A NOVEL METHOD FOR DOCUMENT SUMMARIZATION USING WORD2VEC

Zhibo Wang
Department of Computer Science
Georgia State University
Atlanta, GA 30302-5060, USA
zwang6@student.gsu.edu

Long Ma
Department of Computer Science
Georgia State University
Atlanta, GA 30302-5060, USA
lma5@student.gsu.edu

Yanqing Zhang
Department of Computer Science
Georgia State University
Atlanta, GA 30302-5060, USA
yzhang@gsu.edu

Abstract

Texting mining is a process to extract useful patterns and information from large volume of unstructured text data. Unlike other quantitative data, unstructured text data cannot be directly utilized in machine learning models. Hence, data pre-processing is an essential step to remove vague or redundant data such as punctuations, stop-words, low-frequency words in the corpus, and re-organize the data in a format that computers can understand. Though existing approaches are able to eliminate some symbols and stop-words during the pre-processing step, a portion of words are not used to describe the documents' topics. These irrelevant words not only waste the storage that lessen the efficiency of computing, but also lead to confounding results. In this paper, we propose an optimization method to further remove these irrelevant words which are not highly correlated to the documents' topics. Experimental results indicate that our proposed method significantly compresses the documents, while the resulting documents remain a high discrimination in classification tasks; additionally, storage is greatly reduced according to various criteria.

Keywords: *Data pre-processing; Word2Vec; clustering; document similarity; document representation*

1. INTRODUCTION

Tons of textual resources are created and uploaded to Internet every day, such as Tweets, Blogs, webpages and so on. These text data not only require lots of storage space to save and backup, but also bring in a lot of data organizing, processing and analyzing tasks. Hence, to extract meaningful and non-trivial knowledge from the humongous textual database becomes the overarching goal of text mining. However, unstructured text data, unlike those quantitative data, cannot be simply analyzed at the word level. A data pre-processing step has to be implemented to clean the texts, and convert them to a more understandable format by computers.

Generally, a document always contains some words which are not quite related to the documents' topics. Removing these words reduces the corpus size such that decreases the dimension of features at the same time; additionally, it reduces noises of the documents in order to improve the usability of documents. Stop-words like prepositions, articles, and pro-nouns are high-frequency words in documents. These words do not carry any information regarding contents, but used only for grammar purpose. Therefore, removing these words do not have any effect on the document itself. Moreover, documents always contain various data formats and non-informative features such as date formats, number formats or currency format. There are alternative ways to address these formats. Besides elimination, other ways to handle them include putting all the dates in a container named DATE, and converting numbers into letter-format such as 5 is May or 10 is ten, and etc. But there is not a standard way to pre-process the punctuations according to the purpose of analysis. For example, punctuations are much useful for the sentimental analysis that ":)" is positive and "T_T" is negative. For other purpose, however, punctuations should be removed at the very beginning. Therefore, the way to pre-process the textual databases is highly dependent on its applications.

Normally, there are two ways to convert the textual data into quantitative data used for advanced analysis: one is based on one-hot word representation and the other one is based on the word embedding.

One-hot word representation is a vastly used word representation method, which builds a vocabulary according to the word occurrences in all given documents. It simply represents a word with a vocabulary size vector with 1 at its corresponding positon and 0 at other positions. Bag-of-words model extends one-hot presentation to represent a document by aggregating all one-hot word vectors in the document. It is obvious that the resulting document vector length is the same as

Proc. 2016 IEEE 15th Int'l Conf. on Cognitive Informatics & Cognitive Computing (ICCI*CC'16)
Y. Wang, N. Howard, B. Widrow, K. Plataniotis, & L.A. Zadeh (Eds.)
978-1-5090-3846-6/16/$31.00 ©2016 IEEE

the vocabulary size and each element in the document vector is the frequency of corresponding word [1]. Bag-of-words model has two drawbacks, on one hand, as the number of dimensions of document vector replies on the occurrence of words all documents, the one-hot vectors usually have very high dimensions. On the other hand, bag-of-words model ignores spatial relationship among words since element in vector only represent the counts of the word.

