Journal of AI and Data Mining
Vol 7, No 3, 2019, 443-450
DOI: 10.22044/JADM.2019.7400.1876

A Joint Semantic Vector Representation Model for Text Clustering and Classification

S. Momtazi*, A. Rahbar, D. Salami, and I. Khanijazani

Computer Engineering and Information Technology Department, Amirkabir University of Technology, Tehran, Iran.

Received 23 September 2018; Revised 26 November 2018; Accepted 07 April 2019
**Corresponding author: momtazi@aut.ac.ir (S. Momtazi).*

Abstract
Text clustering and classification are two main tasks of text mining. Feature selection plays a key role in the quality of the clustering and classification results. Although word-based features such as Term Frequency-Inverse Document Frequency (TF-IDF) vectors have been widely used in different applications, their shortcomings in capturing semantic concepts of text have motivated researches to use semantic models for document vector representations. The Latent Dirichlet Allocation (LDA) topic modeling and doc2vec neural document embedding are two well-known techniques for this purpose.
In this work, we first studied the conceptual difference between the two models and showed that they had different behaviors and capture semantic features of texts from different perspectives. We then proposed a hybrid approach for document vector representation to benefit from the advantages of both models. The experimental results on 20newsgroup showed the superiority of the proposed model compared to each one of the baselines on both text clustering and classification tasks. We achieved a 2.6% improvement in F-measure for text clustering and a 2.1% improvement in F-measure in text classification compared to the best baseline model.
Keywords: *Text Mining, Semantic Representation, Topic Modeling, Neural Document Embedding.*

1. Introduction

With the growth of the Internet and social media, a huge amount of texts is produced every day. Processing this amount of data and extracting information from them cannot be done without the help of computational techniques, and this is where text mining comes into play. Different techniques such as machine learning, data mining, and information retrival can be used to get information from the text.
Text clustering and text classification are two major tasks in the field of text mining. In text classification, we are given a set of documents and want to classify them to a pre-defined set of labels. A text classification task usually starts with a set $D = \{d1, d2, ..., , dn\}$ of training examples. Each one of these examples is labeled with a class Ci. The task is then assigning a class label to a new document [1]. Various kinds of machine learning algorithms such as support vector machine (SVM), K-nearest neighbors, and Naïve Bayes can be used as the learning algorithms. News categorization, spam filtering, language identification, and sentiment analysis are some of the well-known applications of text classification. Text clustering is another important topic in text mining. Generally, clustering aims at finding groups of objects that are similar to each other and organize them into a fixed and pre-defined number of clusters. Unlike text classification, in this model, no labeled data is provided. In text clustering, the algorithm is given a set of documents, and it has to provide a set of clusters, each of which contains a proportion of documents that are similar [2]. For the text clustering task, different unsupervised learning algorithms such as K-means and hierarchical clustering are introduced [3]. Search result clustering, grouping similar posts in social networks, analyzing customer/employee feedback, and discovering meaningful implicit subjects across all documents are some of the main applications of this task.
Feature selection plays an important role in the accuracy of text clustering and classification.

Various models have been proposed for representing text in vector space. In this paper, we propose a hybrid vector representation scheme to improve the performance of these two tasks — our proposed model benefits from two different semantic representations of text, namely Latent Dirichlet allocation (LDA) topic modeling [4], and neural doc2vec method. Although semantic modeling of text has been widely studied in different natural language processing tasks, including question answering [5], plagiarism detection [6], query suggestion [7], expert finding [8], and dialog systems [9], to the best knowledge of the authors, the combination of two different semantic representations has not been studied in-depth.

The structure of this paper is as follows: in Section 2, we provide a brief overview of different vector representation approaches for text mining including topic modeling approach and neural text embedding. Section 3 describes our hybrid proposed method and its differences with two baseline models. In Section 4, we describe our experiments, including the dataset, experimental setups (such as algorithms, libraries, pre-processing steps), evaluation metrics, and clustering and classification results using multiple vector representations and our proposed method. Finally, Section 5 summarizes the paper.

2. Vector representation

Most of the text mining tasks, specially text classification and text clustering, require a vector representation for each document. After converting a document to a vector, we can use different unsupervised and supervised learning algorithms to perform clustering and classification tasks, respectively. The quality of the vector we use for clustering or classification has an important impact on the output of the task. Several techniques for converting a document to a vector have been proposed. In this section, the main existing methods for this goal are introduced.

