Topic models for comparative summarization

Michal Campr and Karel Jezek

Department of Computer Science and Engineering, FAV,
University of West Bohemia,
20 March 2013, 301 00, Plzen, Czech Republic
{mcampr,jezek_ka}@kiv.zcu.cz
http://textmining.zcu.cz

Abstract. This paper aims to sum up our work in the area of comparative summarization and to present our results. The focus of comparative summarization is the analysis of input documents and the creation of summaries which depict the most significant differences in them. We experiment with two well known methods – Latent Semantic Analysis and Latent Dirichlet Allocation – to obtain the latent topics of documents. These topics can be compared and thus we can learn the main factual differences and select the most significant sentences into the output summaries. Our algorithms are briefly explained in section 2 and their evaluation on the TAC 2011 dataset with the ROUGE toolkit is then presented in section 3.

Keywords: comparative summarization, latent semantic analysis, latent dirichlet allocation, topic model, rouge

1 Introduction

With the continual growth of the internet as an information source, where a great amount of data is being uploaded every minute, the need for data compression is obvious. This necessity is not important only for audio or video files, but also for textual documents. As the amount of textual data grows, the probability of appearance of documents with very similar features is increasing, e.g. political programs or descriptions of university courses. In any application, when facing a set of documents sharing a similar topic, people are interested to know what are their differences. This problem can be addressed by comparative summarization and it is the primary focus of this paper.

Several papers addressing the comparison of text mining methods [1] and the problem of document comparison have already been published (e.g. [2]), as well as papers covering the area of summarization using a variety of methods, e.g. discriminative sentence selection [3] or linear programming [4]. In this paper, we explore the possibilities of utilising two very well known methods for acquiring latent semantic topics of documents for comparing them, extracting the most characteristic sentences and forming summaries which depict the main differences of the documents. These two methods are Latent Semantic Analysis (LSA) and Latent Dirichlet Allocation (LDA). We have already published papers which discuss the algorithms that use LSA or LDA for basic summarization or for topic comparison so in this paper we focus solely on algorithms for comparative summarization and their evaluation.

2 Topic models for comparative summarization

This section briefly explains the use of LSA and LDA for comparative summarization. Both algorithms operate under the same assumptions and have several similar features.

The principle of comparative summarization is loosely based on update summarization (e.g. [5]) but with some important differences. Its goal is the comparison of two different documents or two sets of documents D_1 and D_2, where we do not assume any familiarity with any of the documents. We just assume, that those two sets of documents refer to a similar topic, but contain some different information about it. The aim is finding those differences.

2.1 Latent Semantic Analysis

LSA is an algebraic method which analyses relations between terms and sentences of a given document. For decomposing matrices, it uses Singular Value Decomposition (SVD) which is a numerical process used for data dimensionality reduction, classification, searching in documents and also for text summarization.

The algorithm starts with creating two matrices A_1 and A_2 for each of the document sets. Column vectors of matrix A_1 or A_2 contain term frequencies in the given sentences. However, both matrices must be created with the same term set (terms combined from both document sets) to avoid inconsistencies with lengths of singular vectors during their comparison. Matrix A_1 has $t \times s_1$ dimensions and matrix A_2 $t \times s_2$ dimensions, where t is the number of terms in both document sets, s_1 is the number of sentences in D_1 and s_2 is the number of sentences in D_2. The values are computed as $a_{ij} = L(t_{ij}) \cdot G(t_{ij})$, where $L(t_{ij})$ is a boolean value (1 if term i is present in sentence j, 0 otherwise) and $G(t_{ij})$ is the global weight for term i in the whole document:

$$G(t_{ij}) = 1 - \sum_j \frac{p_{ij} \log(p_{ij})}{\log(n)}, p_{ij} = \frac{t_{ij}}{g_i}, \qquad (1)$$

where t_{ij} is the frequency of term i in sentence j, g_i is the total number of times that term i occurs in the whole document and n is the number of sentences in the document.

The Singular Value Decomposition of A, constructed over a document with m terms and n sentences, is defined as $A = U\Sigma V^T$, where $U = [u_{ij}]$ is an $m \times n$ matrix and its column vectors are called left singular vectors. Σ is a square diagonal $n \times n$ matrix and contains the singular values. $V^T = [v_{ij}]$ is an $n \times n$ matrix and its columns are called right singular vectors. This decomposition provides latent semantic structure of the input document represented by A. This means, that it provides a decomposition of the document into n linearly independent vectors, which represent the main topics contained in the document. If a specific combination of terms is often present within a document, it is represented by one of the singular vectors. And furthermore, the singular values contained in the matrix Σ represent the significance of these topics. Matrix U provides mapping of terms into topics and V^T provides mapping of sentences into topics.

