given that the term distribution of the occurrences is $p$

$$P_p(d) = \sum_t Q(d|t)p(t).$$

Likewise, the one step Markov evolution of a document distribution $P(d)$ is the term distribution $p_P(t) = \sum_d q(t|d)P(d)$. Since $P(d)$ is the probability to find a term occurrence in document $d$, $p_P$ is the P-weighted average of the term distributions in the documents. Combining these, i.e. running the Markov chain twice, every term distribution gives rise to a new term distribution

$$\bar{p}(t) = p_{P_p}(t) = \sum_d q(t|d)P_p(d) = \sum_{t',d} q(t|d)Q(d|t')p(t')$$

In particular, starting from the degenerate "known to be $z$" term distribution $p_z(t) = p(t|z) = \delta_{tz}$ (1 if $t = z$ and 0 otherwise), we get the *distribution of co-occurring terms* or *co-occurrence distribution* $\bar{p}_z$

$$\bar{p}_z(t) = \sum_{d,t'} q(t|d)Q(d|t')p_z(t') = \sum_d q(t|d)Q(d|z).$$

This distribution is the weighted average of the term distributions of documents containing $z$ with weight the probability $Q(d|z)$ that an instance of term $z$ has source $d$.

If we run the Markov chain twice on the document distribution $q_d(t)$ we get the weighted sum of the co-occurrence distributions by linearity:

$$\bar{q}_d(t) = \sum_{d',t'} q(t|d')Q(d'|t')q(t'|d) = \sum_z q(z|d)\bar{p}_z(t).$$

The distribution $\bar{q}_d$ can be seen as a smoothed version of $q_d$.

If two terms have similar co-occurrence distributions, i.e. if they occur in the same contexts, words are arguable closely related, usually semantically. The probability measure $\bar{p}_z$ is similar to the setup in [16, section 3] to detect semantic similarity. However it is more refined because we keep track of the density of a keyword in a document rather than the mere occurrence or non occurrence of a keyword in a document.

### C. Finding Keywords by Comparing Word Distributions

We follow the common idea that good keywords have two properties: they have importance for the document and they have discriminative power in the whole collection. These are exactly the two components of the *tf.idf* measure. For both criteria we aim to find an alternative measure that takes the relations between words into account, and it is for this purpose that we introduced the co-occurrence distribution.

To express the relevance of a term for a document we have several possibilities. We can compare the co-occurrence distribution of the term with either the term distribution $q_d$ or with its smoothed variant $\bar{q}_d$. In addition we have various possibilities for expressing the similarity between distributions.

Discriminative power can also be expressed with help of co-occurrence distributions: a term is likely to be less discriminative if its co-occurrence distribution is more similar to the background distribution $q$. Terms $z$, for which $\bar{p}_z$ has a large divergence from $q$, tend to be more specific and have higher discriminative power.

Various divergences are natural (dis)similarity measures for distributions. The Kullback-Leibler divergence of probability distributions $p$ and $q$ is defined as

$$D(p||q) = \sum_{x \in X} p(x) \log\left(\frac{p(x)}{q(x)}\right)$$

if $q(x) = 0$ implies $p(x) = 0$ and $\infty$ otherwise. Note that $D(p||q)$ is asymmetrical. The Jenson-Shannon divergence of $p$ and $q$ is defined as: $JSD(p|q) = \frac{1}{2}D(p||m) + \frac{1}{2}D(q||m)$ where $m = \frac{1}{2}p + \frac{1}{2}q$. It is symmetrical, always finite and nonnegative. We refer to [19, sec. 2.3] for details. In previous work ([17], [18]) the divergence of co-occurrence distributions outperformed direct methods to measure the co-occurrence like cosine similarity.

