Expert Systems with Applications 162 (2020) 113770

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Tens-embedding: A Tensor-based document embedding method

Zahra Rahimi, Mohammad Mehdi Homayounpour *

Department of Computer Engineering and Information Technology, Amirkabir University of Technology, No. 350, Hafez Ave, Valiasr Square, Tehran, Iran

ARTICLE INFO

Article history:
Received 18 July 2019
Revised 15 July 2020
Accepted 16 July 2020
Available online 26 July 2020

Keywords:
Natural language processing
Text classification
Text representation
Document embeddings
Tensor factorization
Topic modeling

ABSTRACT

A human is capable of understanding and classifying a text but a computer can understand the underlying semantics of a text when texts are represented in a way comprehensible by computers. The text representation is a fundamental stage in natural language processing (NLP). One of the main drawbacks of existing text representation approaches is that they only utilize one aspect or view of a text e.g. They only consider texts by their words while the topic information can be extracted from text as well. The term-document and document-topic matrix are two views of a text and contain complementary information. We use the strength of both views to extract a richer representation. In this paper, we propose three different text representation methods with the help of these two matrices and tensor factorization to utilize the power of both views. The proposed approach (Tens-Embedding) was applied in the tasks of text classification, sentence-level and document-level sentiment analysis and text clustering wherein the conducted experiments on 20newsgroups, R52, R8, MR and IMDB datasets indicated the superiority of the proposed method in comparison with other document embedding techniques.

1. Introduction

Nowadays, text classification is one of the main tasks in natural language processing (NLP). In a classification task to achieve a high performance, a classifier has to be able to distinguish between documents from different classes. Therefore documents are compelled to be represented in ways comprehensible and distinguishable by a computer. The quality of document representation has a great impact on the performance of natural language processing tasks including text classification, sentiment analysis, document clustering, and machine translation. Document embedding can be useful in a text-based intelligent system. A computer which can extract appropriate information from text corpora is a text-based intelligent system. Some of the applications of a text-based intelligent system are as follows:

- Text extraction: This task converts an unstructured text to more structured one.
- Intelligent retrieval system: retrieves document related to the user's queries.
- Summarization: makes a summary from important parts of a text
- Text classification: tries to assign one of the pre-defined classes to a given text

The way we define a text for a computer has a significant impact on all of the above applications. A central question in document representation is the manner by which a representation reveals the semantics and hidden patterns of a document and the extent to which it is instructive to NLP task.

There is an abundance of methods in the literature for document representation. These approaches can be divided into five categories: Bag of words (BOW) methods, topic modeling-based methods, word embedding-based approaches, neural-network-based approaches and graph-based methods. These categories and some of their representative models are described in summary in this section and with more details in the section of related works. Bag of words methods show the presence and absence of a term in a document by a weighting scheme (Fu, Qu, Huang, & Lu, 2018; Kim, Kim, & Cho, 2017; Li, Mao, Xu, Li, & Zhang, 2020; Salton & McGill, 1987; Zhao & Mao, 2018). These models usually are fast and simple but they have a major shortcoming that cannot show the semantic attributes of a document as well as the relationships between words.

Topic modeling approaches (Blei, Ng, & Jordan, 2003; Das, Zaheer, & Dyer, 2015; Deerwester, Dumais, Furmas, Lander, & Harshman, 1990; Hofmann, 1999; Jiang, Li, Chen, Member, & Cohn, 2018) model the thematic structure of documents using hidden variables which are called topics. These methods often

* Corresponding author.
E-mail addresses: zah-ra@aut.ac.ir (Z. Rahimi), homayoun@aut.ac.ir (M.M. Homayounpour).

https://doi.org/10.1016/j.eswa.2020.113770
0957-4174/

represent a document with more meaningful features than BOW and put documents with similar topic distribution near each other in topic space and have a good generalization power for text classification task (Chemudugunta, Smyth, & Steyvers, 2007). Due to the transfer of documents into topic space, some important features such as rare words in each document are ignored.

Following the appearance of word embeddings in the field of natural language processing, various document representation methods are proposed to utilize the ability of these models which put semantically related words near each other in a vector space. The word embedding-based document embedding models (Kamkarhaghighi & Makrehchi, 2017; Kim et al., 2017; Li et al., 2020; Sinoara, Camacho-Collados, Rossi, Navigli, & Rezende, 2019) capture the semantic of documents better than the bag of words method but most of them use only the average (or max or min) of word embeddings which cannot show the whole concepts of the document (Fu et al., 2018). These methods often employ only one aspect of the document which is a word-word co-occurrences and usually use of local context therefore miss the topical information.

Neural network based methods (Chen, 2017; Kim, 2014; Le & Mikolov, 2014; Liu, Zhao, & Volkovs, 2017; Rao, Huang, Feng, & Cong, 2018) employ neural networks such as long short term memory (LSTM), convolutional neural network (CNN) and feed forward neural networks to represent documents. These models catch the semantics of documents properly and take the order of words into account but they are hard to train (Fu et al., 2018) and most of them only consider the first-level co-occurrence of words in a local context and ignore other useful information such as topics. Graph–based models (Bijari, Zare, Kebriaei, & Veisi, 2020; Sen, Ganguly, & Jones, 2019; Yao, Mao, & Luo, 2018) uses graph structures to generate document representation. These models are also hard to train and ignore the topical information.

As it is mentioned before, one of the shortcomings of previous approaches is using only one aspect of text to generate document embedding while the text contains more information (such as topic information) than simply word-word co-occurrences. Another shortcoming is using only local co-occurrences to extract document embeddings while the global information (document level information) would be useful in generating a richer representation of the text. The salience of global context in document representation lies in its ability to signify topical information and language comprehension and acquisition (Huang, Socher, Manning, & Ng, 2008). To get over these limitations of previous approaches we propose the Tensor-based Document Embedding (Tens-embedding). The proposed method is a kind of multi-view document embedding in which the multiple views correspond to various aspects of information such as term-document co-occurrences or topic information which can be extracted from the text.

The term-document matrix shows the co-occurrence of words and documents. The element Tij of the term-document matrix shows the frequency of word "i" in document "j". The counts in this matrix are extracted based on global context i.e. the context is whole document. The column j of this matrix shows a representation for jth document and the row i is a representation for ith word. In this document representation, every word in the vocabulary is important and effective in the representation.

On the other hand, solely considering the frequency of words in documents is inadequate for discriminating documents in a suitable fashion. It provides us with a very specific view of the documents while sometimes a more general view such as a topical view can add more meaningfulness to our interpretation of a document. For example, words such as 'president' or 'government' are very frequent in a topic like 'politics', whereas in the case of 'travel' as a topic, words like 'hotel' and 'airport' are more prevalent. But the frequency of the word 'budget' might be the same in both topics. Topic information is the chief factor as it contains more discriminative information than the frequency of words in a document. As mentioned in LDA each document is a mixture of topics and each topic is a mixture of words. LDA learns topics based on higher-level word co-occurrences on a document level (Blei et al., 2003). Thus there are two views towards each document; a view which only examines the documents and words and a more abstract view which scrutinizes the documents as well as the topics.

These two views are complementary. The document-topic representation which is generated by LDA is an abstract view of the document and increases the generalization power of document retrieval and text classification (Chemudugunta et al., 2007), but at times misses certain pivotal features (words) of the document. On the contrary, in the term-document matrix representation, each word of the document matters on its own as it is a very specific view of the document. However, this level of specificity puts the generalization ability of natural language processing tasks at risk. Besides, in a term-document matrix only first order co-occurrences of words are considered, whereas LDA can extract the higher order co-occurrences as well.

Aiming to exploit the benefits of both these views, this study seeks to combine the proposed views via a tensor, which then undergoes vector factorization in order to generate the document representation. Tensors act as the hefty tool for combining the views, meanwhile tensor factorization is employed to extract the hidden aspects of data (Kolda & Bader, 2009).

The main contributions of this paper are as follows:

- Proposing a new unsupervised multi-view document embedding method. The views in this paper are abstract (topical) and specific (term-based) views of a document
- Utilizing the advantage of both abstract (topical) and specific views of a document with the help of a tensor.
- Utilizing the power of tensor factorization for revealing the hidden aspects of data to extract document embeddings
- Suggestion of three different methods to construct a tensor using term-document and document-term matrices.
- Performing various experiments on multiple datasets to show the superiority of the model on natural language processing tasks including text classification.

This paper is organized as follows. Section 2 briefly introduces related works in the literature. The proposed approach is explained in Section 3. Section 4 provides the experimental results and finally section 5 concludes the paper.

2. Related works

As it is mentioned in Introduction section, in the literature document embedding methods can be categorized into five categories: bag of words, Topic modeling based approaches, word embeddings based models, graph-based methods and neural-network-based approaches which are described here with more details.