By applying SVD on both matrices A_1 and A_2 separately, we get matrices U_1 and U_2, Σ_1 and Σ_2, V_1^T and V_2^T, which provide the mapping of terms/sentences to topics contained in both document sets. We can then start comparing those topics in matrices U_1 and U_2.

For each topic (left singular vector) from U_2, we want to find the most similar topic in U_1. The redundancy between two vectors is computed as a cosine similarity:

$$red(t) = \frac{\sum_{j=1}^{m} U_1[j,i] * U_2[j,t]}{\sqrt{\sum_{j=1}^{m} U_1[j,i]^2} * \sqrt{\sum_{j=1}^{m} U_2[j,t]^2}}, \tag{2}$$

where t is the index of the topic from U_2, j is the index of topic from U_1. With redundancy computed, we can get the dissimilarity of the given topic as $dis(t) = 1 - red(t)$. We store the values $dis(t)$ in a diagonal matrix DS_1 (Dissimilarity Score) and create the final matrix $F_1 = DS_1 * \Sigma_2 * V_2^T$ which contains the dissimilarity, as well as the importance of individual topics mapped on sentences.

From matrix F_1, we can start selecting sentences into the final extract. This selection is based on finding the longest sentence vectors, i.e. the length s_r of a sentence r is defined as:

$$s_r = \sqrt{\sum_{i=1}^{t} F_1[r,i]^2}. \tag{3}$$

When a vector is selected, we need to make sure that a sentence with similar information will not be selected. We have tested three different solutions (the first provided the best results):

– Set values of the selected vector to 0. This is the simplest solution and it guarantees that once a sentence was selected, a similar one will not be selected again.
– Subtract the selected vector from matrix F. This removes the selected sentence (and information it contains) from the whole matrix.
– Use cosine similarity to detect possible resemblance between the candidate sentence and any of the already selected sentences. This does not make any alterations to matrix F.

This process is run in both directions, i.e. we create matrices F_1 and F_2 which contain the differences of topics in both document sets and search for the most suitable sentences until the resulting summary reaches a predefined length.

2.2 Latent Dirichlet Allocation

LDA [6] can be viewed as a model which breaks down the collection of documents into topics. The collection is represented as a mixture of documents, where the probability (importance) of the document D_B for the collection is denoted as $P(D_B)$. It also represents each document as a mixture of topics with the probability distribution representing the importance of *j-th* topic for the document D_B denoted as $P(T_j|D_B)$. The topics are then represented as a mixture of words with their probability representing the importance of the *i-th* word for the *j-th* topic (denoted as $P(W_i|T_j)$).

The first step of the algorithm is to load the input data from two document sets A and B. The important thing here is that from the perspective of LDA, we treat every sentence as one document. Before we run the Gibbs sampler (we used the JGibbLDA implementation [7]) to obtain the LDA distributions, we have to remove the stop-words and perform term lemmatization. This way we are sure that there are no words that carry no useful information.

The obtained topic-word distributions for each document set are stored in matrices T_A for the document set A and T_B for B, where rows represent topics and columns represent words. A very important aspect of saving the distributions into matrices is matching their dimensions, i.e. to include words that appear only in one set into both matrices (with zero probability).

After this, we can compute topic-sentence matrices U_A and U_B with sentence probabilities :

$$P(S_r|T_j) = \frac{\sum_{W_i \in S_r} P(W_i|T_j) * P(T_j|D_r)}{length(S_r)^l}, \tag{4}$$

where $l \in <0, 1>$ is an optional parameter to configure the handicap of long sentences. The row vectors of U_A and U_B represent topics and the columns are sentences. Next step includes creating a symmetrical diagonal matrix SIM which contains the similarities of topics from both sets. This is accomplished as follows:

$T_A = [T_{A1}, T_{A2}, ..., T_{An}]^T, T_B = [T_{B1}, T_{B2}, ..., T_{Bn}]^T$, where T_{Ai} and T_{Bi} are row vectors representing topics and n is the number of topics. For each T_{Ai} find red_i (redundancy of i-th topic) by computing the largest cosine similarity between T_{Ai} and T_{Bj}, where $j \in <1..n>$ and storing value $1 - red_i$ representing the dissimilarity of i-th topic into matrix SIM.