Distributed representation predicts each word with a very low-dimension vector ranging from 50 to 300 [2]. Generally, a word vector is randomly initialized, and is trained and updated according to each word's contexts. Since contextual information is taken into account, models like Neural Network are able to update each word vector with semantic meanings. Global Vectors for Word Representation (Glove) and Word2Vec are the most popularly used among the distributed representation methods. Glove predicts words and contexts by constructing co-occurrence matrix according to the given documents, and updates the vectors by reducing the dimensionality of the matrix. Word2Vec uses a three-layer neural network to train each word vector with respect to its surroundings, and uses Gradient Descent algorithm to update the weights of the network [3, 4]. The resulting word vectors by Word2Vec are highly correlated with the practical semantics, for example "king – man + woman = queen", and "Paris – France + Italy = Roma" [5]. Comparing with one-hot representation methods, distributed representation models are able to identify the relationship between two words even that one occurs much more frequently than the other. For example, a sentence "There is a dog running in the park." occurs many times, while the other sentence "It is a cat sitting in the park." appears only once. Word2Vec analyzes the contexts of the word 'cat' which is very similar to the contexts' of "dog", and then predicts "cat" at a position closer to "dog" in the semantic space. Recently, researchers have proposed many variations of Word2Vec to determine relationships between labels and words. Doc2Vec is one of these algorithms. Additionally, it integrates paragraph vectors at the input layer that cooperates with the word vectors to investigate correlation between labels and words. [6]. Another hybrid method clusters the word vectors and considers each cluster as a bag-of-words. It thus uses frequency of words in each bag as features of the document [7].

However, these methods also analyze many vague and trivial words that are not highly correlated with document topics. In this paper, we propose a method to further optimize the document size by eliminating the non-topic related words. In a document, some of the words are only used for connecting purpose or descriptions that are not related with the document topics. Although these words are necessary for communications, they are not meaningful for text data analysis. Our proposed method which targets on the issue of these irrelevant words is able to identify and remove them to optimize the process by extending the idea of Word2Vec. Word2Vec is a widely used algorithm to transform each word in the corpus into a word vector with semantic meanings. We consider each cluster formed by word vectors using K-means as a feature that supervises the subsequent optimization by measuring its distance from other target semantic groups. Generally, a document can be divided into multiple partitions such as paragraphs or part of paragraphs. In addition to word vectors, we also transform the document and its partitions into vectors and process them as continuous bags-of-words. Specifically, document partitions, document, and features are represented by single vectors, which dimensions are the same as any word vector in the document. Afterwards, we measure the Cosine distances between them that shorter distance indicates higher correlation, and vice versa. Hence, only partitions meet certain criterion are retained. Experimental results show that using our method size of document is significantly reduced, while accuracy of classification remains at a high level.

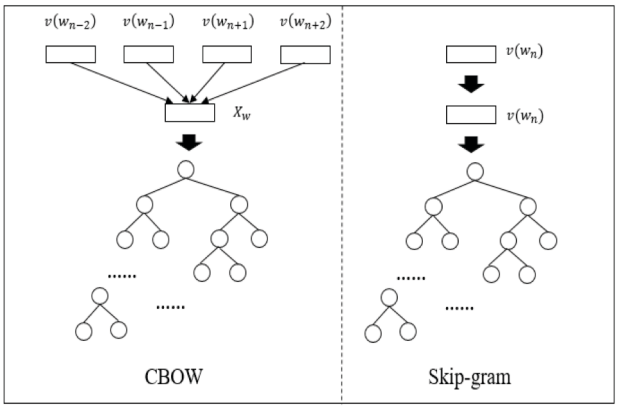

Fig. 1. CBOW and Skip-gram scheme of Word2Vec model

Remaining parts of this paper are organized as follows. The Word2Vec model is introduced in Section 2. In Section 3, we propose and explain the process of the method. In Section 4, experimental results are provided and discussed. Section 5 gives conclusions and future work.

2. WORD2VEC MODEL

Word2Vec is a generative probability model proposed by Google in 2013, which aims to investigate the semantic meanings and relationships among words. It projects each word into a low-dimensional space and predicts it by its surrounding words. Intuitively, Word2Vec attempts to explore whether a combination of words is commonly spoken by human beings with regarding to its probability of occurrence in the corpus.