2.1. Bag of words methods

One of the most straightforward yet significantly convenient methods for document representation is the Bag of Words (BOW) approach (Harris, 1954). In BOW, each document is represented as a vector of word counts, wherein the length of the vector is equal to the size of the vocabulary, and each element shows the frequency of the corresponding word in the vocabulary. Put differently, if the m_{th} word in the vocabulary appears N times in a

document, the m_{th} element in the coinciding BOW vector is equal to N. BOW is a good representation for linear classifiers but fails in cases where the vocabulary size is large or in the presence of a high number of classes or short texts, which causes a very sparse BOW vector and therefore reduces accuracy (Bojanowski et al., 2016). Moreover, BOW is incapable of representing the semantic attributes of a document as well as the relationship between the most relevant terms of a document (Garcia, Rodriguez, Ferro, & Rifon, 2016). Another approach is to represent a text via Tf-Idf (Salton & McGill, 1987). This approach is more preferable than the BOW method; however it is still quite simplistic. To overcome the mentioned shortcomings of BOW, (Zhao & Mao, 2018) propose a fuzzy BOW model (Zhao & Mao, 2018) that uses cosine similarity between word embeddings to measure the semantic correlation between words. (Lakshmi & Baskar, 2019) propose two novel techniques of token weighting with the help of fuzzy logic: a weighting scheme based on each term and its rank in a document and a weighting scheme based on a term and its semantically related terms. Bag of Meta words (Fu et al., 2018) is another document representation approach, in which documents are represented as a bag of meta words. Meta words are independent semantic information of textual contents and are generated by the agency of pre-trained word embedding. This method could potentially rectify the two main issues regarding neural networks and average of word embeddings as mentioned in introduction section. (Wei, Guo, Chen, Tang, & Sun, 2019) introduce a method to represent a document as a conditional co-occurrence matrix. This matrix is not symmetric as usual co-occurrence matrix was and can capture more semantic information from a document than usual co-occurrence matrix.

2.2. Topic-modeling-based methods

Equally important are methods which proceed to represent different documents via more distinguishing features for text classification including topic modeling approaches such as latent semantic indexing (LSI) (Deerwester et al., 1990), which can be used to achieve such distinctive features. LSI works by performing singular value decomposition (SVD) on the Term-Document matrix, and thereby maps the documents to a new space, where documents with similar topics are situated near each other. LSI attempts to minimize the reconstruction error and detects the most representative features rather than the most discriminative ones. Therefore, LSI is somewhat unsuitable for discriminating documents containing different semantics and is not a particularly expressive model. Building on LSI, probabilistic LSI (PLSI) (Hofmann, 1999) was proposed to compensate for the meager expressiveness of the LSI. Each document in PLSI consists of multiple topics and each word is assigned its own topic. PLSI is a generative model which seeks to minimize perplexity. PLSI, however, incorporates a rather large assortment of parameters and fails to avoid over-fitting in the case of inadequate data. A further issue of PLSI is the absence of probabilistic models for representing the corresponding topics. To overcome this concern, the latent Dirichlet allocation (LDA) method was proposed in 2003 (Blei et al., 2003) wherein unlike the PLSI, every topic is a probability distribution over words. LDA is a generative probabilistic model, which considers each document as a mixture of latent topics and each topic as a distribution over words. The outputs of LDA are θ and ϕ. θ is the distribution of topics over documents and ϕ is the distribution of words over topics.

Latent topic text representation (LTTR) (Jiang et al., 2018) is a text representation method, which assumes that words belonging to the same topic have a Gaussian distribution and documents can be recast as Gaussian mixtures. Gaussian LDA (Das et al., 2015) assumes words which belong to the same topic and have a Gaussian distribution, however, it does not account for words belonging to different topics.

2.3. Word embedding-based approaches

Following the advent of word embedding models (Mikolov, Chen, Corrado, & Dean, 2013; Pennington, Socher, & Manning, 2014), certain methods were proposed in order to take advantage of word embeddings for document representation. Word embeddings methods map each word to a vector space. In the vector space the semantic of words are the salient elements and words with similar topics, such as *student* and *school,* are less distant from each other as opposed to words with disconnected topics, like *school* and *cat*. In certain models, a document is considered as a sum or average of its word embeddings (Bollegala, Mu, & Goulermas, 2016; Maas et al., 2011), albeit this kind of representation fails to account for the order of words in a document which can change the semantic of documents. Only an average representation is incapable of conveying the overall meaning of the document (Fu et al., 2018). Word embedding models like Glove and word2vec are trained on general-purpose corpora and thus the pre-trained word vectors ignore the local-context for specific tasks (Kamkarhaghighi & Makrehchi, 2017). In order to overcome this problem, Kamkarhaghighi and Makrehchi (Kamkarhaghighi & Makrehchi, 2017) proposed to incorporate content tree models. The model is based on the correlation between terms in a document and attempts to change the pre-trained word embedding.

Evidently, finding the appropriate document representation which could display further semantic information is of utmost importance. For this reason, (Sinoara et al., 2019) have sought to represent a document by means of word embeddings in conjunction with external resources such as the Babelfy (Flati & Navigli, 2014) and the Nasari (Camacho-Collados, Pilehvar, & Navigli, 2015) algorithms. The intriguing feature of this approach is that it incorporates both word senses as well as word vectors. This approach can be applied on several languages, simultaneously. Another class of methods consists of bag of concepts (BOC) methods. These methods (Bing, Jiang, Lam, Zhang, & Jameel, 2015; Kim et al., 2017; Li et al., 2020) consider each document as a bag of concepts, where each concept is represented as a unit of meaning.

2.4. Graph-based methods

Graph-based approaches are another sort of text representation techniques. Representations generated from these approaches are far more informative than others, as they consider both structural and semantical information (Dourado, Galante, Gonçalves, & da Silva Torres, 2019). Recently (Dourado et al., 2019) have proposed a graph based approach for document representation. In this approach a co-occurrence graph for each document is generated, at which point subgraphs are extracted and certain vocabulary are generated as the basis of the new vector space. Another method (Yao et al., 2018) uses graph convolutional networks (Kipf & Welling, 2017) to represent documents. In this method a network is generated based on word co-occurrences and word-document co-occurrences. A graph convolutional network is then learnt on the target corpus. (Bijari et al., 2020) propose a graph-based document modeling approach for sentence-level sentiment analysis. This method constructs a graph of words with word-word co-occurrences and applies node embeddings for representation learning.

2.5. Neural-network-based approaches

Embedding of words in a document could also be utilized as input to a neural network such as recurrent or convolutional neural

networks. These neural nets are able to generate representations which contain more semantical or syntactical information compared to the average of embeddings. Nevertheless, the training of neural networks is both hard and time consuming (Fu et al., 2018). Text representation by neural networks is another line of research which has attracted a significant amount of attention in recent years. One such approach is the Paragraph-Vector method (Le & Mikolov, 2014), which is quite analogous to the word2vec technique. In this approach each word and each paragraph is assigned a unique vector. The vectors are learned by a feed forward neural network with a single hidden layer. The paragraph vector and word vectors are then concatenated in order to predict the next word. However, Paragraph-Vector is overly expensive to train and test. In 2013 recursive neural networks were used by (Socher et al., 2013) for document representation. Another approach presented by (Rao et al., 2018) aims to consider the intrinsic relations between sentences in a document. By this token, a two layer long short term memory (LSTM) neural network is applied. The first layer extracts the sentence representation while the second layer extracts the document representation. This method considers the order of words in a document, albeit the inference of recurrent neural network is time consuming (Liu et al., 2017). In another approach, (Liu et al., 2017) convolutional neural networks were used for resolving the issue of excessive inference time of recurrent neural networks (RNNs). CNNs are parallelizable and so are superior to RNNs in terms of time complexity. Along these lines, (Bojanowski et al., 2016) consider each document as a sequence of characters and a one-dimensional neural network is used to generate a representation for text. This approach requires a substantial amount of data for training, which is one of the main shortcomings of this model. In (Kim, 2014), a 2-D convolutional neural network is applied for generating a representation for each sentence. In this approach each sentence is presented as a matrix. Each row of this matrix is a vector representation for a word in the sentence. The convolutional neural network in this method consists of four layers including input, convolution, max-pooling and soft-max layers. This method uses row-wise filters with varying widths. Lai, Xu, Liu, and Zhao (2015) use recurrent convolutional neural network for text representation. The recurrent structure is used to achieve long-distance contextual information with a max-pooling layer added to identify words more valuable and effective in text classification. In (Chen, 2017), specific words of a document labelled as irrelevant for classification are deleted. Therefore, the parameters of this model are fewer in comparison with the Doc2Vec model. This model also proposes a regularization term that acts in favor of rare and informative words of documents. Most of the pooling methods for document representation use fix L^1 or L^∞norms therefore they cannot generate good text representations (Wu, Wu, Qi, Cui, & Huang, 2020). (Wu et al., 2020) recently proposed a method which utilizes attentive pooling with trainable norms to generate a suitable representation for a document.

The methods mentioned in this section in addition to their pros and cos are summarized in Table 1.

3. Proposed method

There are several types of information which can be extracted from a text corpus including but not limited to topics and term-document co-occurrences. Each of these factors can be potentially exploited as a means for document representation. Stated differently, each factor is a different aspect of a text. A document can

Table 1
Summary of literature review.

