Knowledge-Based Systems 102 (2016) 76–86

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Improving short text classification by learning vector representations of both words and hidden topics

Heng Zhang[a,*], Guoqiang Zhong[b]

[a] Interactive Digital Media Technology Research Center, Institute of Automation, Chinese Academy of Sciences, 95 Zhongguancun East Road, Beijing 100190, P.R. China
[b] Department of Computer Science and Technology, Ocean University of China, 238 Songling Road, Qingdao 266100, P.R. China

ARTICLE INFO

Article history:
Received 13 September 2015
Revised 25 March 2016
Accepted 27 March 2016
Available online 30 March 2016

Keywords:
Short texts
Topic model
Data enrich
Word and topic vectors

ABSTRACT

This paper presents a general framework for short text classification by learning vector representations of both words and hidden topics together. We refer to a large-scale external data collection named ”corpus” which is topic consistent with short texts to be classified and then use the corpus to build topic model with Latent Dirichlet Allocation (LDA). For all the texts of the corpus and short texts, topics of words are viewed as new words and integrated into texts for data enriching. On the enriched corpus, we can learn vector representations of both words and topics. In this way, feature representations of short texts can be performed based on vectors of both words and topics for training and classification. On an open short text classification data set, learning vectors of both words and topics can significantly help reduce the classification error comparing with learning only word vectors. We also compared the proposed classification method with various baselines and experimental results justified the effectiveness of our word/topic vector representations.

1. Introduction

The popular use of the Internet demands the technology for short text classification [1–3] to deal with the daily/history big data such as search snippets, communication messages, product titles and so on. The bag-of-words (BOW) feature representation has achieved satisfactory results for the analysis of normal text/document based on machine learning methods [4,5] such as SVM, kNN, maximum entropy and so on. But the BOW feature has very little sense about semantics of words and so fails to achieve desired classification accuracy on short texts [6] which do not provide sufficient word co-occurrence or context shared information for effective similarity measure. Therefore, it is necessary to conduct in-depth study on feature representations for short texts.

To tackle the data sparseness problem of short texts, various methods have been proposed in literatures. Zelikovitz et al. [7] described a method for improving the classification of short text strings using a combination of labeled training data plus a secondary corpus of unlabeled but related longer documents. Hu et al. [8] enriched document representations with Wikipedia concepts and category information by mapping text documents to Wikipedia concepts, and further to Wikipedia categories. For query classification, Cao et al. [9] used neighboring queries and their corresponding clicked URLs (Web pages) in search sessions as the context information and incorporated the context information into the problem of query classification by using conditional random field (CRF) models. He et al. [10] employed a supervised-learning method to learn hint verbs, and considered URL and title information to classify snippets into three coarse categories. Dhillon et al. [11] studied a certain spherical k-means algorithm for clustering large sparse text data. Phan et al. [12] derived a set of hidden topics through topic model LDA from one large existing Web corpus for short text expansion. Vo et al. [13] also used LDA model for topic analysis but presented new methods for enhancing features by combining external texts modeled from various types of universal datasets.

Recently, word vector representations have been demonstrated to be able to produce outstanding results in some short text classification work such as sentiment analysis [14–17]. Word vectors can be learned via language modeling [18] or encoding word meaning (semantics) [14] with a probabilistic modeling approach. Based on word vectors, the text feature can be represented as the average of vector representations(*mean representation vector*) for all the words in the document [14] or learned in an unsupervised framework based on continuous distributed vector representations for the document [17]. The text feature can be used for many document analysis work such as clustering, classification and retrieval.

* Corresponding author. Tel.: +8615210237582.
E-mail addresses: heng.zhang@ia.ac.cn (H. Zhang), gqzhong@ouc.edu.cn (G. Zhong).

http://dx.doi.org/10.1016/j.knosys.2016.03.027
0950-7051/© 2016 Elsevier B.V. All rights reserved.

