Study on Methods for Vector Representation of Text for Topic-based Clustering of News Articles

Michal Rott, Petr Červa

Technical University of Liberec
Studentsk 1402/2, 461 17 Liberec 1, Czech Republic
{michal.rott, petr.cerva}@tul.cz

Abstract

This paper deals with methods for vector representation of a text within the task of automatic topic-based clustering of documents. For this purpose, three different methods are evaluated experimentally using two types of test sets compiled from Czech newspaper articles. The first adopted clustering scheme utilizes conventional Latent Semantic Analysis (LSA). In contrast, both the remaining approaches are based on novel techniques such as Random Manhattan Indexing (RMI) or Skip-Gram Model (SGM). Obtained experimental results show that the latter methods lead to better results than LSA and that the best clustering accuracy can be reached by SGM. On the other hand, RMI yields just slightly worse results than SGM and has one important advantage for practical usage: it can handle Out Of Vocabulary (OOV) words without the need for re-training of a log-linear neural network.

1. Introduction

Since the 80's, there has been a strong effort to propose an efficient method for learning vector representations of words that would allow to improve accuracy of machine learning algorithms dealing with unstructured text data in various applications including automatic summarization, machine translation, topic detection, etc.

The main goal of this paper is to find out an approach for vector representation that would be optimal for topic-based clustering of newspaper articles. For this purpose, three different methods are adopted and evaluated experimentally.

The first baseline method is the well-know Latent Semantic Analysis (LSA) (Gong and Liu, 2001), which utilizes a term-document occurrence matrix to create the vector space of the input documents. The main disadvantage of this approach is that the produced vectors are high-dimensional and sparse. As shown in section 4., this fact limits the accuracy of LSA for document clustering even if a dimensionality reduction technique is utilized.

The two remaining methods have recently been proposed. The first method is Random Manhattan Indexing (RMI) (Zadeh and Handschuh, 2014). This approach is based on sparse Cauchy random projections and its advantage is that it combines the construction of a vector space model and dimension reduction into an incremental and fast procedure. The second novel method is the Skip-Gram Model (SGM), introduced in (Mikolov et al., 2013). SGM is based on a log-linear neural network which is trained in a way that each word from the training text corpus forms an input to the network with a projection layer and predict surrounding words.

In contrast to RMI, which utilizes the principle of random vector generation, the vector representations created by SGM allow to capture syntactic and semantic word relationships. On the other hand, when a new word occurs in the input document being clustered, it is necessary to perform retraining of a log-linear neural network. Unfortunately, this process is time-demanding when compared to a time needed to generate a new vector within RMI.

The rest of the paper is structured as follows: The next section describes the clustering scheme adopted within this paper and basic theoretical concepts of individual methods for vector representation of a text. Section 3. then describes experimental setup and the data used for training. Description of test data and results of performed experiments are presented in the next section 4. Finally, section 5. concludes this paper.

2. Document Clustering Process

Usually, the process of document clustering consists of three basic consecutive steps as follows:

1. Pre-processing of input documents.
2. Vector representation of text.
3. Application of a selected clustering algorithm

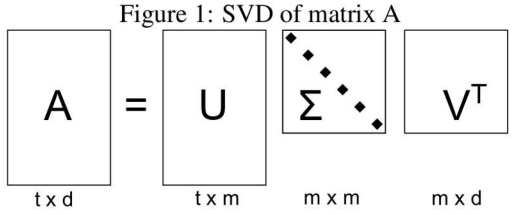
The next subsections describe how these steps were performed within this work and which methods were adopted for this purpose.

2.1. Pre-processing of Input Documents

The main goal of the pre-processing phase is to a) to strip all formatting from the input text including capitalization, punctuation, etc, and b) to lower the dimensionality before vector representation of text is performed within the next phase.