Word2Vec model is a three-layer neural network including input layer, projection layer and output layer. The preceding and subsequent word vectors of the target word compose the input layer which represents contextual information. Projection layer bridges input and output layer, which aggregates word vectors from input layer and prepares to update parameters in output layer. Output layer is represented by a Huffman tree that is able to identify the sematic meanings and relationship between words

in a very efficient way of computation. The process of Word2Vec is shown in Figure 1. There are two types of frameworks of Word2Vec model: (1) Hierarchical Softmax and (2) Negative Sampling, and two different schemes (1) CBOW and (2) Skip-gram can be utilized to conduct each framework. In this paper, we take the Hierarchical Softmax + CBOW as an example to demonstrate the algorithm.

Initially every word in the corpus is randomly initialized with a fixed length vector $\{v(w_1), v(w_2), \dots, v(w_N)\}$, where the dictionary size is N. From input layer to the projection layer, all the contextual vectors $\{v(w_{n-2}), v(w_{n-1}), v(w_{n+1}), v(w_{n+2})\}$ of current word w_n are consolidated as the context vector X_w, where the window size 2 is used as an example. In the output layer, each word is assigned with a Huffman code according its occurrences in all given documents. 1 in the Huffman code represents the left child and 0 represents the right child for any given node except the leaf node. The probability of each node to select its left child or right child is calculated using logistic regression. Calculation of the conditional probability from the contextual vector X_w at the root node to the current word at a leaf node can is shown in equation (1), where w means the current word, l^w is the number of nodes on the path from root node to the current word node, θ is a vector containing parameters at each node.

$$p\left(w_x \middle| context(w_x)\right) = \prod_{i=2}^{l^w} p(x_i^w | X_w, \theta_{i-1}^w) \quad (1)$$

Where

$$p(node_i^w) = \begin{cases} \sigma(X_w^T\theta), & node_i^w = 0 \\ 1 - \sigma(X_w^T\theta), & node_i^w = 1 \end{cases}$$

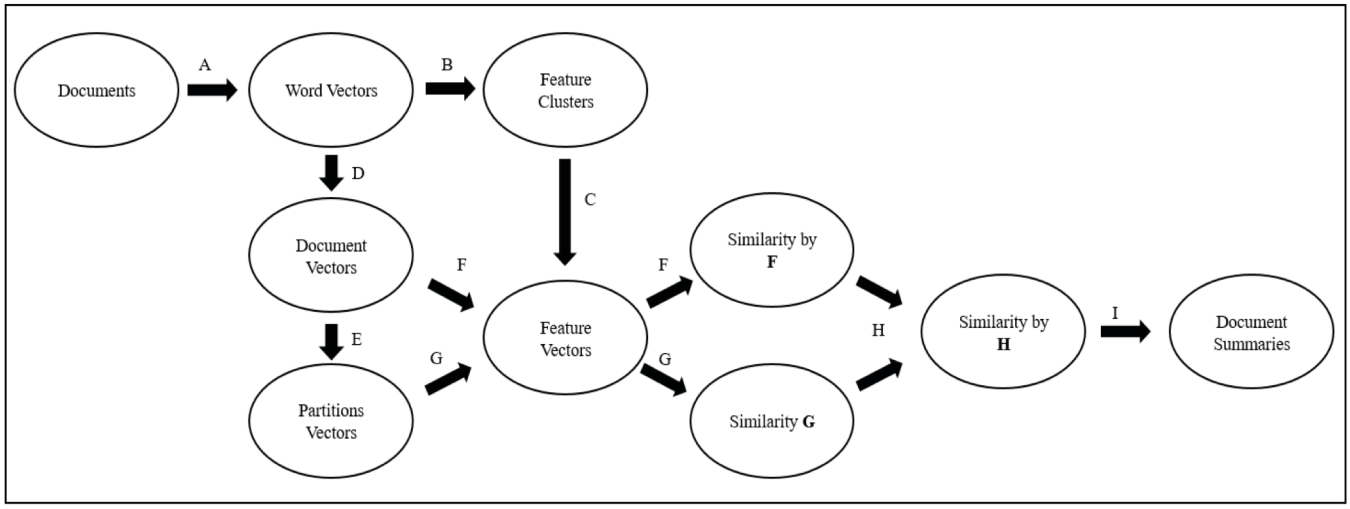

Fig.2. Process of the proposed method

The learning objective function is defined to maximize the log-likelihoods function (2) and update the word vector (3) and θ (4).