Besides of document analysis, word vectors can be also used in NLP applications such as named entity recognition, word sense disambiguation, parsing, tagging and machine translation.

Following the trend of short text enriching and word vector learning, we attempt to learn vector representations of both words and topics to improve short text classification. During the learning of topic model, each word from texts on the corpus is assigned with a topic [12] and these word-topic assignments (each pair of the word and its assigned topic is denoted as a word-topic assignment) at the end of topic learning are used to enrich texts on the corpus (corpus with text enriching is denoted as enriched corpus). Then both topic and word vectors can be learned on the enriched corpus by modifying traditional methods. In short text classification, short texts are enriched in a similar way as the text enriching on corpus: by doing topic inference on short texts, each word of short texts is also assigned with a topic [12] and these word-topic assignments at the end of topic inference are used to enrich short texts (short texts enriched with topics are denoted as enriched short texts). Then, features of short texts can be represented based on vectors of not only words but also topics. By exploiting benefits of topic models for text enriching and the vector based feature representation, our method can achieve the superior performance over many exiting references such as topic feature, knowledge enriching, and traditional word vector learning. The idea of learning word vectors together with topics is similar to models of topical word embeddings (TWE) [42]. TWE models view topics as pseudo words to predict contextual words, while we view topics as new words in the text and learn vectors of words and topics more interactively. Enriching short texts with topics has been successfully proposed in a different way [12] with us. In our framework, enriching texts with topics can not only overcome the data sparseness problem of short texts but also make it possible to learn vector representations of words and topics together.

The rest of this paper is organized as follows: in Section 2, we give a brief review of related works on short text classification. Section 3 gives an overview of the proposed theoretical framework. Section 4 describes the topic modeling with LDA and text enriching with topics. Section 5 gives the algorithm on learning vector representations of both words and topics. Section 6 validate the proposed system over an open data set and Section 7 offers concluding remarks and future work.

2. Related works

In this section, we briefly summarize related works on the basic text representation i.e. the BOW model, the content/feature extension for short texts, word-vector based text classification methods which are more related to ours.

2.1. BOW text representation

The use of popular text representation known as BOW can trace back to Harris's 1954 article on "Distributional structure" [23] and widely used for text classification, clustering and retrieval. In the BOW representation, it is common to weigh terms by various schemes such as TF, TF-IDF [24] and its variants [25–27]. Using these word-based feature representations in the high dimensional space, a great many text classification techniques have been proposed such as Bayesian techniques, k-nearest neighbors (kNN), the so-called Rocchio algorithm from information retrieval, artificial neural networks (ANN), support vector machines (SVM), hidden Markov models (HMMs), and decision tree (DT).

2.2. Content/feature extension for short texts

Most existing short text classification methods focus on the content/feature extension to overcome the data sparseness. One way is to expand short texts by fetching external text/knowledge. Authors in [28–30] use the short text as the query and expand the content by results returned form the search engine. Some works [31–33] exploit Wikipedia as external knowledge to enrich the short text representation with additional features. Others in [34–36] add external concepts from the knowledge base, such as Wordnet and Probase, as additional features. For feature expansion, another way is to model the latent structure of topics to connect the short text through these topics. Phan et al. [12] first build a general framework to learn classifiers with short text features combined with hidden topics. Chen et al. [38] put forward a solution by exploiting topics of multi-granularity and present a systematic way to seamlessly integrate topics and produce discriminative features for short text classification. To overcome the severe data sparsity in the short text, Cheng and Yan et al. [39] propose the biterm topic model (BTM) to learn topics by directly modeling the generation of word co-occurrence patterns (biterms) on the whole corpus.