For this purpose, we utilize a pre-processing module developed originally for the automatic summarization engine of Czech (Rott and Cerva, 2013). This module performs several consecutive operations such as removing of words from a stop list, sentence separation and lemmatization of words. Note that the stop list contained over 200 Czech terms and that separation of sentences was done

Figure 1: SVD of matrix A



using a sequence of regular expressions designed to follow Czech language grammar. The external tool Morphodita (Straková et al., 2014) was used to lemmatize the words of the input documents.

2.2. Vector Representation

In this work, three different approaches were utilized for vector representation of the input text. These methods are described in more detail in the following subsections:

Latent Semantic Analysis

LSA is inspired by latent semantic indexing and uses a term-document occurrence matrix to create the vector space of the given documents. The term-document matrix $\boldsymbol{A}$ is constructed from weighted term frequency values. The size of $\boldsymbol{A}$ is $t \times d$, where t is the number of unique terms in all input documents and d is the total number of input documents. A cell value of i'th row and j'th column represents a weighed term frequency of i'th term in j'th document. The inverse document frequency (Manning et al., 2008) is used as a weighting function and is defined as:

$$\text{idf}(t, D) = \log \frac{|D|}{|\{d \in D : t \in d\}|} \tag{1}$$

where t is the weighted term, $|D|$ is the number of documents in the training set, and $|\{d \in D : t \in d\}|$ is the number of documents in the training set that contain the term t.

In this work, the set used for training IDF contained over 2 million news articles.

LSA employs Singular Value Decomposition (SVD) (Deerwester et al., 1990) to produce the matrices $\boldsymbol{U}$, $\boldsymbol{\Sigma}$ and $\boldsymbol{V}^T$. These matrices then represent the decomposition of matrix $\boldsymbol{A}$ as shown in the figure 1 and describe the concepts of the analysed documents (Baker, 2005). The matrix $\boldsymbol{V}^T$ captures how the concepts are mapped in the documents (every column represents a document vector in a concept space) and vectors of this matrix can further be used in the clustering phase.

Because the original vector space is sparse, the matrix $\boldsymbol{V}^T$ can be significantly reduced. The optimal reduction is given by 50 % of the sum of all singular values (see (Rott and Cerva, 2014) for more details on the process of reduction).

Note that in this work, the numerical analysis and data processing library ALGLIB[1] was used to realize the SVD and other mathematical operations needed for LSA and cosine similarity was employed as the vector similarity metric.

[1] http://www.alglib.net/

Random Manhattan Indexing

The main idea of RMI came from Random Projection (RP). While RP works well in tasks like image clustering, it doesn't give good results in text-based tasks because it uses Euclidean metric to compare similarity between documents.

In contrast, the RMI is designed to use Manhattan metric for this purpose, which is defined as:

$$d(\vec{a}, \vec{b}) = \sum_{i=0}^{N} |a_i - b_i| \tag{2}$$

where $\vec{a}$ and $\vec{b}$ are two vectors from a generated vector space and N is the number of dimensions of this space. Rare unexpected peaks in term frequency are suppressed because the Manhattan metric is not sensitive to non-Gaussian noise (Weeds et al., 2005). The RMI, though, retains the advantages of Random Projection. Especially, the skip of eigenvectors computation is very useful. The RMI method is able to construct L1 normed vector spaces with reduced dimensionality.

The basic presumption of RMI is that a document can be described by its index terms. These terms can be represented using a vector. A combination of the index term vectors creates a document vector. This works as two-step procedure. First, an index vector is assigned to each term. Index vector $\vec{v}$ is then randomly generated with the following probability distribution:

$$v_i = \begin{cases} \frac{-1}{U_1} & \text{with probability } \frac{s}{2} \\ 0 & \text{with probability } 1 - s \\ \frac{1}{U_2} & \text{with probability } \frac{s}{2} \end{cases} \tag{3}$$

where U_1 and U_2 are two independent uniform random variables in (0,1) and s determines the sparseness of the index vector. Value v_i represents i'th value of the index term vector.