2.1. TF-IDF model

The first approach for document representation is considering the words mentioned in the documents. To this aim, the set of top *k* frequent words in the document collection are used as a feature. Each document *d* is then represented as a vector $\vec{d} = (d^{(1)}, d^{(2)}, ..., d^{(k)})$ in a vector space, such that similar vectors will correspond to documents that consist of similar words.

Each dimension of vector space represents a word from the vocabulary. The estimation of the vector elements $d^{(i)}$ for a document d is defined as a mixture of $TF(w,d)$ and $DF(w)$. $TF(w,d)$ is the *term frequency* of the i^{th} term in the feature set; i.e., the number of times word *w* appears in document *d*. $DF(w)$ stands for *document frequency* which means the number of documents in which the word *w* occurs at least one. The *inverse document frequency* of a word ($IDF(w)$) can be calculated from the document frequency [3].

$$IDF(w) = \log(\frac{|D|}{DF(w)}) \quad (1)$$

$|D|$ is the total number of documents. For words that occur in many documents, IDF is low, and for words that occur in only one document, this factor is the highest. The value $d^{(i)}$ of the corresponding word *w* for document *d* is then calculated as follows [10]:

$$d^{(i)} = TF(w,d)IDF(w) \quad (2)$$

Although the TF-IDF model is the most well-known approach for text mining tasks, it does not guarantee to produce good results since this representation does not consider the semantic similarity of words, which is an important issue in text mining. For example, ‘car’ and ‘automobile’ or ‘inventor’ and ‘creator’ have the same meanings, but TF-IDF model does not consider their semantic relations.

2.2. Topic modeling

One of the major methods that can be used to find a vector representation of a document that considers the whole semantic concept of documents is topic modeling. The idea of topic modeling is based on the fact that every document is a combination of abstract topics in which each topic is a probability distribution over words [11]. For each one of the topics, we expect a particular set of words. It can be said that the importance of topic modeling is finding patterns of using words in each document and finding documents with similar usage patterns [12]. Using topic modeling, we can represent each document as a vector of topics that shows the probability of each topic in the document.

There are several methods for topic modeling. Here, we take a brief look at two main methods for topic modeling, namely Probabilistic Latent Semantic Analysis (PLSA) and LDA.

2.2.1. PLSA

PLSA is a topic modeling method proposed in [13]. PLSA tries to distinguish between different contexts where a word can be used in a document [12]. It can be said that "PLSA is based on a statistical model that is referred to as an aspect model. An aspect model is a latent variable model for co-occurrence data" [14].

2.2.2. LDA

LDA [4] is another topic modeling approach used in text mining. This algorithm is mainly based on statistical topic models. In LDA, each document is modeled as a mixture of topics, and, as said earlier, each topic is a distribution over words. Using these topics, we can find an appropriate vector representation of the document that can be used for different purposes. After applying LDA on a corpus, we are given two main outputs: document-topic distribution and topic-term distribution.

Before we can formally define the LDA process, we introduce the following notations:

- D denotes the number of documents in the entire corpus.
- The number of topics, denoted by T, is assumed to be known and fixed.
- Each topic Φ_t, where $1 \le t \le T$, is a distribution over a fixed vocabulary of terms, and Φ_{tw} is the term proportion of term *w* in topic *t*.
- θ_d is the topic mixture of the d^{th} document, and θ_{dt} is the topic proportion of topic *t* in document d.
- z_d is the topic assignments for document *d*, where $z_{d,n}$ is the topic assignment for the *n*th term in document d.
- w_d is the term occurring in document d, where $w_{d,n}$ is the *n*th term in document d. All terms are elements of a fixed vocabulary.
- β is the Dirichlet prior on the topic-term distributions.
- α is the Dirichlet prior on the document-topics distributions.

The generative process works as follows:

1. For every topic, choose a distribution Φ_t from a Dirichlet distribution with parameter β, that is, choose $\Phi_t : Dir(\beta)$, where $1 \le t \le T$.
2. For each document *d*,
   a. Choose $\theta_d : Dir(\alpha)$, and
   b. For each term in document d,
      i. Randomly choose a topic assignment $z_{d,n}$ from the distribution θ_d for the *n*th term in document *d*.
      ii. Then randomly choose a term $w_{d,n}$ from the distribution.