2.3. Word vector based text representation

In the formulation of word vectors induced by language model [18–21], each word is represented by a vector which is concatenated or averaged with other word vectors in a context and the resulting vector is used to predict other words in the context. Based on word vector representations, the text level vector can be achieved in various ways. Maas et al. [14] simply used the average of all the word vectors in the document. Socher et al. [22] combined word vectors in an order given by a parse tree of a sentence, using matrix-vector operations. Le et al. [17] learned the document vector and word vectors together by concatenating the document vector with several word vectors in the document and predict the following word in the given context. All of these works learn only word vectors, while in this paper, we learn vectors of both words and topics. Compared with TWE models [42], our learning method considers more interactions between words and topics.

3. Overview of the proposed framework

In Fig. 1, we present the proposed framework consisting of three parts: topic learning, word/topic vector learning and short text classification. The topic model is estimated with LDA from the corpus, resulting in the topic model for new text inference and topic assignments of words (word-topic assignments) in each text. These word-topic assignments are used to enrich texts on the corpus for learning vectors of words and topics together. After word and topic vector learning, words and topics can be represented as vectors for text feature representation in short text classification. At the beginning of short text classification, topic inference is performed on short texts. During topic inference, each word in short texts is assigned with a topic and word-topic assignments at the end of topic inference are used to enrich each short text. Then, vectors of words and topics can be used to represent text features with more semantic information on enriched short texts. Text features are used for classifier training and new text classification. The whole framework consists of the following issues:

(1) Topic modeling on the corpus.
(2) Enriching the corpus and short texts with hidden topics.
(3) Learning vector representations of both words and topics on the enriched corpus.
(4) Feature representations of enriched short texts based on word and topic vectors.

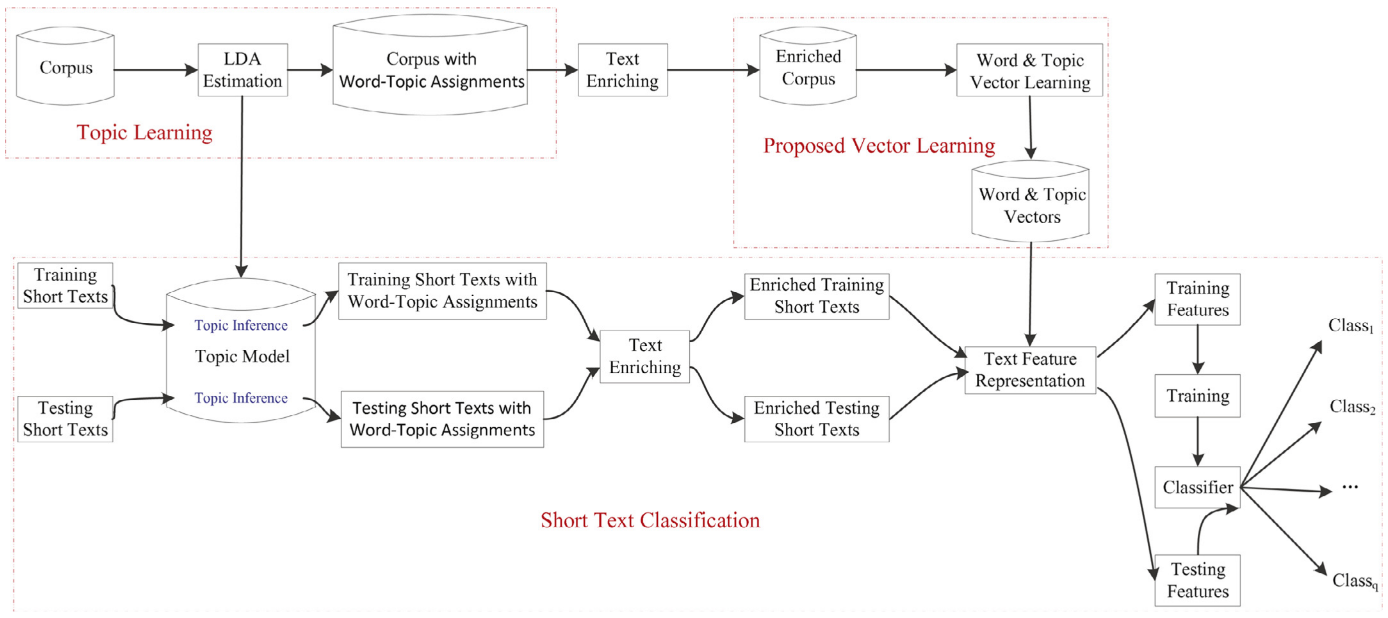


Fig. 1. Short text classification based on learning vectors of both words and topics.