Finally, the document vectors are computed as the sum of the documents index term vectors. This sum is defined as:

$$d_k = \sum_{t \in d} t_k \tag{4}$$

The k'th dimension of document vector d is computed as the sum of k'th dimension of every index term vector of document d.

In RMI, the problem with OOV words is solved very simply; when a vector for a previously unobserved term/words is needed, a new vector is generated from probabilistic distribution (3) and can be added to the index vector set.

Skip-Gram Model

The recently introduced Skip-gram model gained a lot of attention due to several reasons. The one reason is that the learned vectors allow to encode many linguistic patterns that can even be represented just as linear translations.

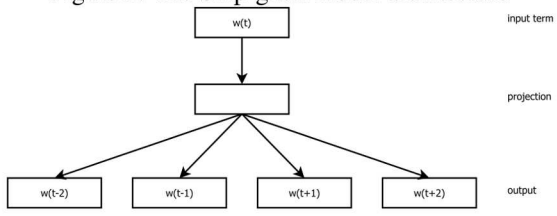
This fact is demonstrated for some interesting examples in (Mikolov et al., 2013). During our experimets, we

have have found that SGN trained on unlemmatised data is also able to represent inflection of the Czech language. For example, the result of a vector calculation vec(Michal) - vec(Michala) + vec(Petr) is closer to vec(Petra) than to any other word and the result of a vector calculation vec(Michal) - vec(Michalovi) + vec(Petr) is the most closest to vec(Petrovi). Note that in Czech, the former example represents genitive case and latter dative.

The next reason for popularity of SGM is that it is based on log-linear neural network architecture. Thus, SGM does not involve any dense matrix multiplication, making training more efficient than in case of recurrent neural networks.

The training objective of SGM is to find optimal vector representation of terms in vector space that best predict the surrounding words in the document.

Figure 2: The Skip-gram model architecture



Formally, the objective is to maximize the average log probability for a given sequence of terms $t_1, t_2, ...t_N$

$$\frac{1}{N} \sum_{n \in N} \sum_{-c \leq j \leq c, j \neq 0} \log(p(t_{n+j}|t_n)) \tag{5}$$

where c is the size of the term context.

Computing log probability is not efficient because the cost of computing depends on the size of the training set. Therefore, the authors of the model define negative sampling as objective, which replaces the log probability computation with its approximation (Mikolov et al., 2013).

SGM also employs a technique of subsampling of frequent words. This technique balances the occurrence of frequent words (e.g., "být", "a", "i" and "v") with rarer terms that have more of an informational value. The probability of terms is computed as

$$P(t_i) = 1 - \sqrt{\frac{h}{f(t_i)}} \tag{6}$$

where $f(t_i)$ is the frequency of term t_i in the training data and h is the heuristically chosen threshold. The authors recommend setting the value of the threshold around 10^{-5}. All features and principles of the Skip-Gram Model are described in Tomas Mikolov's paper (Mikolov et al., 2013).

SGM has also one disadvantage when compared to RMI: when an OOV word occurs, it is necessary to re-train the log-linear neural network model. This process can take several hours (see also section 4.3.) and this fact limits the usage of SGM in some on-line tasks.

2.3. Document Vector Clustering

Once the document vectors have been computed, document clustering can take place. In case of LSA and SGM, conventional K-means algorithm (Jain and Dubes, 1988) was employed for this purpose. In contrast, vectors extracted by using RMI were clustered using K-medians algorithm. The reason is that RMI take advantage of Cauchy distribution of probability, which does not have the mean value.

3. Experimental Setup and Data

3.1. Training data for RMI and SGM

A large corpus compiled from 8.6 GB of Czech texts was utilized to form a dictionary for RMI as well as for training of SGM. The resulting dictionary contained 706,033 unique terms (lemmas). Note that the vector length within SGM training was 200 and that the maximum skip length was 10 terms. We used a wider context than was recommended for English in (Mikolov et al., 2013) due to free order of words in the Czech language. Other parameters were kept on a default (recommended) value. The Word2vec tool[2] was used for SGM training.