As indicated earlier, the generative process is merely a thought experiment. To create a topic model for a given corpus, the process would have to be reversed. The topic-terms distributions, the document-topics distributions, and the topic assignments are all unknown or hidden structures. The documents are the only observed data. The graphical model in figure 1 visualizes the dependencies among these random variables.

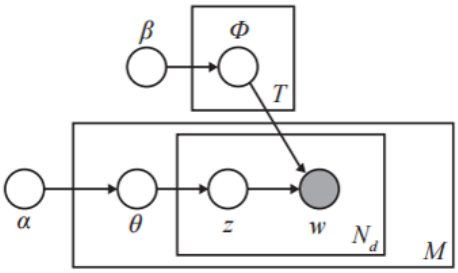

Figure 1. A graphical model for the generative process[1]

The document-topic distribution is a potential candidate for document vector representation, which can be used for text mining tasks. Among PLSA and LDA, the latter one received more attention due to its ability to infer topics for unseen documents, which is an important factor in many applications.

2.3. Neural text embedding

Vector representation of words using neural network models received researchers' attention in the recent years.

Word2vec was introduced by Google as a two-layer neural network, which computes vector representation of words including their context using implementation of continuous bag-of-words and skip-gram architectures [15]. "The output of

[1] The diagram has been taken from https://www.cs.cmu.edu/~nasmith/LS2/gimpel.06.pdf

word2vec is a vocabulary of words, which appear in the original document along with their representations in an n-dimensional vector space" [16].
In neural language modeling, language models are built based on the neural network, in which the probability distribution of each word w_t given its n previous words can be estimated with softmax [17]:

$$P(w_t \mid w_{t-1},...,w_{t-n-1}) = \frac{e^{y_{w_t}}}{\sum_{i=1}^{T} e^{y_i}} \quad (3)$$

where y_t is log-probability of word w_t normalized by the sum over all log-probabilities in a given corpus. Therefore, the objective function tries to minimize this probability over all word T in the corpus:

$$j_\theta = \frac{1}{T}\sum_{t=1}^{T} \log P(w_t \mid w_{t-1},...,w_{t-n-1}) \quad (4)$$

The idea of word embedding is tight with a neural network language model. Continuous bag of words tries to predict the target (center) word w_t based on a sliding window of n (Figure 2). In fact, at training time, each word has two roles, one as the target word and the other as the context of the other words. In contrast to a continuous bag of words, skip-gram tries to predict surrounding words w_{t+j} based on a target word w_t.

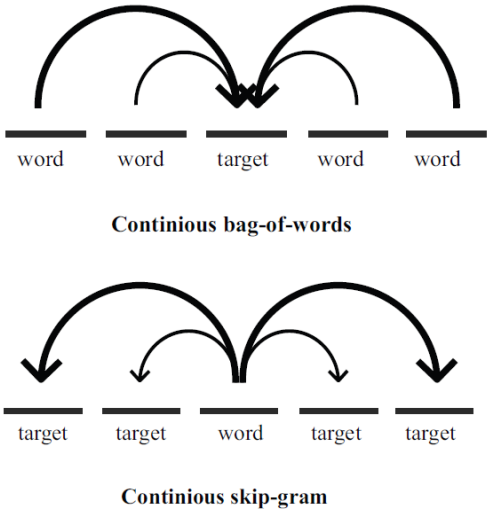

Figure 2. Continuous bag-of-words predict target based on surrounding words, while continuous skip-gram predict surrounding targets based on the current word [18].

The output of the word2vec model has been used in various natural language processing tasks which need word vector representation. The similarity of vectors in this model represents the semantic similarity of their corresponding words.
Since word2vec only provides vector representation for words, we need to combine them in some way to get a representation of the whole document. "*Doc2vec* modifies the word2vec model into unsupervised learning of continuous representation for larger blocks of text, such as sentences, paragraphs or entire documents" [16]. The doc2vec model was introduced by Le and Mikolov [19] as an extension to word2vec model to use this model for representing a sentence instead of a single word.
The doc2vec model is used to learn vector representations of texts with different lengths. Using this model, each document is represented by a dense vector. These vectors are trained to predict the words in the document and have a better performance than the bag-of-words models, in which the ordering of words are lost. In order to learn these representations, the paragraph vectors are concatenated with word vectors to predict the following word in a given context [19].
Doc2vec has two variations: Distributed Bag Of Words *(DBOW)* and Distributed Memory Paragraph Vector *(DMPV)*. *DBOW* is a simple model that ignores word order, while *DMPV* is a more complex model with more parameters that considers the words order [20].
The vector representation of documents produced by doc2vec is another alternative candidate for text mining tasks.