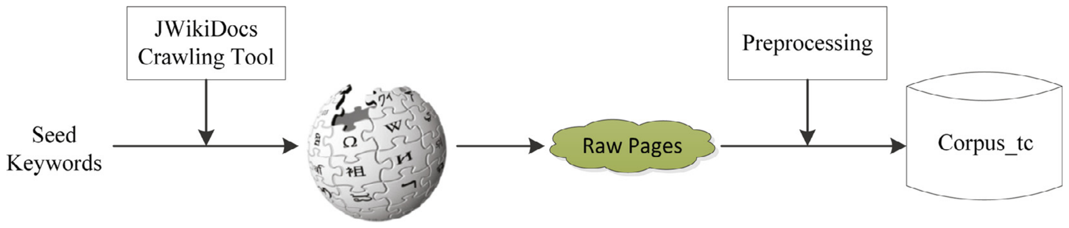


Fig. 2. Topic-consistent corpus collection.

(5) Building the classifier with text features.

The collection of topic-consistent (corpus_tc) corpus is shown in Fig. 2. Querying by each topic-oriented keyword, pages of Wikipedia are crawled using the JWikiDocs tool [40]. After preprocessing raw transactions, resulting clean texts are stored as topic-consistent corpus. For topic modeling of the corpus, we use latent Dirichlet allocation (LDA) proposed in [12]. The LDA is more complete than probabilistic latent semantic analysis (pLSA) in such a way that it follows a full generation process for document collection [37]. Each word in both the corpus and short texts is assigned with a topic which is integrated into the text for data enriching. On the enriched corpus, each topic can be viewed as a new word and so we can learn word and topic vectors together. Proposed vector learning algorithms are modified from two popular methods i.e. continuous bag-of-words (CBOW) and the skip-gram [16]. With the CBOW model, we use the sequence of contextual words and topics to predict the current word or topic. With the skip-gram model, we use the current word or topic to predict contextual words and topics. Given word and topic vectors, the feature of each short text is represented based on the mean representation vector (average of word/topic vectors in the text) which is robust for text representation. We choose one of widely used machine learning algorithms i.e. linear SVM for classifier training and text classification.

4. Topic modeling and text enriching

LDA [12,37] is a generative probabilistic model for text corpora. Topics in LDA are drawn from a conjugate Dirichlet prior that remains the same for all the documents. For each document d in a corpus D, LDA assumes the following generative process:

(1) Sample document length N_d from a Poisson distribution $Poiss(\xi)$.

(2) Pick a topic distribution $\vec{\theta}_d$ from a Dirichlet distribution $Dir(\alpha)$.

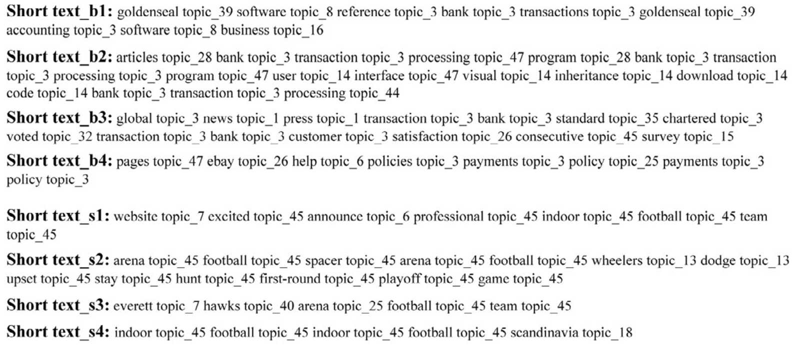
(3) For the n-*th* word in the N_d words:
   (a) Choose a topic $z_{d,\,n}$ from multinomial distribution $Mult(\vec{\theta}_d)$.
   (b) Choose a word $w_{d,\,n}$ from multinomial distribution $Mult(\vec{\varphi}_{z_{d,n}})$ with the topic-word distribution $\vec{\varphi}_{z_{d,n}}$ sampled from a Dirichlet distribution $Dir(\beta)$.