$$\mathcal{L}(w,i) = (1 - node_i^w) \cdot \log[\sigma(X_w^T\theta)] + node_i^w \cdot \log[1 - \sigma(X_w^T\theta)] \quad (2)$$

$$\theta_{i-1}^w := \theta_{i-1}^w + \eta[1 - node_i^w - \sigma(X_w^T\theta)]X_w \quad (3)$$

$$v(\tilde{w}) := [1 - node_i^w - \sigma(X_w^T\theta)]\theta_{i-1}^w \quad (4)$$

3. OUR PROPOSED METHOD

Generally, Word2Vec model builds a semantic space projected from the word vectors that trained and outputted by the model, where each word is considered as a point in the space. It measures and interprets similarity in grammar or semantics through distance between any two distinct points. Both Euclidean and Cosine distance are commonly adopted as ways to calculate the distance between two points in researches. Extending the idea of Word2Vec, we generate different levels of vectors other than only word vectors to compare resemblance. The resulting output contains only "qualified partitions" which are highly correlated with the document topics. Following, we explain specifically the way to select the qualified partitions to represent the document, which is decomposed into 9 consequent steps.

3.1 Projecting Words to Word Vectors

Given a set of D documents $\{d_1, d_2, \dots, d_D\}$, a dictionary containing n words is built accordingly. By applying Word2Vec model, we predicts the word w_x by its contextual words $\{w_{x-2}, w_{x-1}, w_{x+1}, w_{x+2}\}$, here window size 2 is set as an example. Thus, each word in the collection of documents is projected into a m dimension vector $v(w_x)$, where m is a predefined value usually ranging from 50 to 300.

3.2 Clustering Words as Features

Semantically related words are clustered utilizing K-means. Moreover, we are able to extract conceptual features or categories from the word clusters. For example, in processing words in the pool of documents, we group "cat" and "dog" in a

cluster using K-means because of their high correlation. In addition to the cluster, an abstract feature is actually derived like "animal" or "pet'. Conclusively, we adopt K-means to assemble word vectors into N clusters, which centroids represent cumulative conceptual features generated from a series of similar meaning words.

3.3 Transforming Features into Feature Vectors

It is intuitive to consolidate the word vectors that are grouped in the same cluster into a feature vector. However, sizes of clusters may vary dramatically in occasions. In order to diminish the impact of sizes of clusters and unify a criterion for assessment, we adjust the cluster vectors using a size scaler. Equation (5) provides a mathematically demonstration of the aggregating and rescaling process.

$$v(cluster_c) = \frac{\sum_{i=1}^{X} v(w_x^c)}{X} \quad (5)$$

Where X is the number of words in $cluster_c$.

From: lerxst@wam.umd.edu (where's my thing)
Subject: WHAT car is this!?
Nntp-Posting-Host: rac3.wam.umd.edu
Organization: University of Maryland, College Park
Lines: 15
I was wondering if anyone out there could enlighten me on this car I saw the other day. It was a 2-door sports **car**, looked to be from the late 60s nearly 70s. It was called a Bricklin. The doors were really small. In addition, the front bumper was separate from the rest of the body. This is all I know. If anyone can tell me a model name, engine specs, years of production, where this car is made, history, or whatever info you have on this funky looking car, please e-mail.

Thanks,
- IL
---- brought to you by your neighborhood Lerxst ---- "

Fig. 3. An example from 20 newsgroup dataset

3.4 Converting Documents to Document vectors

Furthermore, we develop the idea of converting documents to document vectors. Though documents are consisted of both topic related words and uninteresting trivial words, it is expected that they are closer to location of the document topics be used to evaluate the closeness between topics of document partition document and topics of the document. Similar to the calculation of the cluster vector in step 3, equation (6) shows space. Therefore, we define a document vector $v(document_d)$ the way to determine the document vector. in the semantic as a representation of $document_d$ which can

$$v(document_d) = \frac{\sum_{i=1}^{Y} v(w_x^d)}{Y} \quad (6)$$

Where Y is the number of words in $document_d$.

