KEYSTRACT: Single document keyphrase extraction using sentence clustering and Latent Dirichlet Allocation

Claude Pasquier
Institute of Developmental Biology & Cancer
University of Nice Sophia-Antipolis
UNSA/CNRS UMR-6543
Parc Valrose
06108 NICE Cedex 2, France
claude.pasquier@unice.fr

Abstract

This paper describes the design of KEYSTRACT, a system designed for keyphrase extraction from a single document. The principle of the algorithm is to cluster sentences of the documents in order to highlight parts of text that are semantically related. The clusters of sentences, that reflect the themes of the document, are then analyzed to find the main topics of the text. Finally, the most important words, or groups of words, from these topics are proposed as keyphrases. This method is evaluated on task number 5 (*Automatic Keyphrase Extraction from Scientific Articles*) of SemEval-2010: the 5th International Workshop on Semantic Evaluations.

1 Introduction

Keyphrases are words, or groups of words, that capture the key ideas of a document. They represent important information concerning a document and constitute an alternative, or a complement, to full-text indexing. Pertinent keyphrases are also useful to potential readers who can have a quick overview of the content of a document and can select easily which document to read. Usually, authors of academic papers are requested to give a set of keyphrases with the document. However, this manual annotation may be subjective and may not correspond to the keyphrases selected by independent readers. In addition, numerous documents that are available electronically do not have author assigned keyphrases.

Currently, the most powerful keyphrases extraction algorithms are based on supervised learning. These methods address the problem of associating keyphrases to documents as a classification task. However, the fact that this approach requires a corpus of similar documents, which is not always readily available, constitutes a major drawback. For example, if one encounters a new Web page, one might like to know quickly the main topics addressed. In this case, a domain-independent keyword extraction system that applies to a single document is needed.

The paper describes our keyphrase extraction algorithm from a single document. We show that our system performs well without the need for a corpus.

The paper is organized as follows. The next section describes the principles of our keyphrase extraction system. We present the main parts of the algorithm in section 3 and detail the methods in section 4. Finally, we evaluate the system's performance in section 5 and we conclude the paper.

2 Principles

When an author writes a document, he has to think first at the way he will present his ideas. Most of the time, he establishes a content summary that highlight the main topics of his text. Then he writes the content of the document by carefully selecting the most appropriate words to describe each topic. In this paper, we make the assumption that the words, or the set of words, that are representative of each topic constitute the keyphrases of the document. In the following of this paper, we call *terms*, the components of a documents that constitutes the vocabulary (see the detail of the identification of terms in subsection 4.3).

In statistical natural language processing, one common way of modeling the contributions of different topics to a document is to treat each topic as a probability distribution over words. Therefore, a document is considered as a probabilistic mixture of these topics (Griffiths and Steyvers, 2004).

Generative models can be used to relate a set of observations (in our case, the terms used in a document) to a set of latent variables (the topics). A particular generative model, which is well suited for the modeling of text, is called Latent Dirichlet Allocation (LDA) (Blei et al., 2003). LDA is mostly used to analyze text corpora. Given a set of documents, the algorithms describes each document as a mixture over topics, where each topic is characterized by a distribution over words.

The idea of KEYSTRACT is to perform first a clustering of the sentences of the document based on their semantic similarity. Intuitively, one can see each cluster as a part of the text dealing with semantically related content. Therefore, the initial document is divided into a set of clusters and LDA can then be applied on this new representation.

3 Algorithm

The algorithm is composed of 8 steps:

1. Identification and expansion of abbreviations.
2. Splitting the content of the document into m sentences.
3. Identification of the n unique terms in the document that are potential keyphrases.
4. Creation of a $m \times n$ sentence-term matrix X to identify the occurrences of the n terms within a collection of m sentences.
5. Dimension reduction to transform data in the high-dimensional matrix X to a space of fewer dimensions.
6. Data clustering performed in the reduced space. The result of the clustering is used to build a new representation of the source document, which is now considered as a set of clusters, with each cluster consisting of a bag of terms.
7. Execution of LDA on the new document representation.
8. Selection of best keyphrases by analyzing LDA's results.

4 Methods

KEYSTRACT is build on UIMA (Unstructured Information Management Architecture) (*http://incubator.apache.org/uima/*), a robust and flexible framework that facilitates interoperability between tools. The method process one document at a time by performing the steps described below.

4.1 abbreviations expansion

The program *ExtractAbbrev* (Schwartz and Hearst, 2003) is used to identify abbreviations (short forms) and their corresponding definitions (long forms). Once abbreviations have been identified, each short form is replaced by its corresponding long form in the processed document.