To compare the co-occurrence distribution $\bar{p}_z$ of a term $z$ with the word distribution $q_d$ of a text $d$ one cannot use the Kullback-Leibler divergence $D(\bar{p}_z||q_d)$ since typically $D(\bar{p}_z||q_d) = \infty$. In a first approach, we therefore rank words from the text by minimizing

$$JSD(\bar{p}_z||q_d). \tag{1}$$

Though ranking words according to their Jenson-Shannon divergence to the word distribution of a document gives acceptable keywords, the results are worse than those obtained by simply using *tf.idf*. To improve the results we also have to take the discriminative power of the keywords into account. The obvious way to do this is to compute the divergence of the the co-occurrence distribution $\bar{p}_z$ with the background distribution $q$. If the term $z$ occurs with equal probability in all sorts of texts this divergence will tend to be small. However, if $z$ occurs only in documents on one topic, it will diverge strongly from $q$. Thus, we have to balance minimizing the divergence to $q_d$ and maximizing the divergence to $q$. We therefore maximize the difference

$$\Delta_{z,d} = D(\bar{p}_z||q) - D(\bar{p}_z||\bar{q}_d) = \sum_t \bar{p}_z(t) \log\left(\frac{\bar{q}_d(t)}{q(t)}\right). \tag{2}$$

This last formula can be computed very efficiently for ranking words for a text, since the logarithm is independent of $z$ and only has to be computed once. In this case results using JS-divergence and KL-divergence are similar. Results improved significantly by using $\bar{q}_d$ instead of $q_d$. By comparing $q_z$ with $\bar{q}_d$ we do not test whether $d$ is a typical context for $z$ but whether the words in $d$ are related to $z$.

In (2) we effectively try to match peaks in $\bar{q}_d$ with peaks in $\bar{p}_z$, where peaks in the first case are relative to the background distribution. This gives rise to an alternative in which we do not try to match $\bar{p}_z$ and $\bar{q}_d/q$, but the absolute differences $\bar{p}_z - q$ and $\bar{p}_d - q$. This idea can be expressed by a correlation coefficient that can be understood in the following way. First we move our co-coordinate system such that the back ground distribution is in the origin. Then we compute the cosine between the document and the keyword vectors.

$$r(z,d) = \frac{\sum_t(\bar{q}_d(t) - q(t))(\bar{p}_z(t) - q(t))}{\sqrt{\sum_t(\bar{q}_d(t) - q(t))^2}\sqrt{\sum_t(\bar{p}_z(t) - q(t))^2}}. \tag{3}$$

## IV. Preprocessing

The quality of the keyword extraction procedure depends highly on the ability to determine the right candidate terms. We implemented both preprocessing and keyword ranking in the UIMA framework [20], [21]. Lemmatization and part-of-speech tagging were delegated to the Tree Tagger [22].

As discussed above, we treat a document as a probability distribution over words. However, since we are interested in the topics of the text rather than the linguistic or stylistic properties, we restrict the distribution to the open class words, i.e. nouns, adjectives, verbs (excluding auxiliary verbs), proper names and adverbs. Moreover, words are reduced to their lemma i.e. their canonical lexical form (not to their stem or root) to compute the term distribution of the document. Finally, to make computations more efficient, we reduce the set of lemmas to those occurring at least 5 times in the corpus. In all distributions used, these words will have very low probabilities and will not contribute much to the divergences.

Our preprocessing also includes a multiword detection and a detection of names of persons and companies. The latter is is done using some lists of frequent names and heuristic rules. Multiwords are detected roughly following the approach from [23]. Though this might not be the optimal approach for our purpose, the results seem to be good enough to generate a candidate term set to compare different ranking algorithms.

Not all words are suited as keyword. In particular, adverbs and adjectives are not commonly used as keywords. In addition, proper names that are important for a document usually are added to the meta-data of the document, but not as keyword. Thus we restrict the set of possible keywords to the nouns and verbs occurring in the text. We stress that for the distributions representing a document or term we *do* take all open class words into account.

## V. Evaluation

We use two evaluation sets: a collection of publicly available abstracts of computer science article published by the ACM and a collection of synopses of BBC broadcasts.

The collection of ACM abstracts consists of 10934 texts[1]. For each of the articles keywords are available. There are 27336 distinct keywords, 21634 of which are used in the collection only once, 2 keywords (*evaluation* and *security*) are used more than 100 times. The great majority of all keywords, 21642, consists of more than one word. Our multiword detection algorithm identified 4817 multiwords. Each article has at least 1 and at most 10 keywords, with an average of 4.5 keywords. We consider this set of keyword annotations as a golden standard and evaluate extracted keywords by computing precision and recall for this set. Obviously, many terms which are not selected by the authors or editors of an article might be good keywords nevertheless. About 52% of the articles has a keyword that is selected as a candidate term after preprocessing. Thus the theoretically optimal precision if only one keyword is selected is 0.52.