3. Hybrid vector representation

As stated earlier, to classify and cluster documents, we need a vector representation for each document. In the previous sections, we introduced LDA and doc2vec as two important methods used for representing documents as vectors.
Both of these approaches have been used for text mining tasks and achieved promising results compared to the TF-IDF baseline due to capturing hidden semantic features of texts [21]. These models, however, represent documents with totally different approaches. Analyzing the output of text clustering using LDA and doc2vec representations show that the semantic features of text captured by these approaches are totally different. To prove this statement, we performed an experiment to measure the independence level of clustering output using the mentioned models. The results of this experiment reported in Section 4.4, indicate the difference between the two vector representations.

This motivated us to propose a hybrid vector representation model to benefit from the advantages of both LDA and doc2vec approaches.

To this aim, in the proposed model, the hybrid vector representation is created by concatenation of the two vectors found for each document. The resulting hybrid vector is a vector that is double the size of the previous ones used in each document. The new vector can then be used for both the clustering and classification tasks.

4. Experimental results

4.1. Dataset

The *20Newsgroup* [22] dataset is a collection of approximately 20,000 newsgroup documents, partitioned across 20 different newsgroups, each corresponding to a different topic. The 20newsgroup collection has become a popular dataset for experiments in text applications of machine learning techniques such as text classification and text clustering.

In our experiments, we used a subset of the 20Newsgroup dataset that has 12 categories, 6926 documents, and 146090 unique words. The dataset is in English, and all experiments are done for the English language.

4.2. Experimental setup

To perform our experiments, we used the SVM algorithm for classification as it is one of the well-known algorithms in various applications of this task [23]. For the clustering task, the K-means algorithm is utilized [24], which is also one of the well-known techniques for text clustering. For the SVM and K-means algorithm, the *sklearn* library of Python is used.

The SVM kernel is set to RBF and 0.1 is used for both C and gamma parameters.

In the pre-processing step, before computing the vector representations, we removed the stop words, header, footer, quoting, and punctuations from documents using the *NLTK* library.

As mentioned earlier, three different vector representations were used in our experiment. The TF-IDF model was used as the baseline, while the LDA and doc2vec models were used as the basis of the hybrid model. For the TF-IDF model, the *sklearn* library of Python was used. For LDA topic modeling, we made use of the MAchine Learning for LanguagE Toolkit (MALLET), a Java-based suite of algorithms for statistical natural language processing written by [25]. For the doc2vec model, the *gensim* library of Python with *dbow* mode was utilized.

The Dirichlet priors α and β were set to $\alpha = 50/K$ and $\beta = 0.1$, which are the common settings in the literature.

The source code of this project is available on GitHub[2].

4.3. Evaluation metrics

For evaluating text classification task, we used 5-fold cross-validation such that four folds were used as the training data and one fold as the test data. We performed 5-fold cross-validation; and we calculated precision, recall, and f-measure for each fold and then we averaged the results.

Precision and recall are defined as follows:

$$precision = \frac{tp}{tp + fp} \tag{5}$$

$$recall = \frac{tp}{tp + fn} \tag{6}$$

where *tp* is the number of instances correctly classified in the target class, *fp* is the number of instances incorrectly classified in the target class, and *fn* is the number of instances in the target class that is not recognized by the classifier.

F-measure is defined as the harmonic mean of precision and recall. After finding precision and recall, f-measure is calculated as follows:

$$f - measure = \frac{2 \times precision \times recall}{precision + recall} \tag{7}$$

To evaluate text clustering task with different vector representations, we first assign a label to each cluster. The label of a cluster will be the most frequent label in that cluster. Then the label of each cluster will be assigned to all documents in that cluster. Having gold labels and assigned labels of documents in hand, we can calculate precision, recall, and f-measure.

4.4. Clustering results

In the first step of our experiments, we performed the text clustering algorithm using TF-IDF, LDA, and doc2vec vectors.