3.2. Evaluation Metric Used

Our previous work (Rott and Cerva, 2014) shows that the metrics based on vector properties, e.g. angle or distance between clusters), are for evaluation of clustering almost unusable. Therefore, only the Rand Index (RI) (Rand, 1971) is used as the evaluation metric in this work.

Rand index compares automatically created clusters C with ideal reference clusters R and it is defined as:

$$RI = \frac{TP + TN}{TP + TN + FP + FN} \tag{7}$$

where TP is the number of document pairs in the same cluster in R and in the same cluster in C; TN is the number of document pairs in different clusters in R and in different clusters in C; FP is the number of pairs in different clusters in R and in the same cluster in C; and FN is the number of pairs in the same cluster in R and in different clusters in C.

3.3. Data for Evaluation

Two test sets are used for experimental evaluation within this work. Both are compiled from Czech newspapers articles.

The first one was formed by entering the query "Česká národní banka" into a web search engine at first and after that, 20 relevant documents from the result were chosen. Accordingly, we further call this test set as query-based. Ten annotators were then appointed to cluster all of these documents into five distinct clusters and as a result of this process, 10 x 5 reference clusters were created.

Table 1 describes how the created annotations correspond between individual annotators (the first row) and between annotators and the article categories (the second row). It is evident that the average annotator's opinion corresponds with the others at 87 % on average, and with the category at 88 %.

[2] https://code.google.com/p/word2vec/

Table 1: Manually created clusters and categories

	min RI	avg RI	max RI
human vs. humans	0.816	0.871	0.908
human vs. category	0.805	0.875	0.958

Table 2: Test sets used for evaluation

test set	query-based	category-based
# of documents	20	1000
average # of characters	2046	1723
average # of words	194	287

Note that the query-based set originally contained 100 articles, but complaints from the annotators forced us to shrink it to 20. The annotators mostly complained about the impossibility of remembering so many articles.

In contrast to the first set, the second test set is category-based. It is composed of 1000 newspaper articles belonging to 10 different topic categories such as culture, economics, etc. That means that in this case, the documents were clustered just using information on their category. Note that each of the 10 category-based clusters contains the same number of 100 articles.

The statistics of both the test sets are summarized in Table 2.

4. Experiments

4.1. Appropriate Similarity Metric for SGM

The first experiment was performed to find out the most suitable similarity metric for the vectors generated by SGM. For this purpose, cosine similarity was adopted from (Rott and Cerva, 2014) as it yields good results for clustering based on the LSA approach at first. The second metric used was the Manhattan distance as it generally yields good results in NLP tasks (see section 2.). For the query-based test set, the obtained results were compared to all 10 reference clusters. Thus, the mean values of RI are presented in this case.

Table 3 shows that the use of Manhattan distance yields result that correspond better with human annotations and document categories than results given by the cosine similarity. Therefore, Manhattan distance is employed for SGM within all next experiments.

4.2. RMI Dimensionality

The second experiment was conducted to find out appropriate length of vectors within the RMI approach. Note that the vector length corresponds to the number of dimensions and that the authors of RMI do not recommend using less than 100 dimensions for English (Zadeh and Handschuh, 2014).

Table 3: Results after SGM-based clustering using cosine similarity and Manhattan distance

	query-based	category-based
cosine similarity	0.576	0.681
Manhattan distance	0.702	0.833

Table 4: Results after RMI-based clustering using various vector lengths

vector length	query-based test set	category-based test set
100	0.693	0.814
200	0.682	0.813
400	0.679	0.813
800	0.691	0.812

Table 5: Results of automatic clustering using LSA, RMI and SGM

method	Queries			Categories
	min RI	mean RI	max RI	
LSA	0.458	0.505	0.558	0.745
SGM	0.668	0.702	0.721	0.833
RMI	0.653	0.682	0.706	0.813

Obtained results in table 4 show that the values of vector lengths have only slight impact on results of clustering. In the next experiments, we use 200-dimension model.