Methods	Pros	Cons
Bag of words methods: each document is represented as a vector of word counts, wherein the length of the vector is equal to the size of the vocabulary, and each element shows the frequency of the corresponding word in the vocabulary (Fu, Qu, Huang, & Lu, 2018; Kim, Kim, & Cho, 2017; Li, Mao, Xu, Li, & Zhang, 2020; Salton & McGill, 1987; Zhao & Mao, 2018)	• Very simple and fast • interpretable by humans • suitable for linear classifiers	• Fails to account for the order of words • Vectors are very sparse and curse of dimensionality can happen • BOW is incapable of representing the semantic attributes of a document as well as the relationship between the most relevant terms of a document • Only considers the number of times a word occurs in a document and does not consider the number of times a word occurs in the whole corpus
Topic modeling-based approaches such as LSI, LDA, PLSI: These methods consider probabilistic model for modeling of documents. (Blei et al., 2003; Das et al., 2015; Deerwester et al., 1990; Hofmann, 1999; Jiang et al., 2018)	• Puts documents with the same topic near each other • Provides a good generalization power for text classification task	• Fails to account for the order of words • The appropriate number of topics are challenging • Due to transfer of documents into topic space, some important features such as rare words in that document are ignored. • Only considers the co-occurrence of words in a global context • Presents documents only by topics and do not consider the words. Therefore ignores them as valuable features of text
word embedding based methods (Kamkarhaghighi & Makrehchi, 2017; Kim et al., 2017; Li et al., 2020; Sinoara et al., 2019)	• Simple and fast • consider the semantics of documents more than BOW models	• Does not consider the order of words • Ignores the local contexts for a specific task because the pre-trained word vectors are general purpose • Only a sum or average vector cannot show the semantic of a document properly
Neural network based approaches These approaches tries to generate representations of documents with the help of a neural network such as Convolutional neural networks, Recursive neural networks, Recurrent neural networks (RNN) or Long short-term memory (LSTM) (Chen, 2017; Kim, 2014; Le & Mikolov, 2014; Liu et al., 2017; Rao et al., 2018; Wu et al., 2020)	• Capture the semantics of documents properly • takes the order of words into account specially in RNN or LSTM networks	• Hard to train • Most of them only consider the first-level co-occurrence of words in a local context • Only considers documents as a sequence of words and do not use other useful features such as topics
Graph-based approaches consider each document as a graph of words (Bijari, Zare, Kebriaei, & Veisi, 2020; Sen et al., 2019)	• Capture the semantics of documents by the co-occurrence graph for each document	• Hard to train • Fails to account for word-order • Consider only one aspect of a text • Do not consider the topics or other useful features which can be extracted from a text

Table 2
Statistics of R52 dataset.

Classes	Number of documents	Average number of words per document	Classes	Number of documents	Average number of words per document
acq	2292	118.24	**Jobs**	49	109.5
alum	50	138.06	**Lead**	8	123.25
bop	31	147.55	**Lei**	14	99.93
carcass	11	172.9	**Livestock**	18	133.6
cocoa	61	206.23	**Lumber**	11	175.64
coffee	112	199.1	**Meal-feed**	7	135.9
copper	44	151.68	**Money-fx**	293	161.9
cotton	24	131.17	**Money-supply**	151	96.7
cpi	71	102.4	**Nat-gas**	36	140.56
cpu	4	87.5	**nickel**	4	202.75
crude	374	187.3	**orange**	22	180.41
dlr	6	168.17	**Pet-chem**	19	132.58
earn	3923	65.64	**platinum**	3	191.33
fuel	11	101.9	**potato**	5	117.6
gas	18	166.6	**reserves**	49	105.67
gnp	73	219.0	**retail**	20	173.1
gold	90	143.8	**rubber**	40	213
grain	51	183.8	**ship**	144	149.9
heat	10	93.1	**Strategic-metal**	15	161.7
housing	17	92.94	**sugar**	122	167.6
income	11	99.7	**tea**	5	187.8
Install-debt	6	97	**tin**	27	236.85
interest	271	118.7	**Trade**	326	234.46
ipi	44	139.4	**Veg-oil**	30	172.53
Iron-steel	38	146.8	**wpi**	23	105.35
jet	3	44.3	**zinc**	13	87.9

be represented solely by their individual topics extracted by methods like LSI (Deerwester et al., 1990) or LDA (Blei et al., 2003). In this case, documents are mapped to a latent space which is more abstract than the bag of words technique and permits a higher rate of generalization in terms of document retrieval or classification tasks (Chemudugunta et al., 2007). However, topic models tend to over-generalize, causing certain important features to be disregarded in the latent space. This can be further elaborated in document retrieval. Consider the query sea + forest + Avicenna. The goal is to match up this query with a set of crawled documents. Using LDA to represent the documents and query can lead to a loss of information. Given that Avicenna is a rare word, it is very likely that the retrieved documents may not include Avicenna. With this representation, documents relating to sea or forest are given higher priority (even if they do not include Avicenna). In this case, the specific word Avicenna, which is not frequent enough to appear amongst the most probable words of any topic, is lost. On the contrary, word-based techniques such as term frequency-inverse document frequency (tf-idf) are overly specific in terms of matching words (Chemudugunta et al., 2007). Besides, in LDA, the second-order co-occurrences are taken into account, while tf-idf or BOW only includes first-order co-occurrences.

Aspiring to reap the benefits of both approaches, this study uses a tensor representation to combine the information extracted from the term-document matrix alongside information extracted from topic-document matrix. Thereupon, tensor decomposition is used to extract representation of documents.

The overall view of the proposed approach is shown in Fig. 1. This approach contains two main steps (A) tensor construction and (B) tensor decomposition. Step A is itself comprised of two phases including creation of document-document matrix by term-document matrix information and creation of the document-document matrix by document-topic information. Step 2 can be done in three schemes: applying a soft clustering on rows of document-topic matrix, applying k-means to cluster rows of document-topic matrix or calculating the similarity of rows using cosine similarity.

3.1. Preliminaries and notation

Tensors are essentially represented as n-way or multi-dimensional arrays (Kolda & Bader, 2009). A one-way array is a vector, a 2-way array is a matrix, and an array with more than two ways is called higher order tensor. Scalars are denoted by lowercase letters. Vectors are denoted by bold lowercase letters. Matrices are denoted by uppercase letters and higher dimensional tensors are denoted by Euler scripts letters, for instance $\mathscr{A}$. The i-th entry of vector a is denoted by a_i. The (i,j) th element of matrix A is denoted by a_{ij} and (i, j, k)th element of tensor $\mathscr{A}$ is denoted by a_{ijk}.

Slices are two-dimensional segments of a tensor (Kolda & Bader, 2009), which are defined by fixing all indices except two target indices. Slices can be categorized into three types: Frontal, horizontal and lateral slices. $\mathscr{A}_{::k}$,$\mathscr{A}_{i::}$, $\mathscr{A}_{:j:}$ are the kth, ith and jth frontal, horizontal and lateral slices of tensor $\mathscr{A}$ respectively. $\odot$ is the outer product of two matrix. $\circledast$ is the Hadamard product of two matrix. DD is the document-document matrix. DT is the document-topic matrix and DW is the document-word matrix.

3.1.1. Tensor matricization

Tensor matricization is the process of converting a tensor to a matrix. There are n possible matricizations for an n-mode tensor. The nth mode matricized form is shown by $\mathscr{A}_{(n)}$. In this representation, the columns of the matrix are constructed by the mode-n fibers of the tensor. A fiber is obtained when all the indices except one of them are kept constant (Kolda & Bader, 2009). There are three kinds of fibers available in a 3-mode tensor: Mode-1, Mode-2, and Mode-3 fibers.

3.2. Tensor construction

Tensors are a good mathematical foundation to combine multiple views and tensor factorization can be employed to reveal the hidden aspects of data (Kolda & Bader, 2009). One way to construct

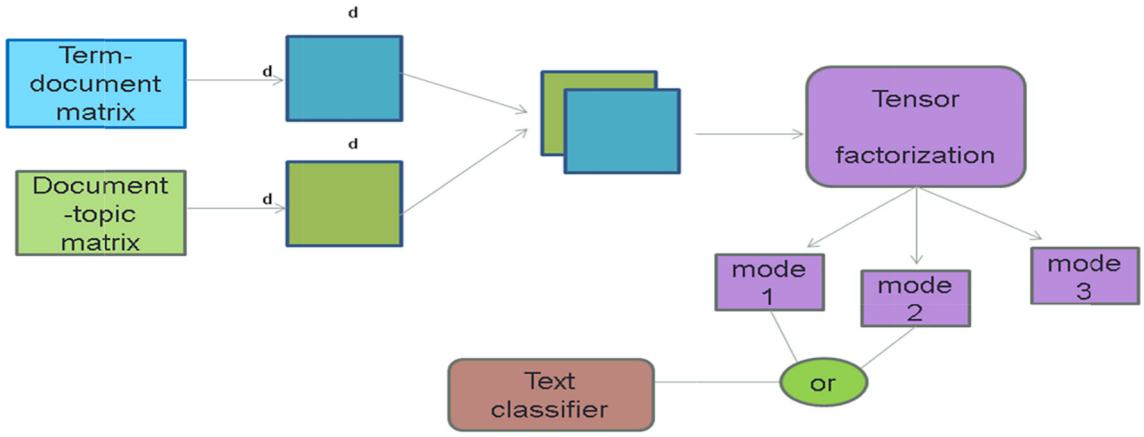

Fig. 1. Schematic view of proposed method, d shows the number of documents.

a tensor with available views is to incorporate the help of frontal-slices. Each frontal slice is made up of information from one of the views. This process is depicted in Fig. 2.

There are two kinds of information: information which is obtained from the term-document matrix and information which is extracted from the topic-document matrix. We can construct a 3-way tensor with size $d \times d \times 2$, where d is the number of documents. There are two $d \times d$ frontal slices in this tensor. One can be constructed by Term-document information and the other can be made up of document-topic information.