The collection of synopses from BBC television broadcasts is not publicly available and was kindly provided by BBC Research. The collection consists of the synopses of 2879 programs, and is quite different in nature from the ACM collection. Many of the synopses are very short and say virtually nothing about the actual content of the broadcast (e.g. "Dominic Arkwright chairs the discussion programme."). Moreover, for many broadcasts that are episodes from a series the synopsis consists of a general part that is identical for all episodes of the series with a small episode specific part. There are 1748 distinct keywords, 898 of which are only used once in the collection. There were 8 keywords used more than a 100 times, and 792 of the keywords consist of more than one word. The multiword detection algorithm found 168 multiwords. Each article has at least 1 and at most 22 keywords. 915 programs have only 1 keyword. On average, each article has $2,9$ keywords. About 57% of the articles has a keyword that is selected as a candidate term after preprocessing.

Though these data sets are not very large, it has to be noted that most data sets used for keyword evaluation are much smaller. We observed that results can be completely different when using subsets with less than 1000 documents.

### A. Comparison procedure

Evaluation of keyword extraction algorithms is methodologically somewhat problematic ([24]): only about a half of the keywords that are assigned to the documents can be extracted from those texts. Moreover, it is known that the inter documentalists agreement on keywords is usually not very high. This means that keywords can be perfect keywords, but that they are not actually assigned to the document.

Notwithstanding these general reservations we will use precision and recall against manually assigned keywords as no better possibilities are available to evaluate large volumes of keyword assignments. Moreover, every algorithm that we want to evaluate suffers largely from the same problems, which makes the comparison between different statistical extraction methods possible to some degree.

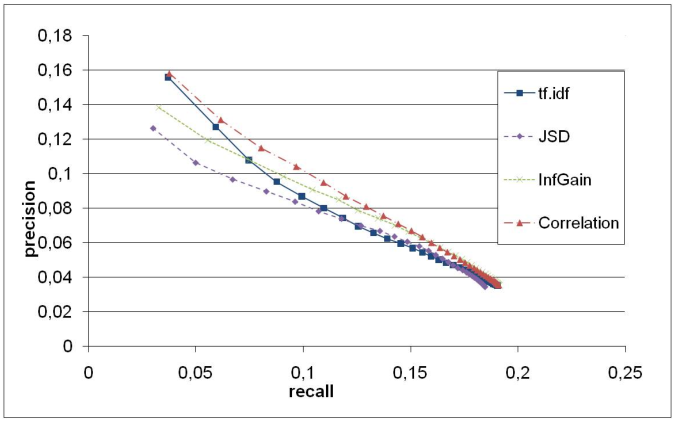

Fig. 1. Precision and recall for 4 algorithms with the ACM data set

We compare the following methods: *tf.idf* as the baseline, and the methods described in equations (1), (2), and (3). The *tf.idf* value for a term $t$ in document $d$ is computed as: $tf.idf(t,d) = n(d,t)\frac{N}{\log df(t)}$ where $df$ is the number of documents $d'$ for which $n(d',t) > 0$.

For each ranking method and for each abstract at most 34 keywords are generated. Precision and recall are computed for the top $n$ keywords, with $0 < n < 35$. Since our candidate terms are always lemmas and classical keywords are usually plural nouns and gerunds, we consider singular forms of the plural in the reference set as a true positive. The difference between infinitives and gerunds is treated likewise.

### B. Results

Figures 1 and Figure 2 give precision and recall for each of the four algorithms for each cut-off value between 0 and 35. Table I gives the exact numbers for precision and recall for the top 5 and top 10 set of keywords for the baseline *tf.idf* and the best of the proposed algorithms. Given the completely different nature of both datasets the similarity of the picture for the datasets is striking. For the ACM data we see that the proposed correlation of the co-occurrence distributions is clearly better than *tf.idf*. In the case of the BBC data the improvement is only minor and for the first three keywords performance is even worse than *tf-idf*.