3.5 Projecting Partitions of Document to Partition Vectors

Accordingly, we assume partitions of a document share the same property of document that each partition concentrates around the neighborhood of its topics in the semantic space. Thus, we propose a partition vector by accumulate information of the word vectors in it. In the calculation of a partition vector, the size scaler is denoted as Z in equation (7).

$$v(partition_k^d) = \frac{\sum_{i=1}^{z} v(w_x^k)}{z} \quad (7)$$

3.6 Measuring Similarity Between Documents and Features

Construction of document vectors and feature vectors allows us to measure the similarity between a document and any semantic category that K-means generates. In this step, we introduce the intermediate factor — features rather than compare the similarity between a document and its partitions directly. Since a feature is derived to represent a more conclusive concept, adopting it enables us to generate the semantic meanings of both documents and document partitions by a global view. Additionally, it allows us to perceive the topics hidden in the document partitions which is measured in

Table 1. Cosine distance as measurement of similarity between a document and partition

No.	Contents of Partition	Distance
1	lerxst, wam, umd, ed, thing, subject	0.0143
2	car, nntp, posting, host, rac, wam	0.0182
3	umd, ed, organization, university, maryland, college	0.0110
4	park, lines, wondering, anyone, could, enlighten	0.0101
5	car, saw, day, door, sports, car	**0.0071**
6	looked, late, early, called, bricklin, doors	0.0126
7	really, small, addition, front, bumper, separate	**0.0081**
8	rest, body, know, anyone, tellme, model	**0.0065**
9	name, engine, specs, years, production, car	**0.0071**
10	made, history, whatever, info, funky, looking	**0.0070**
11	car, please, mail, thanks, brought, neighborhood	0.0196
12	lerxst	0.0341

the following step. Cosine distance is calculated by equation (8). Thus, each document can be further represented by a distance distribution over all features in the form of a vector (9).

$$dis_d^c = 1 - \frac{v(document_d) \cdot v(cluster_c)}{\|v(document_d)\|_2 \|v(cluster_c)\|_2} \quad (8)$$

$$v(document_d)_{NEW} = (dis_d^1, dis_d^2, \dots, dis_d^{N-1}, dis_d^N) \quad (9)$$

Where N is the total number of clusters generated from the pool of documents $\{d_1, d_2, \dots, d_D\}$.

3.7 Evaluating Similarity Between a Document Partion and Features

Likewise, we use equation (10) to calculate the cosine distance from a document partition to a feature which is a

representation of similarity between the document partition and a topic. Furthermore, each partition is restructured into a new vector, where each element is the cosine distance (11).

$$dis_k^{d,c} = 1 - \frac{v(partition_k^d) \cdot v(cluster_c)}{\|v(partition_k^d)\|_2 \|v(cluster_c)\|_2} \quad (10)$$

$$v(partition_k^d)_{NEW} = (dis_k^{d,1}, dis_k^{d,2}, \dots, dis_k^{d,N-1}, dis_k^{d,N}) \quad (11)$$

3.8 Assessing Similarity Between Documents and Document Partitions Using Distance Vectors

In this step, semantic similarities between a document and its partitions are measured using the newly constructed distance vectors (12).

$$dis_k^d = 1 - \frac{v(partition_k^d)_{NEW} \cdot v(document_d)_{NEW}}{\|v(partition_k^d)_{NEW}\|_2 \|v(document_d)_{NEW}\|_2} \quad (12)$$

3.9 Identifiying non-topic related partitions

Distance measured in the preceding step represents similarity between a document and a partition of it. Shorter distance indicates higher correlation in their topics, and vice versa. Thus, we set a criterion to evaluate the correlations. Finally, partitions with longer distance comparing to the cut-off value are eliminated. If no partition of a given document meet the predefined criteria, the one with smallest distance is retained. Therefore, the remaining partitions construct a new representation of the document topics.