4.2 sentences detection

Splitting the content of a document into sentences is an important step of the method. To perform this task, we used the OpenNLP's sentence detector module (*http://opennlp.sourceforge.net/*) trained on a corpus of general english texts.

4.3 terms identification

Word categories are identified by using the LingPipe's general english part-of-speech (POS) tagger trained on the Brown Corpus (*http://alias-i.com/lingpipe/*). We leverage POS information to collect, for each sentence, nominal groups that are potential keyphrases. We use a very simple rule to identify nominal groups: a nominal group is an adjacent list of words tagged as determiner, adjective, noun, pronoun, gerund, present participle, past participle or adverb. These nominal groups represent the vocabulary of the document. They are called *terms* in this paper.

4.4 matrix creation

Let $D = \{d_1, d_2, \ldots, d_n\}$ be the complete vocabulary set of the document identified in subsection 4.3 above. We build a $m \times n$ matrix $X = [x_{ik}]$ where m is the number of sentences in the document, n is the number of terms and x_{ij} is the weight of the $j-th$ term in the $i-th$ sentence. The weight of a term in a sentence is the product of a local and global weight given by $x_{ij} = l_{ij} \times g_j$, where l_{ij} is the local weight of term j within sentence i, and g_j is the global weight of term j in the document. The local weighting function measures the importance of a term within a sentence and the global weighting function measures the importance of a term across the entire document. Three local weighting functions were investigated: term frequency, log of term frequency and binary. Five global weighting functions were also investigated: Normal, GFIDF (Global frequency $\times$ Inverse document frequency, IDF (Inverse document frequency), Entropy and none (details of calculation can be found in Dumais (1991) paper).

4.5 dimension reduction

The matrix X is a representation of a document in a high-dimensional space. Singular Value Decomposition (SVD) (Forsythe et al., 1977) and Non-Negative Matrix Factorization (NMF) (Lee and Seung, 1999) are two matrix decomposition techniques that can be used to transform data in the high-dimensional space to a space of fewer dimensions.

With SVD, the original matrix X is decomposed as a factor of three other matrices U, Σ and V such as:

$$X = U\Sigma V^T$$

where U is an $m \times m$ matrix, Σ is a $m \times n$ diagonal matrix with nonnegative real numbers on the diagonal, and V^T denotes the transpose of V, an $n \times n$ matrix. It is often useful to approximate X using only r singular values (with $r < min(m, n)$), so that we have $X = U_r \Sigma_k V_r^T + E$, where E is an error or residual matrix, U_r is an $m \times r$ matrix, Σ_r is a $k \times r$ diagonal matrix, and V_r is an $n \times r$ matrix.

NMF is a matrix factorization algorithm that decomposes a matrix with only positive elements into two positive elements matrices, with $X = WH + E$. Usually, only r components are fit, so E is an error or residual matrix, W is a non-negative $m \times r$ matrix and H is a non-negative $r \times n$ matrix. There are several ways in which W and H may be found. In our system, we use Lee and Seung's multiplicative update method (Lee and Seung, 1999).

All matrix calculations are executed on the R software environment (*http://cran.r-project.org/*). The SVD decomposition of the matrix is computed by the base function $svd()$ and the method $nnmf()$ from package NMFN (*http://cran.r-project.org/web/packages/NMFN/*) is used for NMF decomposition.

4.6 sentences clustering

The clustering of sentences is performed in the reduced space by using the cosine similarity between sentence vectors. Several clustering techniques have been investigated: k-means clustering, Markov Cluster Process (MCL) (Dongen, 2008) and ClassDens (Guénoche, 2004).

However, while the latent semantic space derived by SVD does not provide a direct indication of the data partitions, with NMF, the cluster membership of each document can be easily identified, directly on the W matrix (Xu et al., 2003). Each value w_{ij} of matrix W, indicates, indeed, to which degree sentence i is associated with cluster j. If NMF was calculated with the rank r, then r clusters are represented on the matrix. We use a simple rule to determine the content of each cluster: sentence i belongs to cluster j if $w_{ij} > a \max_{k \in \{1...m\}} w_{ik}$. In our system, we fixed $a = 0.1$.

4.7 applying LDA on the new document representation

By using the result of the clustering, the source document is now represented by c clusters of terms. The terms associated with a cluster c_i is the sum of the terms belonging to all the sentences in the cluster. JGibbLDA (*http://jgibblda.sourceforge.net/*) is used to execute LDA on the new dataset. We tried to extract different numbers of topics t (with $t \in \{2, 5, 10, 20, 50, 100\}$) and we choose the Dirichlet hyperparameters such as $\alpha = 0.1$ and $\beta = 50/t$. LDA inferences a topic model by estimating the cluster-topic distribution Θ and the topic-word distribution Φ (Griffiths and Steyvers, 2004; Blei et al., 2003).