Finally, we construct matrices $F_A = SIM * U_A$ and $F_B = SIM^T * U_B$ combining the probabilities of sentences with the dissimilarity of topics. Then, it is a simple matter to find sentences with the best score and including them in the summary. For better results, it is essential to compare (using cosine similarity) the candidate sentence with already selected sentences to avoid information redundancy. If a sentence is selected, the respective vector in F_A or F_B is set to 0 in order to remove the information from the matrix.

The output of this algorithm consists of two summaries of predefined length depicting the most significant information in which the documents differ.

3 Evaluation

For the evaluation of our algorithms we used a very well known tool – ROUGE which stands for Recall-Oriented Understudy for Gisting Evaluation. This family of measures, which are based on the similarity of overlapping units such as n-grams [1], word sequences, and word pairs between the computer-generated summary and the ideal summaries (created by humans), was firstly introduced in 2003 [8]. The ROUGE scores have been widely used for the evaluation of summarization algorithms since then and so we have also decided to compute six different ROUGE scores:

- ROUGE-1, ROUGE-2, ROUGE-3 and ROUGE-4 are N-gram based metrics.
- ROUGE-W is a weighted longest common subsequence measure.
- ROUGE-SU4 is a bigram measure that enables at most 4 unigrams inside of bigram components to be skipped.

3.1 The experiment

Due to the lack of unified testing data for evaluating comparative summarization, we utilized data from the TAC 2011 conference and created our own dataset to find out if the proposed methods brings the expected results.

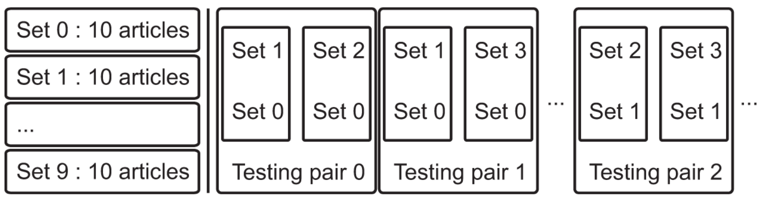

Fig. 1. Arrangement of testing data

The available data consist of 100 news articles, divided into 10 topics, 10 articles each. We have created pairs of sets of articles by combining different topics (Figure 1). In every pair, there is one identical topic present in both sets and one topic for each of the sets that is different. The purpose is to simulate two sets of documents which have something in common, but also some differences. This arrangement allows us also to easily compute the precision of selecting sentences because we know from which topic the sentences should be.

The reason for using the TAC 2011 dataset is also the fact, that it contains three human-created summaries for each of the 10 topics. This allows us to further evaluate our method with the ROUGE package.

[1] An n-gram is a subsequence of n words from a given text.

3.2 The results

With the use of the method, that was described in the previous section, we obtained 360 combinations of topics. Each of these combinations was then used as an input set of documents for our algorithms (producing two summaries for each input set), so in the end, we acquired 720 summaries. These summaries were then compared with the human-made summaries (from the TAC 2011 dataset) with ROUGE.