Table 2. Classification results under different scenarios using bag-of-words model

Avg. Document Size	No. of Clusters	Distance Threshold	Std. Dev.	Avg. 10-Fold F1-Score
Proc. Doc				
24	50	0.005	0.009	0.638
53	50	0.010	0.009	0.732
92	50	0.015	0.010	0.792
115	50	0.020	0.008	0.825
50	100	0.005	0.010	0.707
90	100	0.010	0.011	0.803
110	100	0.015	0.008	0.835
126	100	0.020	0.007	**0.850**
25	150	0.005	0.009	0.646
68	150	0.010	0.007	0.751
101	150	0.015	0.008	0.807
124	150	0.020	0.007	0.833
Orig. Doc				
160	-	-	0.009	**0.841**

Table 3. Classification results under different scenarios using distributed representation model

Avg. Document Size	No. of Clusters	Distance Threshold	Std. Dev.	Avg. 10-Fold F1-Score
Proc. Doc				
24	50	0.005	0.010	0.702
53	50	0.010	0.011	0.765
92	50	0.015	0.012	0.812
115	50	0.020	0.013	0.801
50	100	0.005	0.012	0.694
90	100	0.010	0.014	0.762
110	100	0.015	0.009	**0.834**
126	100	0.020	0.013	0.810
25	150	0.005	0.011	0.699
68	150	0.010	0.013	0.766
101	150	0.015	0.012	0.823
124	150	0.020	0.012	0.805
Orig. Doc				
160	-	-	0.012	**0.806**

4. EXPERIMENTAL RESULTS

We implement our proposed method on the 20 Newsgroups dataset, which consists of 18,846 labeled newsgroup documents proposed by Ken Lang [8]. The entire datasets are utilized to generate word vectors using the Word2Vec model. To perform the experiment, we apply Hierarchical Softmax framework and CBOW scheme as algorithm of the Word2Vec model, and execute it by Python Gensim package with default settings [9].

In addition, Python Scikit-learn package is selected to implement the K-means cluster algorithm and the Support Vector Machine (SVM) classification algorithm [10].

Figure 3 illustrates an example of a newsgroup document in the dataset. Table 1 shows the analysis results according to it. We divide the document in Figure 3 into 11 partitions after removing the symbols, numbers, stop-words and some low-

frequency words. Each of these partitions includes 6 words, which are listed in column 2 of the table. Following the steps described in the previous section, the distances between each partition and the document are measured and recorded in column 3. It is easy to acquire the topic from the original document which is related with the keyword — "car". Setting criterion to 0.001, partitions 5, 7, 8, 9, and 10 are selected as representations of the document which are "car" related. This result agrees with our assumption that shorter distances indicate higher correlations. Other non-topic related partitions, therefore, are removed to reduce the size of the document.

To quantify the effects of our proposed method, additional experiments are implemented. We conduct classification analysis on both our processed documents that consists only of topic-related partitions and the "original" documents. Both types of documents are pre-processed by removing symbols, numbers, stop-words and very low-frequency words. The only difference between them is our processed documents eliminate non-topic related partitions. This analysis attempts to evaluate the impact of reduced document size to subsequent analyses by comparing classification accuracies between the two sets of documents

We explore the performance of the proposed method under several scenarios. Different clusters are tested to assess the impact of various topic specification scales. Word clusters group into different semantic categories with predefined number 50, 100, and 150. Additionally, we contrast the outcomes by various distance criteria as well, which are set to 0.005, 0.010, 0.015, and 0.020 respectively. Utilizing SVM we conduct the classification analysis under bag-of-words and distributed representation model, and measure the results with average 10-fold F1-score as well as standard deviation. Table 2 compares classification accuracy under different scenarios and also demonstrates the average document size after removing non-topic related partitions.