[2] https://github.com/ayhansalami/hybrid-method-vector-representation

For the clustering tasks, the labels of the documents are ignored, and we aimed at clustering the dataset into *50* clusters. The documents are represented using TF-IDF as word-base representation and LDA and doc2vec vectors as semantic representations. The TF-IDF vector includes 1000 most frequent words of the data. The LDA and doc2vec vector size is 100. The result of the algorithm can be seen in table 1.

As it can be seen in the results, both semantic representation models outperform the word TF-IDF representation. Comparing the results of LDA and doc2vec, we can see that the doc2vec model performs better than LDA.

Table 1. Comparison of Vector Representation Models in Clustering

Model	F-Measure
TF-IDF	0.378
LDA	0.530
doc2vec	0.667

As mentioned in Section 3, we aimed at concatenating the two semantic vector representations and building a hybrid vector representation to increase the text clustering and classification performance. Before doing this, we need to estimate the degree of independency between LDA and doc2vec representations. To find that, we used two measures, namely the Jaccard Index and the Adjusted Rand Index, for comparing the output of clustering for the two approaches [26].

These two metrics have expected value zero for independent clusterings and maximum value 1 (for identical clusterings). These two measures were computed with vector sizes of 100. Based on the results shown in table 2, the dependency between the two clustering outputs is very low. This result show that although both approaches focus on the semantic features of texts, the basis of their techniques differs from each other; i.e., there are some hidden semantic features that are only captured by LDA vector representation, and vice versa. Based on this experiment, we performed clustering with the proposed hybrid vector representation model.

Table 2. Clustering Independence Level.

Metric	Results
Jaccard Index	0.0942
Adjusted Rand Index	0.1332

Table 3. Comparison of Hybrid Model with Baseline Models in Clustering (10 clusters).

Vector Model	Vector Size	F-Measure
TF-IDF	100	0.225
TF-IDF	200	0.227
TF-IDF	1000	0.364
LDA	100	0.420
LDA	200	0.420
doc2vec	100	0.580
doc2vec	200	0.527
TF-IDF/Doc2Vec	100	0.500
TF-IDF/Doc2Vec	200	0.566
Hybrid	100	0.575
Hybrid	200	0.572

Table 4. Comparison of Hybrid Model with Baseline Models in Clustering (50 clusters).

Vector Model	Vector Size	F-Measure
TF-IDF	100	0.290
TF-IDF	200	0.329
TF-IDF	1000	0.378
LDA	100	0.530
LDA	200	0.466
doc2vec	100	0.667
doc2vec	200	0.676
TF-IDF/Doc2Vec	100	0.555
TF-IDF/Doc2Vec	200	0.645
Hybrid	100	0.675
Hybrid	200	0.690

Table 5. Comparison of Hybrid Model with Baseline Models in Clustering (100 clusters).

Vector Model	Vector Size	F-Measure
TF-IDF	100	0.308
TF-IDF	200	0.350
TF-IDF	1000	0.433
LDA	100	0.642
LDA	200	0.642
doc2vec	100	0.670
doc2vec	200	0.675
TF-IDF/Doc2Vec	100	0.567
TF-IDF/Doc2Vec	200	0.641
Hybrid	100	0.701
Hybrid	200	0.665

The results of the proposed model with different numbers of clusters are shown in tables 3-5. As it can be seen in the table, the proposed hybrid model outperforms both individual models. This indicates that by merging both vector representations, we can capture more semantic features from the texts and achieve a better performance as a result. As mentioned, since in the proposed model the size of the vector is doubled, we also compared it with the baselines with double vector size to show that the improved results are not due to the higher vector dimension.

4.5. Classification results

In the next step of our experiments, we performed text classification on the 20Newsgroup dataset using the SVM algorithm. Similar to clustering, we used two different vector sizes 100 and 200 for LDA and doc2vec vector representation.

The classification experiments are done based on 5-fold cross-validation. The results of the proposed hybrid model, as well as each of the baseline models, is shown in table 6. As it can be seen in this table, we have an improvement in results when using any of the semantic vector representations compared to the TF-IDF model. Comparing the LDA and doc2vec models, we observed that the LDA model performed better than doc2vec, although it had a lower performance in clustering. These results show that the semantic features captured by LDA are more adequate than doc2vec for the classification task, while doc2vec model is a better fit the clustering task. The proposed hybrid model performs the best for the classification too, such that the classification performance improved *1.9%* compared to LDA and *6.7%* compared to doc2vec.