3.2.1. Using term-document matrix to construct a frontal slice

Term-document matrices can be extracted from a corpus, and each entry of this matrix shows the frequency of a term in a document. The following elaborates how one can find a relationship between documents using this matrix. A document-document matrix can be generated using the term-document matrix using the following formula:

$$DD = WD^T \times WD \tag{1}$$

where DD is the document-document matrix and WD is the term-document matrix and WD^T is the transpose of the term-document matrix. Each entry of DD shows the number of shared words between pairs of documents. An example of this process is shown in Fig. 3.

3.2.2. Construction of a frontal slice using document-topic matrix

The output of LDA is a document-topic matrix. A document-document matrix, on the other hand, can be generated in one of three possible ways:

Soft-clustering

Documents can be clustered based on topic distribution over each document. The number of clusters is equal to the number of topics in this case. The cluster for each document is determined based on the topic with the highest probability in the topic distribution of the corresponding document. Following, every two documents are considered as members of one cluster and can be considered as co-occurring documents. Accordingly, if DD is the co-occurrence matrix of documents and DD_{ij} is an element of this matrix, showing the co-occurrence count between the ith and the jth document, then:

$$DD_{ij} = \begin{cases} 0 \; i \wedge j \; are \; members \; of \; the \; same \; cluster \\ 1 \; i \wedge j \; are \; not \; members \; of \; the \; same \; cluster \end{cases} \tag{2}$$

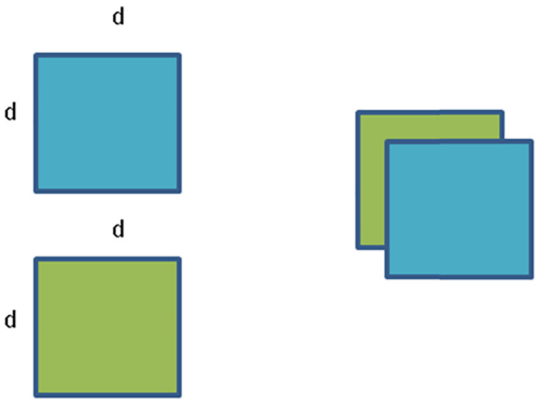

Fig. 2. Construction of a tensor by frontal slices, d shows the number of documents.

This co-occurrency is implemented by the dint of topics, whereby documents belonging to the same topic are considered as co-occurring documents. This process is depicted in Fig. 4.

3.2.3. K-means

Each row of the document-topic matrix can be considered as a vector representation for the corresponding document. These vectors can be clustered using K-means clustering algorithm. Each row of the document-topic matrix is a multinomial distribution, which in a way corresponds to an assortment of compositional data. As K-means is applied on real coordinates, each row must be mapped into real coordinates, prior to applying K-means. One form of mapping is the isometric log-ratio transformation (ilr) (Egozcue, Pawlowsky-Glahn, Mateu-Figueras, & Barceló-Vidal, 2003). This transformation maps compositional data from Aitchison simplex to the real coordinate space. This transformation is isometric and so preserves all metric properties.

$$ilr : S^D \rightarrow \mathbb{R}^{D-1} \tag{3}$$

where $S^D$is the Aitchison simplex.

Following the application of ilr, every two documents in a cluster can be considered as co-occurred documents. Accordingly, if DD is the co-occurrence matrix of documents and DD_{ij} is an element of this matrix which shows the co-occurrence count between the ith and the jth document, then:

$$D_{ij} = \begin{cases} 1 \quad i \wedge j \; are \; members \; of \; the \; same \; cluster \\ 0 \quad i \wedge j \; are \; not \; members \; of \; the \; same \; cluster \end{cases} \tag{4}$$

	Term1	Term2	Term3
Document1	2	1	0
Document2	0	1	1

×

	Document1	Document2
Term1	2	0
Term2	1	1
Term3	0	1

=

	Document1	Document2
Document1	0	1
Document2	1	0

Fig. 3. Construction of a frontal slice with the help of term-document matrix.

Fig. 4. Using the document-topic matrix to construct a frontal slice, di shows the ith document and ti shows the ith topic.

The two $d \times d$ frontal slices are then generated accordingly.

Tensor multiplication

A document-document matrix can be generated using the document-topic matrix as follows:

$$DD = DT^T \times DT \tag{5}$$

where DD is the document-document matrix and DT is the document-topic matrix and DT^T is the transpose of the document-topic matrix. Each element of DD is then divided by the product of the norm of corresponding documents. This process is formulated as Eq. (6).

$$DD_{ij} = \frac{DD_{ij}}{\|D_i\|_2 \|D_j\|_2} \forall_{i,j} \tag{6}$$

where DD_{ij} is the one element of DD matrix corresponding to the ith and the jth documents. D_i is the ith document and D_j is the jth document. Cosine similarities between rows of the document-topic matrix can also be calculated using Eqs. (5) and (6). In this case, the document-document matrix is a similarity matrix.

3.3. Tensor decomposition

Canonical Decomposition/Parallel Factors (CP)

CP factorizes a tensor into a sum of rank-one tensors (Kolda & Bader, 2009). Let $\mathscr{D} \in \mathbb{R}^{I \times J \times k}$ be a 3-order tensor then this factorization is formulated as follows:

$$\min_{\widehat{\mathscr{D}}} \|\mathscr{D} - \widehat{\mathscr{D}}\|, \ \widehat{\mathscr{D}} \approx \sum_{r=1}^{R} a_r \circledcirc b_r \circledcirc c_r \tag{7}$$

where R is the rank of $\mathscr{D}$. $a_r \in \mathbb{R}^I$, $b_r \in \mathbb{R}^J$ and $c_r \in \mathbb{R}^K$ for $r = 1, \cdots, R$.

Factor matrices refer to those matrices which contain the rank-one components in their columns i.e. $A = [a_1 a_2 a_3 \cdots a_R]$ and likewise for B and C.

3.4. Factor matrix as a representation

The size of our tensor is $N_d \times N_d \times 2$, where N_d is the number of documents in the training set and the number of views is two. A and B are defined as matrices of size $N_d \times r$, where r is the tensor rank considered as the embedding dimension of each document. Each row of A or B is a representation for a document. The sum and concatenation of A, B can be used for the representation of the documents as well.

3.5. Vector extraction for test and unseen data

If the representation of documents is produced with the approach above, then the representation of unseen documents can be extracted without the need for repeating the whole process with new data. In this case, factors which are derived in the previous step are applied to generate the representation for unseen or test data.

Let $\mathscr{D}_{train}$ be a document tensor for training, and A, B, and C be the factors which are derived by CP decomposition of $\mathscr{D}_{train}$, then according to (Kolda & Bader, 2009), $\mathscr{D}_{train}$ can be written in mode-1 and mode-2 matricized forms as the following Equations:

$$\mathscr{D}_{train(1)} = A(C \odot B) \tag{8}$$

$$\mathscr{D}_{train(2)} = B(C \odot A) \tag{9}$$

where $\odot$ is the Khatri-Rao product. Eq. (7) can be written as below in mode-1 and mode-2 matricized form (Kolda & Bader, 2009):

$$\min_{A} \|\mathscr{D}_{train(1)} - A(C \odot B)\|_F \tag{10}$$

$$\min_{B} \|\mathscr{D}_{train(2)} - B(C \odot A)\|_F \tag{11}$$

In the test time, the $\widehat{A}$ and $\widehat{B}$ are calculated for $\mathscr{D}_{test}$. Eqs. (12) and (13) can be used for extracting $\widehat{A}$ and $\widehat{B}$ as follows:

$$\min_{\widehat{A}} \|\mathscr{D}_{test(1)} - \widehat{A}(C \odot B)^T\|_F \tag{12}$$

$$\min_{\widehat{B}} \|\mathscr{D}_{test(2)} - \widehat{B}(C \odot A)^T\|_F \tag{13}$$

Eqs. (12) and (13) produce exact answers which can be shown as Eqs. (14) and (15), respectively:

$$\widehat{A} = \mathscr{D}_{test(1)}(C \odot B)\left(C^T C \otimes B^T B\right)^{\dagger} \tag{14}$$

$$\widehat{B} = \mathscr{D}_{test(2)}(C \odot A)\left(C^T C \otimes A^T A\right)^{\dagger} \tag{15}$$

where $\otimes$ is the Hadamard product.

4. Experiments

This section provides the explanation of datasets, short definition of baseline methods and experimental results of the proposed model on text classification, sentiment analysis and document clustering tasks.

4.1. Data sets

Text classification is a task of natural language processing which undertakes the task of classifying certain documents into their respective categories. The performance and appropriateness of generated document representations are investigated in the text classification task. Highly accurate representations can be exploited to extract discriminative features for each document.

The datasets used in this study for text classification include 20newsgroups[1] (20NG), R52[2] and R8[3]. All datasets were preprocessed in order to remove punctuations and stop words. In the case of the 20NG dataset, all headers and footers were removed as well as they generally contain specific information about classes.

The R8 dataset is a part of the larger Reuter-2157 dataset, wherein the documents are categorized into 8 different classes, with a total of 7674 documents. R8 contains 5485 documents for training and 2189 documents for testing. Statistics of this dataset are shown in Table 3. The 20-newsgroups dataset is a set of newsgroups contents containing 18,818 documents from 20 classes. All the headers, footers and quotes were deleted. Further details are shown in Table 4. The R52 is part of the Reuter-21578 dataset, comprised of 9100 documents from 52 classes. 6532 documents from this dataset were used for training and the remaining for testing. Some statistics of this dataset are shown in Table 2.