Results of Table 2 and Table 3 are derived from bag-of-words method and distributed representation method respectively. After removing symbols, numbers, stop-words and other low-frequency words, the average size of the original documents is 160 in the collection of the 18,846 documents. The average size of the processed documents varies according to different scenarios. The smaller distance criterion is selected, the less words are retained the documents. Since smaller and stricter criterion is associated with higher correlation. Thus, setting the criterion to 0.005 enables the method to remove more moderately related partitions than setting it to 0.020. However, the average document size is not linearly associated with the predefined number of clusters. Smaller number of clusters implies the generated features are broader and more general. Topic of a feature could be very ambiguous because the feature is an aggregation of a very wide range of words. In contrary, larger number of clusters indicates the derived features are very specific. Topics of this type of features could be very vague. Since these features probably are constructed with extremely less words than its natural semantic categories should contain.

Apparently the most accurate classifications identified from Table 2 and Table 3 belong to the processed documents group. 85% of the documents are detected to the correct labels using bag-of-words method, 83.4% are properly classified implementing while distributed representation method, and in contrast results from the original documents are 84.1% and 80.6%. Both improve the accuracy of converting the original documents under different methods. Moreover, with the improvement in accuracy, the document sizes reduced significantly. Comparing with the average size of original documents 160, the number of words in the best scenarios are 126 and 110. With a decrease of document sizes about 21% - 31%, considerable storage space is released. Hence, our proposed method significantly abbreviates the original documents while maintains important topic-related information as a suggested step prior to further analyses.

5. CONCLUSION AND FUTURE WORK

In this paper, we develop an optimization pre-process method to further eliminate non-topic related words after removing symbols, numbers, stop-words and other non-informative words. We extend the idea of Word2Vec and introduce the concepts of document partitions and distance vectors. Non-topic related document partitions are eliminated according to its correlation with the document topics. Experimental results show that the proposed method reduces the size of the targeting documents by a considerably high proportion while the retained partitions maintain very accurate in classification. Dramatically decrease in size of documents enables appreciably release of the storage space which is an essential factor to take into account in processing big data. Additionally, experiments also exhibit a favorable property of the method that the remaining document partitions well preserve important, explicit, topic-related information.

However, some challenges still stand for us to further research on. Based on our experiments, both selections of number of clusters and cut-off values impact prediction accuracy considerably. Attaining of high discrimination accuracy depends on experience heavily. Therefore, we plan to investigate the effects of choices of different parameters. In the subsequent researches, ranges of parameters will be recommended according to the properties of document collections. Another challenge rises from construction of document partitions. Our proposed method removes the entire partition if certain criteria are not met. Nonetheless, a certain proportion of words in a retained partition is ambiguous or vaguely related to the document topics. Hence, additional explorations on eliminating non-topic related words from a retained partition is another goal of our future research. Moreover, sizes of our experimental documents from the 20 Newsgroup dataset are comparably small. Assessment of our proposed method effects on large size documents will be further conducted.

References

[1] Z. Harris, "Distributional Structure". Word. Vol. 10, pp. 146–62, 1954.

[2] Y. Bengio, H. Schwenk, J. S. Senécal , F. Morin, and J. L.Gauvain, “Neural probabilistic language models,” *Innovations in Machine Learning* Vol. 194, pp. 137-186, 2006.
[3] J. Pennington, R. Socher, and C. D. Manning, “Glove: Global vectors for word representation,” *Proceedings of the Empirical Methods in Natural Language Processing*, 2014.
[4] Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” *Advances in Neural Information Processing Systems*, pp. 3111–3119, 2013.
[5] https://code.google.com/archive/p/word2vec/
[6] Q. V. Le and T. Mikolov, “Distributed Representations of Sentences and Documents,” *International Conference on Machine Learning*, 2014.
[7] L. Ma, and Y. Zhang, “Using Word2Vec to Process Big Text Data,” *2015 IEEE International Conference on Big Data*, pp. 2895-2897, 2015.
[8] K. Lang, “20 newsgroup data set,” 30-Sep-2015, qwone.com/~jason/20Newsgroups/
[9] R. Reh˚ ˇ u˘rek and P. Sojka, “Software Framework for Topic Modelling with Large Corpora,” *New Challenges for NLP Frameworks*, 2010.
[10] F. Pedregosa, Fabian, et al., "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, Vol 12, pp. 2825-2830, 2011.