Table 6. Comparison of Hybrid Model with Baseline Models in Classification.

Vector Model	Vector Size	F-Measure
TF-IDF	100	0.362
TF-IDF	200	0.426
TF-IDF	1000	0.609
LDA	100	0.763
LDA	200	0.754
doc2vec	100	0.708
doc2vec	200	0.713
TF-IDF/Doc2Vec	100	0.717
TF-IDF/Doc2Vec	200	0.727
Hybrid	100	0.784
Hybrid	200	0.782

5. Conclusion

In this paper, we provided a comparative study on different vector representation models for text clustering and classification, and showed that semantic features captured more information from the texts in both tasks. Moreover, we proposed a joint semantic representation model that uses both LDA topic modeling and doc2vec neural text embedding as a feature vector of text. The experimental results on the 20newsgroup collection showed that the proposed model outperformed each of the baseline models for both clustering and classification of documents.

References

[1] Korde, V. & Mahender, C. N. (2012). Text classification and classifiers: A survey. International Journal of Artificial Intelligence & Applications, vol. 3, no. 2, p. 85.

[2] Hotho, A., Nürnberger, A., & Paaß, G. (2005). A brief survey of text mining. in Ldv Forum, vol. 20, no. 1, pp. 19–62.

[3] Aggarwal, C. C. & Zhai, C. (2012). A survey of text clustering algorithms, in Mining text data, Springer, pp. 77–128.

[4] Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. Journal of machine Learning research, vol. 3, no. Jan, pp. 993–1022.

[5] Momtazi, S. & Klakow, D. (2015). Bridging the Vocabulary Gap Between Questions and Answer Sentences. Information Processing & Management, vol. 51, no. 5, pp. 595–615.

[6] Safi-Esfahani, F., Rakian, S., & Nadimi-Shahraki, M. H. (2017). English-Persian Plagiarism Detection based on a Semantic Approach. Journal of AI and Data Mining, vol. 5, no. 2, pp. 275–284.

[7] Momtazi, S. & Lindenberg, F. (2016). Generating Query Suggestions by Exploiting Latent Semantics in Query Logs. Journal of Information Science, vol. 42, no. 4, pp. 437–448.

[8] Van Gysel, C., de Rijke, M., & Worring, M. (2016). Unsupervised, Efficient and Semantic Expertise Retrieval. Proceedings of the 25th International Conference on World Wide Web, Montréal, Québec, Canada, 2016.

[9] Celikyilmaz, A., Deng, L., & Hakkani-Tür, D. (2018). Deep Learning in Spoken and Text-Based Dialog Systems, in: Deng, L. and Liu, Y., (Eds.), Deep Learning in Natural Language Processing. Springer Singapore, Singapore, pp. 49–78.

[10] Joachims, T. (1996). A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization. Proceedings of the 14th. San Francisco, CA, USA, 1996.

[11] Steyvers, M. & Griffiths, T. (2007). Probabilistic

topic models. Handbook of latent semantic analysis, vol. 427, no. 7, pp. 424–440.

[12] Alghamdi, R. & Alfalqi, K. (2015). A survey of topic modeling in text mining. International Journal of Advanced Computer Science and Applications (IJACSA), vol. 6, no. 1.

[13] Hofmann, T. (2017). Probabilistic latent semantic indexing. ACM SIGIR Forum, vol. 51, no. 2, pp. 211–218.

[14] Kakkonen, T., Myller, N., Sutinen, E., & Timonen, J. (2008). Comparison of dimension reduction methods for automated essay grading. Journal of Educational Technology & Society, vol. 11, no. 3.

[15] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality, in: Michael I. J., Yann L. and Sara A. S. (Eds.), Advances in neural information processing systems, MIT Press, pp. 3111–3119.

[16] Campr, M. & Ježek, K. (2015). Comparing semantic models for evaluating automatic document summarization. 18th International Conference on Text, Speech, and Dialogue, Pilsen, Czech Republic, 2015.

[17] Bengio, Y., Ducharme, R., Vincent, P., & Jauvin, C. (2003). A neural probabilistic language model, Journal of machine learning research, vol. 3, no. Feb, pp. 1137–1155.