Sentiment analysis is a natural language processing tasks which specifies the polarity of a document. The datasets we used for sentiment analysis are Movie review (MR) [4](Pang & Lee, 2005) and large movie review (IMDB) (Maas et al., 2011) [5]. MR which is a dataset for sentence-level sentiment analysis contains 10,662 short movie reviews. 7108 reviews were used for training and the remaining for testing. IMDB which was employed for document-level sentiment analysis contains 50,000 movie reviews. 25,000 documents are used for training and remaining for testing. The statistics of these two datasets are shown in Table 5.

Table 3
Statistics of R8 dataset.

Classes	Number of documents per class	Average number of words per document
acq	2292	118.24
crude	374	187.3
earn	3923	65.64
grain	51	183.8
interest	271	118.6
Money-fx	293	161.9
ship	144	149.9
trade	326	234.46

4.2. Methods compared with the proposed approach

The following document embedding methods were compared with our proposed method:

- **TextGcn** (Yao et al., 2018): a graph based approach which uses graph convolution
- **Bag of concepts** (Kim et al., 2017): a concept-based approach, wherein the concepts are generated by clustering pre-trained word embeddings which are then used to obtain document representations.
- **CNN** (Kim, 2014): Employs a convolutional neural network to generate sentence embeddings.
- **CNN-non-static** (Kim, 2014): this method is the same as CNN, with a slight difference that it changes one channel of the input during back-propagation. Therefore, the word embeddings are fine-tuned during training.
- **CNN-pad** (Liu et al., 2017): an unsupervised document modeling using convolutional neural network
- **Paragraph-Vector** (Le & Mikolov, 2014): paragraph-Vector uses a 3-layer feed-forward neural network to learn document embedding. This method is similar to the CBOW, however, it contains the documents' information as well.
- **TWE** (Liu, Liu, Chua, & Sun, 2015): this is an acronym for topical word embedding (Liu et al., 2015). This approach works in similar to the CBOW, with the exception that the neural network inputs are both topics and words. Besides the embeddings are generated for both topics and words.
- **LTTR** (Jiang et al., 2018): a text representation method, which assumes that words belonging to the same topic have a Gaussian distribution and documents can be recast as Gaussian mixtures.
- **Gaussian LDA** (Das et al., 2015): this approach assumes words which belong to the same topic and have a Gaussian distribution, however, it does not account for words belonging to different topics.
- **LSTM** (Liu, Qiu, & Huang, 2016): in this method, an LSTM network is used, in which the final hidden state is used as the representation of the corresponding document.
- **Bi-LSTM** (Kipf & Welling, 2017): employs a Bi-LSTM network for document classification, accepting pre-trained word embeddings as input.
- **Content-tree** (Kamkarhaghighi & Makrehchi, 2017)**:** a post-processing model for improving word embeddings based on the correlation between terms.
- **Swrank** (Tang et al., 2014): a supervised neural network based sentiment rich word embedding model.
- **Maas** (Maas et al., 2011)**:** a probabilistic document and word embedding model to capture semantic and sentiment of words.

Our proposed method is named **Tens-Embedding+"soft"+"number of clusters"** and **Tens-Embedding+"kmeans"+"number of clusters"** where soft and k-means refer to approaches applied for generating frontal slices using document-topic matrix.

4.3. Experimental setups

The support vector machine (SVM) model is used as the classifier for all settings. For each dataset, the topics are employed and the number of clusters is equal to the number of classes for comparing the accuracy of text classification. Nonetheless, the effects of the number of topics and the number of dimensions on the final results have also been investigated. The dimension of embeddings was selected as 300 for all embedding methods. SPLATT[6] (Smith, Park, & Karypis, 2015) was used for tensor decomposition. SPLATT is essentially a distributed version of the CP decomposition.

[1] http://qwone.com/~jason/20Newsgroups/
[2] https://www.cs.umb.edu/~smimarog/textmining/datasets/
[3] https://www.cs.umb.edu/~smimarog/textmining/datasets/
[4] http://www.cs.cornell.edu/people/pabo/movie-review-data/
[5] https://ai.stanford.edu/~amaas/data/sentiment/

[6] Parallel Sparse Tensor Decomposition

Table 4
Statistics of 20Newsgroups dataset.

Classes	Number of documents per class	Average number of words per document	Classes	Number of documents per classe	Average number of words per document
Alt.atheism	799	336.4	**Rec.sport.hockey**	999	240.7
Comp.graphics	973	249.6	**Sci-crypt**	991	317.9
Comp.os.ms-windows.misc	966	177.9	**Sci.electronics**	984	192.9
Comp.sys.ibm.pc.hardware	982	184.5	**Sci.med**	990	284.0
Comp.sys.mac.hardware	963	168.5	**Sci.space**	987	275.4
Comp.windows.x	985	274.4	**Soc.religion.christian**	966	378.6
Misc.forsale	975	120.3	**Talk.politics.guns**	909	327.9
Rec.autos	989	208	**Talk.politics.mideast**	940	488.2
Rec.motorcycles	996	187.2	**Talk.politics.misc**	775	429.6
Talk.sport.baseball	944	211.7	**Talk.religion.misc**	628	353.2

Table 5
The statistics of MR and IMDB datasets.

Dataset	Number of instances per positive class	Number of instance per negative class
MR	5331	5331
IMDB	25,000	25,000

4.4. Text classification

In this section the results of text classification for the proposed method compared with other methods are presented. The results are reported on R52, R8, and 20NG datasets.

4.4.1. Results on R52 dataset

As can be seen in the Table 6, the proposed Tens-Embedding approach (Tensor + Soft + 52 clus and Tensor + Kmeans + 52 clus) is superior to the other methods on the R52 dataset. The accuracy for the tensor model on R52 dataset is 94.5% with respect to the second best method, which is txt-gcn with an accuracy of 93.5%. This fine performance is due to the fact that the proposed approach uses both topic and tf-idf information. Document-topic is used to cluster documents based on topic information, which provides a more abstract view compared to the term-document. In the term-document matrix, all words are considered important and each document is seen from a word-level, and so it gives a specific view of document that does not provide a good generalization power for text classification task. However, when documents are shown at a topic-level, they can be generalized well in tasks including text-classification. These results indicate that the combination of two views leads to a rather better performance for text classification.

The number of parameters for our approach is substantially fewer than neural networks such as those used in the paragraph-vector approach. Therefore, the proposed approach performs better than other methods on the R52 dataset which is a medium-size dataset.

4.4.2. Results on R8 dataset

The results of text classification on R8 are shown in Table 7. The accuracy on R8 dataset for the proposed approach is 97.2% which is better than other approaches and it is close to text-Gcn. The proposed approach is much better than TWE and topic modeling approaches like LTTR and LDA. The accuracy of Paragraph-Vector on R8 is 85% which is almost 10% worse than proposed approach. The reason behind this poor performance is that the R8 is a small-size dataset, so does not provide enough training samples to achieve a good performance using Paragraph-Vector model. Also the Paragraph-Vector model only considers the local context of words to generate document embeddings. This model neglects the global context of a document whereas the global context contains topical and structural information of a document.

Table 6
Results of text classification on R52 dataset.

Methods	Test-Accuracy (%)
tf-idf	92.5
paragraph-vector (CBOW)	78.00
Text-Gcn	93.5
Bag of concepts	86.00
CNN-rand	85.00
CNN-Non-static	87.00
Bi-LSTM	90.00
LSTM	90.00
Tens-Embedding + Soft + 52 clus	**94.5**
Tens-Embedding + Kmeans + 52 clus	94.03

4.4.3. Results on 20-NG datasets

The results of text classification on 20-NG are shown in Table 8. The results on 20-NG dataset using our proposed method are far more superior to the results obtained for other methods. Results reported for the k-means approach correspond to an accuracy of 76.74%, which is 2.74% higher than the second best method. The results on 20NG are compared with the TWE, which uses both topic and co-occurrence information for generating word embeddings. The accuracy for TWE is 68.87%, a significant 8% less than that of the proposed method. The proposed approach has better accuracy than other topic modeling approaches including LDA, Gaussian LDA, LSI and LTTR. This is most likely due to tensor factorization and its power to extract the hidden aspects of data.

4.4.4. Results on the effect of number of topics

In this section, the effect of number of topics on the text classification results is investigated for three datasets. The number of topics in each table is selected based on dataset. These results are shown in Table 9, Table 10 and Table 11 for R52, R8 and 20NG datasets respectively. In all datasets the results show no significant change as the

Table 7
Results of text classification on R8 dataset.

Methods	Test-Accuracy (%)
tf-idf	92.5
paragraph-vector (CBOW)	85.87
Text-Gcn	97.07
TWE	91.67
CNN-rand	94.00
CNN-Non-static	95.7
Fast text	96.0
Bi-LSTM	90.00
LSTM	90.00
LDA	93.65
Gaussian LDA	93.78
LTTR	93.55
Tens-Embedding + Soft + 8 clus	**97.2**
Tens-Embedding + Kmeans + 8 clus	97.07

Table 8
Results of text classification on 20NG dataset.

Methods	Test-Accuracy (%)
tf-idf	70.0
paragraph-vector (CBOW)	71.01
TWE	68.87
W2vec averaging	70.56
LDA	69.85
Gaussian LDA	72.43
LTTR	73.22
Bag of Concepts	53.02
LSI	74.0
Tens-Embedding + Soft + 20 clus	75.75
Tens-Embedding + Kmeans + 20 clus	**76.74**

Table 9
Results for different number of topics on R52 dataset.