Estimating parameters of LDA is to use one approximate estimation method named Gibbs Sampling [41]. For each word w, we sample the topic assignment of this word following the multinomial distribution:

$$P(z_w = k|\vec{z}_{-w}, \vec{w}, \alpha, \beta) = \frac{n_{k,-w} + \beta}{\sum_{v=1}^{|\vec{w}|}(n_{k,v} + \beta) - 1} \frac{n_{d,-w}^{k} + \alpha}{\sum_{j=1}^{K}(n_d^j + \alpha) - 1}, \tag{1}$$

where $\vec{z}_{-w}$ denotes topic assignments of all the words except the current assignment, $n_{k,-w}$ is the number of times topic k assigned to the word w except the current assignment, $\sum_{v=1}^{|\vec{w}|} n_{k,v} - 1$ is the total number of times topic k assigned to words in vocabulary $\vec{w}$ except the current assignment, $n_{d,-w}^k$ is the number of words in document d assigned to topic k except the current assignment, $\sum_{j=1}^{K} n_d^j - 1$ is the total number of words in document d except the current word. At the last iteration of Gibbs Sampling, the topic assignment of each word is saved for corpus enriching and further topic/word vector learning (detailed in Section 5.2).

a



Short text_b1: goldenseal topic_39 software topic_8 reference topic_3 bank topic_3 transactions topic_3 goldenseal topic_39 accounting topic_3 software topic_8 business topic_16

Short text_b2: articles topic_28 bank topic_3 transaction topic_3 processing topic_47 program topic_28 bank topic_3 transaction topic_3 processing topic_3 program topic_47 user topic_14 interface topic_47 visual topic_14 inheritance topic_14 download topic_14 code topic_14 bank topic_3 transaction topic_3 processing topic_44

Short text_b3: global topic_3 news topic_1 press topic_1 transaction topic_3 bank topic_3 standard topic_35 chartered topic_3 voted topic_32 transaction topic_3 bank topic_3 customer topic_3 satisfaction topic_26 consecutive topic_45 survey topic_15

Short text_b4: pages topic_47 ebay topic_26 help topic_6 policies topic_3 payments topic_3 policy topic_25 payments topic_3 policy topic_3

Short text_s1: website topic_7 excited topic_45 announce topic_6 professional topic_45 indoor topic_45 football topic_45 team topic_45

Short text_s2: arena topic_45 football topic_45 spacer topic_45 arena topic_45 football topic_45 wheelers topic_13 dodge topic_13 upset topic_45 stay topic_45 hunt topic_45 first-round topic_45 playoff topic_45 game topic_45

Short text_s3: everett topic_7 hawks topic_40 arena topic_25 football topic_45 team topic_45

Short text_s4: indoor topic_45 football topic_45 indoor topic_45 football topic_45 scandinavia topic_18