[18] Hadifar, A. & Momtazi, S. (2018). The impact of corpus domain on word representation: a study on Persian word embeddings. Language Resources and Evaluation, vol. 52, no. 4, pp. 997–1019.

[19] Le, Q. & Mikolov, T. (2014). Distributed Representations of Sentences and Documents. Proceedings of the 31st International Conference on Machine Learning, Beijing, China, 2014.

[20] Lau, J. H. & Baldwin, T. (2016). An empirical evaluation of doc2vec with practical insights into document embedding generation. Proceedings of the 1st Workshop on Representation Learning for NLP, Berlin, Germany, 2016.

[21] Christou, D. (2016). Feature extraction using latent dirichlet allocation and neural networks: a case study on movie synopses. Journal of CoRR, vol. abs/1604.01272.

[22] Ken Lang 20Newsgroup Data Set (2018), Available: http://qwone.com/~jason/20Newsgroups/.

[23] Uysal, A. K. (2016). An improved global feature selection scheme for text classification. Expert systems with Applications, vol. 43, pp. 82–92.

[24] Steinbach, M., Karypis, G., Kumar, V., et al. (2000). A comparison of document clustering techniques. KDD workshop on text mining, vol. 400, no. 1, pp. 525–526.

[25] McCallum, A. K. (2002). Mallet: A machine learning for language toolkit.

[26] Wagner, S. & Wagner, D. (2007). Comparing clusterings: an overview. Universität Karlsruhe, Fakultät für Informatik Karlsruhe, vol. 2006, no. 4

نشریه هوش مصنوعی و داده کاوی

یک مدل بازنمایی برداری معنایی توام برای خوشه‌بندی و دسته‌بندی متن

سعیده ممتازی*، آرمان رهبر، داریوش سلامی و ایمان خانی جازانی

دانشکده مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه صنعتی امیرکبیر، تهران، ایران.

ارسال ۲۰۱۸/۰۹/۲۳؛ بازنگری ۲۰۱۸/۱۱/۲۶؛ پذیرش ۲۰۱۹/۰۴/۰۷

چکیده:

خوشه‌بندی و دسته‌بندی متن از مهم‌ترین وظایف متن‌کاوی به شمار می‌روند. انتخاب ویژگی نقش مهمی در کیفیت نتایج خوشه‌بندی و دسته‌بندی ایفا می‌کند. با وجود اینکه ویژگی‌های مبتنی بر کلمه مانند بردارهای بسامد واژه - معکوس بسامد سند[1] به صورت گسترده در کاربردهای گوناگونی استفاده شده‌است، نقاط ضعف آن‌ها در به‌دست آوردن مفاهیم معنایی متن باعث شده‌است تا محققین به مدل‌های معنایی برای بازنمایی‌های برداری سند روی بیاورند. مدل‌سازی موضوعی تخصیص پنهان دریکله[2] و تعبیه عصبی سند[3] دو مورد از معروف‌ترین روش‌ها برای این منظور هستند.

در این پژوهش، ابتدا تفاوت‌های مفهومی بین این دو مدل مورد بررسی قرار می‌گیرد و نشان داده خواهد شد که دو مدل مذکور با در نظر گرفتن دو رویکرد متمایز، از دو منظر متفاوت اقدام به استخراج ویژگی‌های معنایی متن‌ها می‌کنند. سپس یک روش ترکیبی برای بازنمایی برداری سند ارائه می‌گردد که از نقاط قوت هر دو مدل بهره ببرد. نتایج آزمایش‌ها بر روی مجموعه داده‌ی 20newsgroup نشان می‌دهد که مدل پیشنهادی در قیاس با مدل‌های پایه، هم برای خوشه‌بندی و هم برای دسته‌بندی متن بهتر عمل می‌کند. مدل پیشنهادی، به صورت میانگین، ۲٫۶٪ در امتیاز F[4] برای خوشه‌بندی و ۲٫۱٪ در امتیاز F برای دسته‌بندی متن نسبت به مدل‌های پایه، بهتر عمل می‌کند.

کلمات کلیدی: متن‌کاوی، بازنمایی معنایی، مدل‌سازی موضوع، تعبیه عصبی متن.

[1] Term Frequency – Inverse of Document Frequency (TF-IDF)
[2] Latent Dirichlet Allocation (LDA)
[3] Neural document embedding
[4] F-Measure