#Topics	Methods	Accuracy	F1-score
20	K-means	94.00	93.00
	Soft-clustering	93.96	93.00
52	k-means	94.03	94.00
	Soft-clustering	94.51	94.00
80	k-means	93.2	93.00
	Soft-clustering	94.00	93.00

Table 10
Results for different number of topics on R8 dataset.

#Topics	Methods	Accuracy	F1-score
8	K-means	96.75	97.0
	Soft-clustering	96.8	97.0
16	k-means	97.16	97.0
	Soft-clustering	96.98	97.0
24	k-means	96.5	97.0
	Soft-clustering	97.35	97.0

Table 11
Results for different number of topics on 20NG dataset.

#Topics	Methods	Accuracy	F1-score
20	K-means	76.3	76.0
	Soft-clustering	75.75	76.0
50	k-means	77.12	77.0
	Soft-clustering	75.6	75.00
80	k-means	76.78	77.0
	Soft-clustering	75.56	75.0

number of topics is altered. In R8 and 20NG the results for 16 and 50 topics were better than the other number of topics, respectively. No change was observed for results on the R52 dataset.

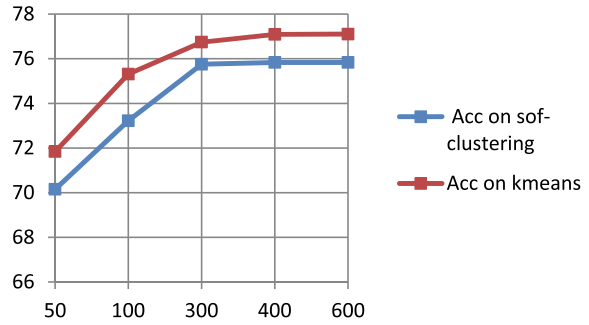

Fig. 5. The accuracy vs. number of dimensions on 20NG. The Acc in the figure is accuracy.

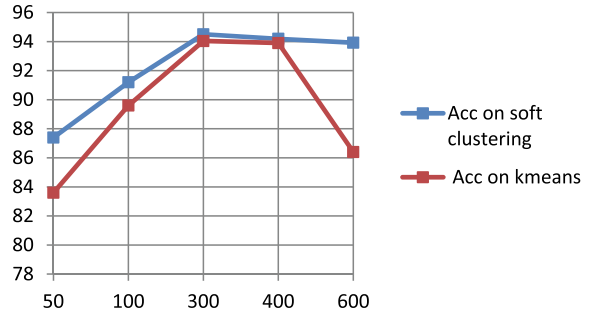

Fig. 6. The accuracy vs. number of dimensions on R52 dataset. The Acc in the figure is accuracy.

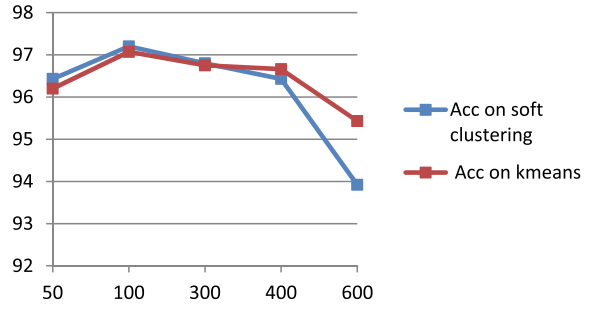

Fig. 7. the accuracy vs. number of dimensions on R8 dataset. The Acc in the figure is accuracy.

4.4.5. Number of word embedding dimension

In this section, the effect of dimension of word embeddings is investigated and the results are shown in Figs. 5, 6 and 7. According to these figures, there is a direct relationship between the two factors, i.e. the higher is the number of dimensions, the more parameters are needed in the model. Accordingly, the accuracy improves as can be expected on the 20NG dataset which contains more documents than two other datasets, and then remains almost constant after 400 dimensions. In R52 the accuracy escalates until reaching a dimension of 400 but drops at a dimension of 600 to accuracy worse than the 300 and 400 dimensions. The accuracy on R8 steadily increases from 50 to 100 dimensions followed by a plummet in accuracy after 100 dimensions. The reason lies behind the size of R8.

4.5. Sentiment analysis

To show the scalability of the model, the sentence-level and document-level sentiment analysis using the proposed method

Table 12
Results of document-level sentiment analysis on IMDB corpus.

Method	Accuracy (%)
tf-idf	87.9
CNN-pad	89.6
Word2vec	85.02
SIF	86.1
Content_tree(Glove)	86.8
LDA	67.4
LSA	83.8
paragraph-vector (CBOW)	88.2
Maas	88.3
SWrank	88.55
Tensembedtfidf + soft + 50clus	**90**
Tensembedtfidf + kmeans + 50clus	**90.27**

Table 13
The results on MR corpus.

Method	Accuracy (%)
tf-idf	74.6
CNN-rand	75
CNN-Non static	77.7
Fasttext	75.14
LSTM	75.1
Text-gcn	76.7
paragraph-vector (CBOW)	61.1
TWE	59
Tensembedtfidf + soft + 4clus	77
Tensembedtfidf + kmeans + 4clus	76.68

are evaluated in comparison with the other methods and the results are presented in this section. The experiments were conducted on MR, and IMDB datasets. The embedding dimension is set to 300.

4.5.1. Document-level sentiment analysis

The results of document-level sentiment analysis on large movie review dataset (IMDB) are presented in Table 12. According to the results, the accuracy of proposed method is higher than other methods. Maas (Maas et al., 2011) and SWrank (Tang et al., 2014) methods are two models proposed to learn embeddings for sentiment analysis. Maas model has two parts which one of them assumes a probabilistic model for document to catch the semantic similarities and the other part tries to put the words with similar sentiment near each other in vector space using a supervised model. The first part of this model is unsupervised and the second part is supervised. The objective function of the model is the summation of the objective functions of these two parts. This model uses tf-idf weighting sum of word vectors to represent documents which cannot show the whole meaning of the document. SWrank is a supervised word embedding approach which utilizes document polarities to extract sentiment rich word embedding and uses average of word embeddings to represent the documents. LSA and LDA only use topics as features to represent a document so misses the specific information provided by words which is valuable to represent a document and Doc2vec only consider the local word co-occurrences which cannot represent the valuable topical and global information. CNN-pad (Liu et al., 2017) is an unsupervised document embedding model which employs convolutional neural network for document representation. This approach is a kind of language modeling which employs a convolutional neural network and receives words of a document as inputs of the neural network and thus does not consider the topical information.

4.5.2. Results for Sentence-level sentiment analysis

The results of sentiment analysis on MR corpus are presented in Table 13. According to this table, the Tens-Embedding has better results than Doc2vec, TWE, Fasttext which are unsupervised document embedding methods but in the case of supervised methods Tens-embedding has comparable performance because the documents of MR dataset have short lengths and the LDA has not a perfect performance on short documents. The CNN-non static is proposed specifically for sentence-level sentiment analysis so has higher accuracy with respect to our model but it is still comparable.

4.6. Text clustering

Text clustering is a task to group unlabeled documents into multiple clusters such that similar documents are placed in the same cluster. Document clustering is a crucial tool to analyze a huge number of unsupervised data available in recent years. In this section, text clustering is used to examine the document vectors generated by our approach in comparison to the other methods on news data. Results of document clustering are presented on R8 and R52 datasets and are shown on Figs. 8 and 9. The measures used for evaluation are homogeneity and completeness of clusters (Becker, 2011). Homogeneity means all members of a cluster are from the same class. Completeness means that all members of a class are allocated in the same cluster. The values of these measures are bound to happen in the interval of [0, 1] where one is perfect score. The homogeneity and completeness are obtained using Eqs. (16) and (17) respectively where $H(c|k)$ is the cross entropy of categories given the cluster allocations and $H(c)$ is classes entropy and $H(k)$ is clusters entropy.

$$h = \frac{H(c|k)}{H(c)} \tag{16}$$

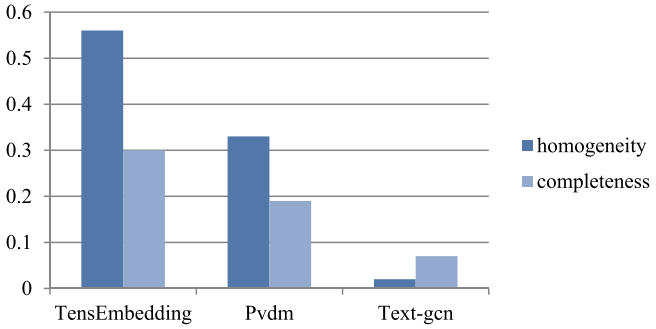

Fig. 8. The results of text clustering o R52 dataset. TensEmbedding is proposed method and pvdm is paragraph-vector approach.

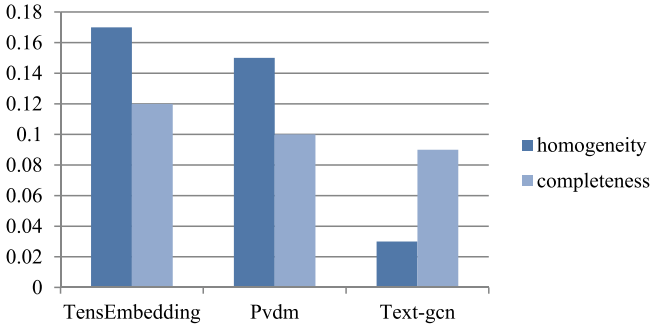

Fig. 9. The results of text clustering on R8 dataset. TensEmbedding is proposed method and pvdm is paragraph-vector approach.