A reason for the difference in behavior between the two datasets could be that the BBC dataset is much smaller. Therefore the co-occurrence statistics might not be as good as for the larger ACM dataset. Another likely reason could be the fact that the BBC keywords were assigned manually, but that annotators received suggestions for keywords that are automatically generated from the synopsis. The exact algorithm used for those suggestions is unknown to us, but given the popularity of *tf.idf* it is likely that this measure was used. This would have the effect that the selection of keywords is biased towards words with high *tf.idf* values.

The results show that the best method is the correlation coefficient given by (3). This eventually could be explained

[1] The URLs of the used abstracts can be downloaded from *https://doc.novay.nl/dsweb/Get/Document-115737/ACM-URLs.txt*

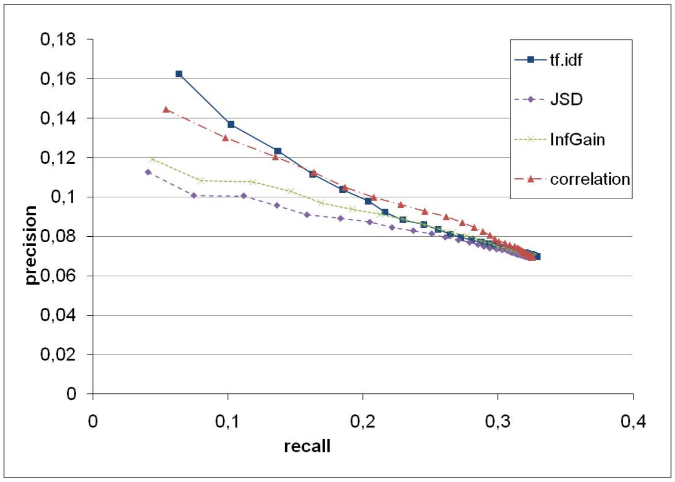

Fig. 2. Precision and recall for 4 algorithms with the BBC data set

by the fact that humans are more sensible to deviations of the average than to information gain.

## VI. Conclusion

We presented three statistical methods to improve keyword extraction that go beyond the use of *tf-idf*. All three methods try to implement the *tf-idf* strategy of balancing the relevance for the text with discriminative power. Unlike the classical *tf-idf* measure however, we take the relations and context of words into account by using the so called co-occurrence distribution. For the task of ranking keyword candidates, this leads to an improvement over tf-idf based ranking when evaluated with two substantially different human annotated datasets: ACM abstracts and BBC synopses. A correlation coefficient between co-occurrence distributions outperformed a more principled information theoretic approach.

## Acknowledgments

The work described in this paper was part of the MyMedia project and has received funding from the European Community's Seventh Framework Program under grant agreement n$^o$ 215006. We would like to thank our project partners from BBC-research for providing a collection of synopses.

|  | Precision | Recall | F1 |
|---|---|---|---|
| BBC-tf.idf-top5 | 0.10 | 0.18 | 0.13 |
| BBC-correl.-top5 | 0.10 | 0.19 | 0.13 |
| Improvement | 1.2% | 1.6% | 1.3% |
| BBC-tf.idf-top10 | 0.084 | 0.26 | 0.13 |
| BBC-correl.-top10 | 0.087 | 0.27 | 0.13 |
| Improvement | 4.2% | 7.0% | 4.9% |
| ACM-tf.idf-top5 | 0.087 | 0.099 | 0.093 |
| ACM-correl.-top5 | 0.095 | 0.11 | 0.10 |
| Improvement | 9.2% | 10% | 9.7% |
| ACM-tf.idf-top10 | 0.062 | 0.14 | 0.086 |
| ACM-correl.-top10 | 0.067 | 0.15 | 0.092 |
| Improvement | 6.9% | 8.0% | 7.3% |

TABLE I
Precision, recall and F-measure for top 5 and top 10 keywords: absolute values and improvement in percents.

## References

[1] G. Salton and C. Buckley, "Term weighting approaches in automatic text retrieval." Cornell University, Tech. Rep., 1987. [Online]. Available: http://hdl.handle.net/1813/6721

[2] K. Spärck Jones, "A statistical interpretation of term specificity and its application in retrieval," *Journal of documentation*, vol. 60, pp. 493–502, 2004.

[3] S. Robertson and K. Jones, "Relevance weighting of search terms," *Journal of the American Society for Information Science*, vol. 27, no. 3, 1976.