4.8 terms ranking and keyphrases selection

We assume that topics covering a significant portion of the document content are more important than those covering little content. To reflect this assumption, we calculate the importance of a term in the document (its score) with a function that takes into account the distribution of topics over clusters given by θ, the distribution of terms over topics given by Φ and the clusters' size.

$$score(i) = \max_{j \in \{1...n\}} (\Phi_{ji} \sum_{k=1}^{c} (\Theta_{kj} p(s(k)))$$

where $score(i)$ represents the score of term i and $s(k)$ is the size of the cluster k. We tested three different functions for p: the constant function $p(i) = 1$, the linear function $p(i) = i$ and the exponential function $p(i) = i^2$.

When a score is attributed to each term of the vocabulary, our system simply selects the top terms with the highest score and propose them as keyphrases.

5 Evaluation

We evaluated our system on the trial and train data provided in order to estimate the best options and parameters to use. Numerous parameters have influence on the method: the weighting of the terms in the document matrix, the dimension reduction method used, the number of dimension retained, the clustering algorithm, the number of topics used to execute LDA and the way best keyphrases are selected.

The parameter that most affects the performance is the method used to perform the dimension reduction. In all cases, whatever the other parameters, NMF performs better. We found that using only 10 components for the factorization is sufficient. There was no significant performance increase by using more factors.

The second more important parameter is the clustering method used. When NMF is used, the best results were achieved by retrieving clusters from the W matrix. With SVD, ClassDens gets the best results. We tested the performance of k-means clustering by specifying a number of clusters varying from 5 to 100. The best performances were achieved with a number of clusters ≥ 20. However, k-means scores a little bit below ClassDens and MCL is found to be the worst method.

The choice of the global weighting function is also important. In our experiments, the use of IDF and no global weighting gave the worst results. This confirms the conclusion stated by Lee et al. (2005). Entropy and normal weighting gave the best results but, in average, entropy performs a little better than normal weight. In the final version, the global weighting function used is entropy.

The last parameters that have a visible influence of the quality of extracted keyphrases is the selection of keyphrases from LDA's results. In our experiments, this is the exponential function that performs best.

The remaining parameters do not have notable influence on the results. As already stated by Lee et al. (2005), the choice of local weighting function makes relatively little difference. Similarly, the number of topics used for LDA has little influence. In KEYSTRACT we used term frequency as local weighting and executed LDA with a number of expected topics of 10.

6 Conclusion

The performance of our system are in the average of other submitted systems. The F-scores for the top 15 candidates over reader-assigned keyword is 0.1775 while the F-score over both author-assigned and reader-assigned keywords is 0.1852. On the basis of these two measures, the system ranks 13th and 10th respectively out of 20 submitted systems.

However, one has to note that KEYSTRACT uses only the information available from a single document. Therefore, the algorithm described here is an interesting alternative to supervised learning methods when no corpus of similar documents is available.

References

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022.

Stijn Van Dongen. 2008. Graph clustering via a discrete uncoupling process. *SIAM J. Matrix Anal. Appl.*, 30(1):121–141.

Susan T. Dumais. 1991. Improving the retrieval of information from external sources. *Behavior Research Methods, Instruments, & Comp.*, 23(2):229–236.

George Forsythe, Michael Malcolm, and Cleve Moler. 1977. *Computer Methods for Mathematical Computations*. Englewood Cliffs, NJ: Prentice Hall.

Thomas L. Griffiths and Mark Steyvers. 2004. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101:5228–5235, April.

Alain Guénoche. 2004. Clustering by vertex density in a graph. In *Classification, Clustering and Data Mining, D. Banks et al. (Eds.), Springer, 15-23.*

Daniel D. Lee and H. Sebastian Seung. 1999. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788, october.

Michael D. Lee, Brandon Pincombe, and Matthew Welsh. 2005. A comparison of machine measures of text document similarity with human judgments. In *proceedings of CogSci2005*, pages 1254–1259.

Ariel S. Schwartz and Marti A. Hearst. 2003. A simple algorithm for identifying abbreviation definitions in biomedical text. In *proceedings of PSB 2003*, pages 451–462.

Wei Xu, Xin Liu, and Yihong Gong. 2003. Document clustering based on non-negative matrix factorization. In *proceedings of SIGIR 03*, pages 267–273.