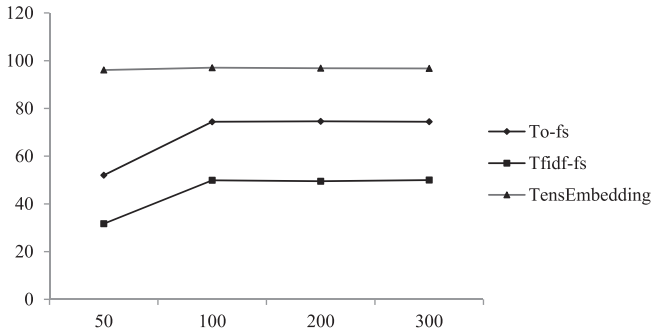

Fig. 10. Ablation study on different components of the model on R8 dataset. To-fs shows the accuracy achieved using tensor frontal slice which was produced with the help of topic-document matrix and tf-idf-fs shows the accuracy achieved using tensor frontal slice which was generated by employing term-document matrix.

$$c = \frac{H(c|k)}{H(k)} \tag{17}$$

The spectral clustering is applied on data to group document vectors into clusters. As it is shown in Figs. 8 and 9, the homogeneity and completeness of clusters extracted by employing document vectors generated by the proposed method are higher than the other two methods. As mentioned before, the paragraph vector is an unsupervised neural network based method for document representation which only uses local word co-occurrence information and text-gcn is a supervised graph-based method. The performance of text-gcn is worse than the other two methods on text clustering.

4.7. Ablation study

We run singular value decomposition (SVD) (The CP decomposition is the extension of matrix rank decomposition such as SVD to tensors) on each frontal slice of tensor to investigate the impact of different components of the model and also the tensor factorization. The Fig. 10 shows the accuracy of text classification on R8 dataset (as we observed it is almost the same for other datasets) for various ranks (50–300). As it is shown in this figure, the accuracy of proposed model which employs CP tensor decomposition is much higher than SVD while applying on each frontal slice. With CP decomposition, we can utilize the benefits of both term-document and document-topic information. As it is mentioned in the introduction section, The document-topic representation which is generated by LDA is an abstract view of the document and increases the generalization power of document representation and text classification (Chemudugunta et al., 2007), but ignores certain valuable features (words) of the document. On the contrary, in the term-document matrix, each word of the document is important therefore it is a very specific view of the document. Nonetheless, this level of specificity imperils the generalization ability of natural language processing tasks. Moreover, in a term-document matrix only first order co-occurrences of words are considered, whereas LDA can exploit the higher order co-occurrences as well.

The results in Fig. 10 and Fig. 5, Figs. 6 and 7 show robustness with respect to the embedding dimensionality. As it is shown in Fig. 10, the accuracy of To-fs and Tfidf-fs steadily increase from 50 to 100 dimensions and the changes are very small from 100 to 300. For TensEmbedding (the curve with triangle), the accuracy increases from 50 to 100 dimensions but the changes are insignificant from 100 to 300.

4.8. Discussion

According to Tables 6, 7 and 8, the accuracy of tensor approaches are higher than the baseline methods. Each one of these baseline methods represents one of the various groups mentioned in Table 1. TWE is a method which uses topics in addition to local co-occurrence of words to extract word vectors. This method uses average of word vectors for document representation. TWE has 10% lower accuracy than our approach on 20NG dataset and almost 6% lower accuracy on R8. This result reveals that our method finds better features for representing a document. Our method and TWE have some differences: TWE uses weighted sum of corresponding topic vectors of each word to find a representation for a document but Tens-embedding employs a tensor for this. TWE only considers the topic variation of words nevertheless does not take into account the topic variation of the documents. TWE uses tf-idf of words in a document as a weight for generating document embedding but does not consider the similarity of documents based on these tf-idfs. Our Tens-Embedding method considers the similarity of documents based on the tf-idf of words through multiplying the document-term by term-document matrix. Each entry of new document-document matrix shows the weighted number of words that corresponding documents share.

Another approach that utilizes topical information for generating document embedding is LTTR. The results show that the Tens-Embedding has higher accuracy than this method. LTTR assumes that each topic is a Gaussian mixture of words and the probability of a word in a document is achieved by the weighted sum of Gaussian components (topics) of that word. Each weight shows the portion of each topic in that document. The accuracy of LTTR is higher than TWE because this method considers topic variation of documents. However, Using the Gaussian distribution as the probability distribution for topics and mixture of Gaussian for documents is simple but is not a true assumption necessarily and can affect the accuracy of the model. On the other hand, LTTR does not benefit from the specific view of the documents as provided by tf-idfs.

The accuracies of Paragraph-vector method which only uses local co-occurrences are lower than the accuracies of our method on three datasets. It shows the superiority of Tens-embedding method which uses document-level co-occurrences and topical information. The other method is Text-Gcn which is a supervised graph-based approach; Tens-embedding has higher accuracy than Text-Gcn while being simpler. The Nodes of the graph in Text-Gcn are words and documents and the edges are between two words or between a word and a document. If two words are co-occurred in the corpus then they have to be connected by an edge in the graph and if a word appears in a document then an edge is created between them in the graph. The weights of the edges between two words are the point-wise mutual information (PMI) between them and the weight between a document and a word corresponds to the tf-idf value of that word in that document. Tens-Embedding considers the relation between documents in term-level and in topic-level but Text-Gcn only considers the relation between documents in term-level. Using the topical information in the representation can improve the generalization power of a classifier (Chemudugunta et al., 2007) therefore the accuracy of text classification are higher for Tens-Embedding.

The approaches such as LSI and Gaussian LDA which only use topical information miss the rare words of a document. The rare words can be utilized as distinguishing features of that document. CNN, LSTM and bi-LSTM are neural network based methods which consider the order of words in a text but the global co-occurrence information and the topical information are not considered in these methods. Therefore Tens-embedding method has better accuracy than these approaches. The word vector averaging uses the average of w2vec pre-trained word vectors for document representation. But as mentioned in introduction only an average vector does not contain enough information for representing a document so it has lower accuracy than Tens-embedding and some other approaches such as LTTR, Gaussian LDA, LDA, LSI and paragraph vector.

5. Conclusion and future works

In this paper we proposed Tens-embedding, a tensor-based document embedding method to incorporate both document-topic probabilities and term-document frequencies to generate a representation for documents. The document-topic is an aspect of documents which contains topic information. Representing documents only by information obtained from document-topic matrix can provide an overall suitable generalization for tasks such as document classification and information retrieval but fails to account for certain important features i.e. words. In the term-document matrix all words are taken into account, so the generalization of this kind of representation is not well. We utilize the strength of these two views with the help of a tensor to generate more discriminative document embeddings for text classification task. Experiments are performed on three natural language processing tasks and on five benchmark datasets and compared with various methods which use only one aspect of documents and some methods which consider topics as well. The results show outstanding performance and far better results of our proposed method with respect to other methods. Therefore the Tens-embedding method can find better features and embeddings with respect to baseline methods.

The complexity of Tens-Embedding is dependent on the number of non-zero elements of the tensor (O(nnz(x)log(nnz(x)) where nnz is the number of non-zero elements in tensor x). When the number of documents becomes large the tensor factorization becomes more time consuming but it is tractable for any number of documents. The other limitation is that the number of clusters for k-means and soft-clustering is important and can change the results and is dependent on the dataset but there is similar limitations for neural network approaches (number of hidden layers or number of neurons in each layer).

However tens-embedding has some weaknesses. First, this model does not consider the order of words in documents which are useful and important information and can change the meaning of a sentence. To resolve this issue, someone can benefit from the ability of LSTM to take the word order into account. Another weakness of the model is the sparsity of tensor since factorization of a sparse tensor cannot find the best hidden aspects of data, so we aim to solve this problem using some side information such as synonymy of words. The construction of tensor can be changed also and can be built at once instead of constructing tensor by frontal slices. In Tens-Embedding, two frontal slices have the same weights in tensor factorization so we can find a suitable weighting scheme which defines the significance of each frontal slice. This weighting can be performed on the objective function of tensor factorization or can be performed on data before factorization.

CRediT authorship contribution statement

Zahra Rahimi: Conceptualization, Methodology, Validation, Formal analysis, Investigation, Writing - original draft. **Mohammad Mehdi Homayounpour:** Resources, Validation, Supervision, Writing - review & editing.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The authors wish to express their thanks for the financial support of Iran National Science foundation (INSF), Project No 97009308.

References

Becker, H. (2011). Identification and Characterization of Events in Social Media. 193. Retrieved from http://www.cs.columbia.edu/~hila/hila-thesis-distributed.pdf

Bijari, K., Zare, H., Kebriaei, E., & Veisi, H. (2020). Leveraging deep graph-based text representation for sentiment polarity applications. *Expert Systems with Applications, 144*, 113090. https://doi.org/10.1016/j.eswa.2019.113090.

Bing, L., Jiang, S., Lam, W., Zhang, Y., & Jameel, S. (2015). Adaptive concept resolution for document representation and its applications in text mining. *Knowledge-Based Systems, 74*, 1–13. https://doi.org/10.1016/j.knosys.2014.10.003.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet Allocation. *Journal of Machine Learning Research, 3*, 993–1022. https://doi.org/10.1162/jmlr.2003.3.4-5.993.