[4] S. Dumais, "Improving the retrieval of information from external sources," *Behavior Research Methods, Instruments and Computers*, vol. 23, no. 2, pp. 229–236, 1991.

[5] W. Greiff, "A theory of term weighting based on exploratory data analysis," in *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM New York, NY, USA, 1998, pp. 11–19.

[6] D. Hiemstra, "A probabilistic justification for using tf× idf term weighting in information retrieval," *International Journal on Digital Libraries*, vol. 3, no. 2, pp. 131–139, 2000.

[7] A. N. Aizawa, "An information-theoretic perspective of tf-idf measures," *Inf. Process. Manage.*, vol. 39, no. 1, pp. 45–65, 2003.

[8] E. Frank, G. W. Paynter, I. H. Witten, C. Gutwin, and C. G. Nevill-Manning, "Domain-specific keyphrase extraction," in *IJCAI*, T. Dean, Ed. Morgan Kaufmann, 1999, pp. 668–673.

[9] P. D. Turney, "Learning algorithms for keyphrase extraction," *Inf. Retr.*, vol. 2, no. 4, pp. 303–336, 2000.

[10] J. Tang, J.-Z. Li, K. Wang, and Y.-R. Cai, "Loss minimization based keyword distillation," in *APWeb*, ser. Lecture Notes in Computer Science, J. X. Yu, X. Lin, H. Lu, and Y. Zhang, Eds., vol. 3007. Springer, 2004, pp. 572–577.

[11] K. Zhang, H. Xu, J. Tang, and J.-Z. Li, "Keyword extraction using support vector machine," in *WAIM*, ser. Lecture Notes in Computer Science, J. X. Yu, M. Kitsuregawa, and H. V. Leong, Eds., vol. 4016. Springer, 2006, pp. 85–96.

[12] R. Mihalcea and P. Tarau, "TextRank: Bringing order into texts," in *Proceedings of EMNLP*, vol. 4. Barcelona: ACL, 2004, pp. 404–411.

[13] Y. Matsuo and M. Ishizuka, "Keyword extraction from a single document using word co-occurrence statistical information," in *FLAIRS Conference*, I. Russell and S. M. Haller, Eds. AAAI Press, 2003, pp. 392–396.

[14] Z. Harris, *Mathematical structures of language*. Wiley, 1968.

[15] K. Lindén and J. Piitulainen, "Discovering synonyms and other related words," *CompuTerm 2004*, pp. 63–70, 2004.

[16] H. Li and K. Yamanishi, "Topic analysis using a finite mixture model," *Inf. Process. Manage.*, vol. 39, no. 4, pp. 521–541, 2003.

[17] C. Wartena and R. Brussee, "Instance-based mapping between thesauri and folksonomies," in *International Semantic Web Conference*, ser. Lecture Notes in Computer Science, A. P. Sheth, S. Staab, M. Dean, M. Paolucci, D. Maynard, T. W. Finin, and K. Thirunarayan, Eds., vol. 5318. Springer, 2008, pp. 356–370.

[18] ——, "Topic detection by clustering keywords," in *DEXA Workshops*. IEEE Computer Society, 2008, pp. 54–58.

[19] T. Cover and J. Thomas, *Elements of information theory*. John Wiley and Sons, Inc., 1991.

[20] D. Ferrucci and A. Lally, "UIMA: an architectural approach to unstructured information processing in the corporate research environment," *Natural Language Engineering*, vol. 10, no. 3-4, pp. 327–348, 2004.

[21] "Apache uima," http://incubator.apache.org/uima/.

[22] H. Schmid, "Probabilistic part-of-speech tagging using decision trees," in *Proceedings of International Conference on New Methods in Language Processing*, vol. 12. Manchester, UK, 1994.

[23] J. Justeson and S. Katz, "Technical terminology: some linguistic properties and an algorithm for identification in text," *Natural language engineering*, vol. 1, no. 1, pp. 9–27, 1995.

[24] L. Gazendam, C. Wartena, V. Malaise, G. Schreiber, A. de Jong, and H. Brugman, "Automatic Annotation Suggestions for Audiovisual Archives: Evaluation Aspects," *Interdisciplinary Science Reviews, 34*, vol. 2, no. 3, pp. 172–188, 2009.