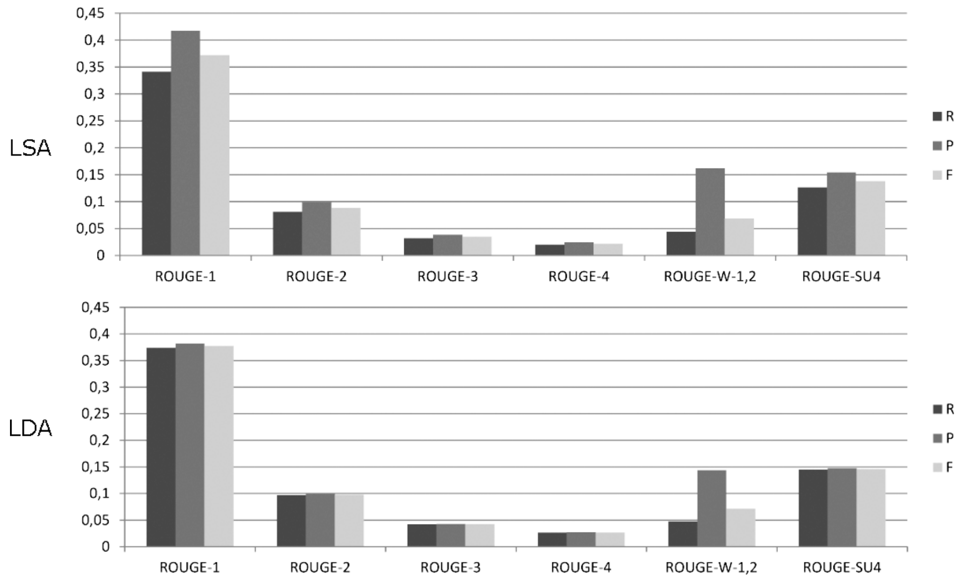

Fig. 2. ROUGE scores (P – precision, R – recall, F – F-score). LSA-based algorithm on the top, LDA-based algorithm on the bottom.

The Figure 2 shows the average scores for both our algorithms. Evidently, both techniques (LSA and LDA) show comparable results, however, LDA performs slightly better considering recall scores, but its precision is lower.

The results are also comparable with the ROUGE scores of other summarization methods where the recall scores ranged from 0.117 to 0.19 (i.e. [9]). It is important to note, that the used reference summaries were primarily meant to be used for basic summarization, however, due to the construction of our experiment, it is possible to use them for our evaluation. Because we extracted the information from a larger set of documents which contained "an interference" in the form of additional information, we obtained lower ROUGE scores, which was expected. The highest average ROUGE-2 recall score for LDA we achieved was 0.097 and 0.081 for LSA.

4 Conclusion and further research

This paper is focused on our experiments in the area of comparative summarization. We experimented with two well known topic models – LSA and LDA – from which the LDA provided better recall results but LSA higher precision. However, for summarizing very large data, it is worth also speculating about the aspect of computation time. LSA performs significantly faster for smaller data, but with the increasing length of documents, LSA starts to slow down due to the computations of large sparse matrices.

For full comparison of document content is necessary to consider not only its variety of topics, but also the polarity of sentences (their sentiment). Sentiment analysis and contrastive summarization will be the focus of our future work, i.e. comparing and summarizing the differences in opinions expressed in one or multiple documents like product reviews or political debates.

Acknowledgement

This project was supported by grant SGS-2013-029 Advanced computing and information systems.

The access to computing and storage facilities owned by parties and projects contributing to the National Grid Infrastructure MetaCentrum, provided under the programme ”Projects of Large Infrastructure for Research, Development, and Innovations” (LM2010005) is highly appreciated.

References

[1] Lee, Sangno and Baker, Jeff and Song, Jaeki and Wetherbe, James C. An Empirical Comparison of Four Text Mining Methods. *Proceedings of the 2010 43rd Hawaii International Conference on System Sciences*, pages 1–10, 2010.

[2] Gelbukh, Alexander F. and Sidorov, Grigori and Guzman-Arenas, Adolfo. Document Comparison with a Weighted Topic Hierarchy. *Proceedings of the 10th International Workshop on Database & Expert Systems Applications*, pages 566–, 1999.

[3] Wang, Dingding and Zhu, Shenghuo and Li, Tao and Gong, Yihong. Comparative document summarization via discriminative sentence selection. *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 1963–1966, 2009.

[4] Huang, Xiaojiang and Wan, Xiaojun and Xiao, Jianguo. Comparative news summarization using linear programming. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers - Volume 2*, pages 648–653, 2011.

[5] Steinberger, Josef and Jezek, Karel. Update Summarization Based on Latent Semantic Analysis. *In Text, Speech and Dialogue*, pages 77-84, Berlin: Springer, 2009.

[6] DM Blei, AY Ng, and MI Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, pages 993–1022, 2003.

[7] Xuan-Hieu Phan, Cam-Tu Nguyen. `http://jgibblda.sourceforge.net/`.

[8] Lin, Chin-Yew and Hovy, Eduard. Automatic evaluation of summaries using N-gram co-occurrence statistics. *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 71–78, 2003.

[9] Steinberger, Josef and Jezek, Karel. Update summarization based on novel topic distribution. *Proceedings of the 9th ACM symposium on Document engineering*, pages 205–213, Munich, Germany, 2009.