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2016). Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics, 5*, 135–146. https://doi.org/10.1162/tacl_a_00051.

Bollegala, D., Mu, T., & Goulermas, J. Y. (2016). Cross-Domain Sentiment Classification Using Sentiment Sensitive Embeddings. *IEEE Transactions on Knowledge and Data Engineering, 28*(2), 398–410. https://doi.org/10.1109/TKDE.2015.2475761.

Camacho-Collados, J., Pilehvar, M. T., & Navigli, R. (2015). NASARI: A novel approach to a semantically-aware representation of items. In *NAACL HLT 2015–2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference* (pp. 567–577). https://doi.org/10.3115/v1/n15-1059.

Chemudugunta, C., Smyth, P., & Steyvers, M. (2007). Modeling General and Specific Aspects of Documents with a Probabilistic Topic Model. *Advances in Neural Information Processing Systems, 241–248.* https://doi.org/10.1016/j.juro.2016.02.196.

Chen, M. (2017). Efficient Vector Representation for Documents through Corruption Retrieved from. *ICLR, 2017*, 1–13 http://arxiv.org/abs/1707.02377.

Das, R., Zaheer, M., & Dyer, C. (2015). Gaussian LDA for Topic Models with Word Embeddings. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguisticsand the 7th International Joint Conference on Natural Language Processing* (pp. 795–804).

Deerwester, S., Dumias, S. T., W.Furmas, G., K.Lander, T., & Harshman, R. (1990). Indexing by Latent Semantic Analysis. Jornal of the American Society for Information Science, 41(6), 391–407. Retrieved from https://asistdl.onlinelibrary.wiley.com/doi/abs/10.1002/(SICI)1097-4571(199009)41:6%3C391::AID-ASI1%3E3.0.CO;2-9

Dourado, Í. C., Galante, R., Gonçalves, M. A., & da Silva Torres, R. (2019). Bag of textual graphs (BoTG): A general graph-based text representation model. Journal of the Association for Information Science and Technology, (April). https://doi.org/10.1002/asi.24167

Egozcue, J. J., Pawlowsky-Glahn, V., Mateu-Figueras, G., & Barceló-Vidal, C. (2003). Isometric logratio for compositional data analysis. *Mathematical Geology, 35*(3), 279–300. https://doi.org/10.1023/A:1023818214614.

Flati, T., & Navigli, R. (2014). Three birds (in the LLOD cloud) with one stone: BabelNet, Babelfy and the Wikipedia Bitaxonomy. *Proceedings of SEMANTiCS*.

Fu, M., Qu, H., Huang, L., & Lu, L. (2018). Bag of meta-words: A novel method to represent document for the sentiment classification. *Expert Systems with Applications, 113*, 33–43. https://doi.org/10.1016/j.eswa.2018.06.052.

Garcia, M. A. M., Rodriguez, R. P., Ferro, M. V., & Rifon, L. A. (2016). Wikipedia-Based Hybrid Document Representation for Textual News Classification. 2016 3rd International Conference on Soft Computing & Machine Intelligence (ISCMI), (November), 148–153. https://doi.org/10.1109/ISCMI.2016.31

Harris, Z. S. (1954). *Distributional Structure. WORD, 10*(2–3), 146–162. https://doi.org/10.1080/00437956.1954.11659520.

Hofmann, T. (1999). probabilistic latent semantic analysis. *Hofmann, Thomas. "Probabilistic Latent Semantic Analysis"*. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence* (pp. 289–296). https://doi.org/10.1007/978-81-322-2734-2_16.

Jiang, B., Li, Z., Chen, H., Member, S., & Cohn, A. G. (2018). Latent Topic Text Representation Learning on Statistical Manifolds. *IEEE Transactions on Neural Networks and Learning Systems, 29*(11), 5643–5654.

Kamkarhaghighi, M., & Makrehchi, M. (2017). Content Tree Word Embedding for document representation. *Expert Systems with Applications, 90*, 241–249. https://doi.org/10.1016/j.eswa.2017.08.021.

Kim, H. K., Kim, H., & Cho, S. (2017). Bag-of-concepts: Comprehending document representation through clustering words in distributed representation. *Neurocomputing, 266*, 336–352. https://doi.org/10.1016/j.neucom.2017.05.046.

Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification Retrieved from http://arxiv.org/abs/1408.5882.

Kolda, T. G., & Bader, B. W. (2009). Tensor Decompositions and Applications. *SIAM Review, 51*(3), 455–500. https://doi.org/10.1137/07070111x.

Lai, S., Xu, L., Liu, K., & Zhao, J. (2015). Recurrent Convolutional Neural Networks for Text Classification. Aaai'15, 2267–2273. https://doi.org/10.1145/2808719.2808746

Lakshmi, R., & Baskar, S. (2019). Novel term weighting schemes for document representation based on ranking of terms and Fuzzy logic with semantic relationship of terms. *Expert Systems with Applications, 137*, 493–503. https://doi.org/10.1016/j.eswa.2019.07.022.

Le, Q., & Mikolov, T. (2014). Distributed representations of sentences and documents. *International Conference on Machine Learning, 32*, 1188–1196. https://doi.org/10.1145/2740908.2742760.

Li, Pengfei, Mao, Kezhi, Xu, Yuecong, Li, Qi, & Zhang, Jiaheng (2020). Bag-of-Concepts representation for document classification based on automatic knowledge acquisition from probabilistic knowledge base. *Knowledge-Based Systems, 193*, 105436. https://doi.org/10.1016/j.knosys.2019.105436.

Liu, C., Zhao, S., & Volkovs, M. (2017). Unsupervised Document Embedding With CNNs. Retrieved from http://arxiv.org/abs/1711.04168

Liu, P., Qiu, X., & Huang, X. (2016). Recurrent Neural Network for Text Classification with Multi-Task Learning. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligen* (pp. 2873–2879).

Liu, Y., Liu, Z., Chua, T., & Sun, M. (2015). Topical Word Embedding. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence Topical* (pp. 2418–2424).

Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., & Potts, C. (2011). Learning word vectors for sentiment analysis. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL-2011), (February), 142–150. Retrieved from https://www.aclweb.org/anthology/P11-1015

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems, 3111–3119*. https://doi.org/10.1162/jmlr.2003.3.4-5.951.

Kipf, N., & T., & Welling, M. (2017). Semi-Supervised classification with Graph Convolusional Networks. *Iclr, 1–11*. https://doi.org/10.1051/0004-6361/201527329.

Pang, B., & Lee, L. (2005). Seeing Stars: Exploiting class relationships for sentiment categorization with respect to rating scales. *Proceedings of the ACL, 2005*, 115–124. https://doi.org/10.3115/1219840.1219855.

Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 1532–1543). https://doi.org/10.1093/nq/s5-IV.96.346-c.

Rao, G., Huang, W., Feng, Z., & Cong, Q. (2018). LSTM with sentence representations for document-level sentiment classification. *Neurocomputing, 308*(May), 49–57. https://doi.org/10.1016/j.neucom.2018.04.045.

Salton, G., & McGill, M. J. (1987). Introduction to Modern Information Retrieval.

Sen, P., Ganguly, D., & Jones, G. (2019). Word-Node2Vec: Improving Word Embedding with Document-Level Non-Local Word Co-occurrences. Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics Human Language Technologies, Volume 1 (Long and Short Papers), 1041–1051. Retrieved from https://www.aclweb.org/anthology/N19-1109

Sinoara, R. A., Camacho-Collados, J., Rossi, R. G., Navigli, R., & Rezende, S. O. (2019). Knowledge-enhanced document embeddings for text classification. *Knowledge-Based Systems, 163*, 955–971. https://doi.org/10.1016/j.knosys.2018.10.026.

Smith, S., Park, J., & Karypis, G. (2015). SPLATT : Efficient and Parallel Sparse Tensor-Matrix Multiplication Sparse Tensor Factorization on Many-Core Processors with High-Bandwidth Memory. (May). https://doi.org/10.1109/IPDPS.2015.27

Socher, R., Perelygin, A., Wu, Y., & J., Chuang, J., Christopher D. Manning, A. Y. N., & Potts, C. (2013). Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. *PLoS ONE, 8*(9), 1631–1642. https://doi.org/10.1371/journal.pone.0073791.

Tang, D., Wei, F., Yang, N., Zhou, M., Liu, T., & Qin, B. (2014). Learning Sentiment-Specific Word Embedding for Twitter Sentiment Classification. Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Long Papers), 1, 1555–1565. https://doi.org/10.3115/v1/P14-1146

Wei, W., Guo, C., Chen, J., Tang, L., & Sun, L. (2019). CCODM: Conditional co-occurrence degree matrix document representation method. *Soft Computing, 23*(4), 1239–1255. https://doi.org/10.1007/s00500-017-2844-8.

Wu, C., Wu, F., Qi, T., Cui, X., & Huang, Y. (2020). Attentive Pooling with Learnable Norms for Text Representation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (pp. 2961–2970).

Yao, L., Mao, C., & Luo, Y. (2018). graph convolutional networks for text classification.

Zhao, R., & Mao, K. (2018). Fuzzy Bag-of-Words Model for Document Representation. *IEEE Transactions on Fuzzy Systems, 26*(2), 794–804. https://doi.org/10.1109/TFUZZ.2017.2690222.