4.3. Comparison of LSA, RMI and SGM

The last performed experiment shows results of individual methods studied in this work. The obtained results are presented in table 5.

They show that the both the novel methods, i.e. RMI and Ski-gram model, outperform significantly the conventional clustering approach based on LSA.

It is also evident that the best clustering accuracy in terms of RI can be obtained using SGM. This method yields results that are the most similar to that of human annotators. However, the slow training and difficult handling OOVs limit the usability of this method in on-line systems. In contrast, when fast clustering algorithm is demanded, the best option is the Random Manhattan Indexing. This method offers just slightly worse results than SGM but provides an elegant and fast solution to the problem of OOV words. For example in our case, the process of SGM re-training takes 6 hours while RMI needs just several milliseconds to create a new vector for a new OOV word (these values were measured using Intel Xeon Processor E5-2620).

Table 5 also shows that RMI as well as SGM-based clustering give for the query-based test set values of RI that have lower variance when compared to results of LSA (compare the min, max and mean values of RI on individual rows of the table).

At last, it should also be noted that a particular disadvantage of RMI for some applications may lay in the fact that the produced vectors are randomly distributed while SGM approach explicitly encodes many linguistic patterns.

5. Conclusion

In this paper, three different methods for document clustering were evaluated on query- and category-based test sets compiled from a set of Czech newspaper articles. The obtained results show that RMI and SGM yield better results than LSA. Although the accuracy of RMI is slightly worse when compared to the output from SGM, the main advantage of RMI for practical usage consists in the fact that it is not necessary to perform any time expensive retraining when an OOV word occurs. The values of RI produced by RMI were by 17 % worse on average than in the case of human annotators. In future work, we plan to further investigate the use of RMI in other natural language processing tasks such as automatic summarization of newspaper articles.

Note that all the test sets used within this work are available on-line on a web page `http://nlp.ite.tul.cz`.

Acknowledgment

This work was supported by the Technology Agency of the Czech Republic (project no. TA04010199) and by the Student Grant Scheme 2015 (SGS) at the Technical University of Liberec.

6. References

Baker, Kirk, 2005. Singular value decomposition tutorial. *The Ohio State University*, 2005:1–24.

Deerwester, Scott, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman, 1990. Indexing by latent semantic analysis. *JOURNAL OF THE AMERICAN SOCIETY FOR INFORMATION SCIENCE*, 41(6):391–407.

Gong, Yihong and Xin Liu, 2001. Generic text summarization using relevance measure and latent semantic analysis. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.*

Jain, Anil K. and Richard C. Dubes, 1988. *Algorithms for Clustering Data*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc.

Manning, Christopher D., Prabhakar Raghavan, and Hinrich Schütze, 2008. *Introduction to Information Retrieval*. New York, NY, USA: Cambridge University Press.

Mikolov, T., I. Sutskever, K. Chen, G. Corrado, and J. Dean, 2013. Distributed representations ofwords and phrases and their compositionality. In *Advances in Neural Information Processing Systems*.

Rand, William M., 1971. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850.

Rott, M. and P. Cerva, 2013. Summec: A summarization engine for czech. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8082 LNAI:527–535.

Rott, M. and P. Cerva, 2014. Investigation of latent semantic analysis for clustering of czech news articles. In *Proceedings - International Workshop on Database and Expert Systems Applications, DEXA*.

Straková, J., M. Straka, and J. Hajič, 2014. Open-Source Tools for Morphology, Lemmatization, POS Tagging and Named Entity Recognition. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. Baltimore, Maryland: Association for Computational Linguistics.

Weeds, J., J. Dowdall, G. Schneider, B. Keller, and D. Weir, 2005. Using distributional similarity to organise biomedical terminology. *Terminology*, 11(1):107–141.

Zadeh, B. Q. and S. Handschuh, 2014. Random manhattan indexing. In *Proceedings - International Workshop on Database and Expert Systems Applications, DEXA*.