b

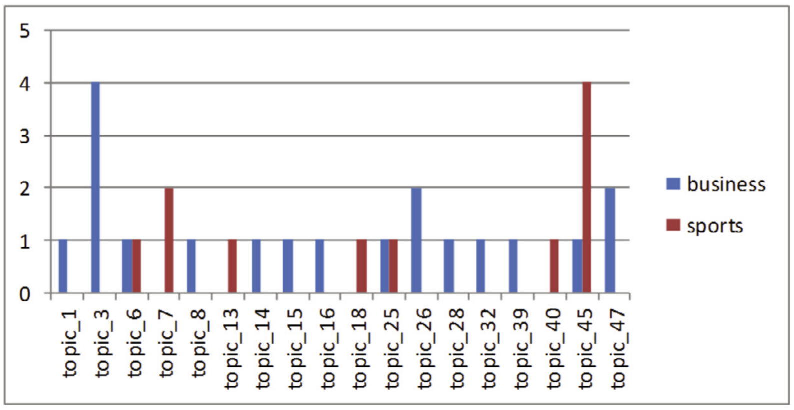


Fig. 3. (a) Enriched short texts (b* denotes business and s* sports); (b) visualization of shared topics among snippets after data enriching.

Topic inference for new documents is also performed by Gibbs Sampling. Let $\vec{w}$ and $\vec{z}$ be all the words and their topic assignments on the corpus, and $\vec{w'}$ and $\vec{z'}$ be all the words and their topic assignments of the new data. The topic assignment for the word w' depends on the current topics of all the other words in $\vec{w'}$ and topics of all the words in $\vec{w}$

$$P(z'_{w'} = k | \vec{z'}_{-w'}, \vec{w'}, \vec{z}, \vec{w}, \alpha, \beta) = \frac{n_{k,w'} + n'_{k,-w'} + \beta}{\sum_{v=1}^{|\vec{w}|}(n_{k,v} + n'_{k,v} + \beta) - 1} \frac{n'^{k}_{d_{\prime},-w'} + \alpha}{\sum_{j=1}^{K}(n'^{j}_{d_{\prime}} + \alpha) - 1}, \tag{2}$$

where $\vec{z'}_{-w'}$ denotes the topic assignments of all the words on the new data except the current assignment, $n_{k,w'}$ is the number of times topic k assigned to the word w' on the corpus in topic learning, $n'_{k,-w'}$ is the number of times topic k assigned to the word w' on the new data except the current assignment, $\sum_{v=1}^{|\vec{w}|} n'_{k,v} - 1$ is the total number of times topic k assigned to all the words on the new data except the current assignment, $n'^{k}_{d',-w'}$ is the number of words in document d' assigned to topic k except the current assignment, $\sum_{j=1}^{K} n'^{j}_{d'} - 1$ is the total number of words in document d' except the current word. At the end of Gibbs Sampling on short texts, we use word-topic assignments for further text enriching and classification.

After topic inference, each word in the short text is assigned a topic which is inserted into the text for data enriching. Fig. 3(a) shows the example of the enriched short text where the “Short text_b” denotes texts from class “business” and “Short text_s” for class “sports”. Fig. 3(b) demonstrates visualization of shared topics among the same class snippets and fewer shared topics among different classes. We can see that most topics are shared in texts from the same class and almost not appear in the other classes. Some topics are shared in all the texts of the same class and almost not in the different classes such as Topic_3 and Topic_45. After text enriching, more words are shared by texts from the same class in a semantic way. Compared to the classification method in [42] which uses topical word vectors, the enriching of short texts can make text features less sparse and more discriminative for classification by integrating topics into texts, making the data more related for the same class and less for different classes.

5. Learning vectors of both words and topics

In this section, we first give an overview of the two word vector learning methods i.e. CBOW and skip-gram. Then we show how to reply on CBOW and skip-gram for learning vectors of both words and topics. At the end, the parameter optimization process conducted by stochastic gradient descent are described in details.

5.1. Overview of word vector learning

The two well known frameworks for learning word vectors [16] are shown in Fig. 4. In the CBOW framework, both history and future words are used to predict the current word denoted as w_t. Contexts are usually fixed-length and sampled from a sliding window sized as s over the text. There are three layers in the CBOW learning framework: input, projection and output. The projection layer is a vector (denoted as a bar with shading in Fig. 4), which is an average of input word vectors to predict the output word. Given a word sequence w_1, w_2, ..., w_T, the objective of the CBOW model

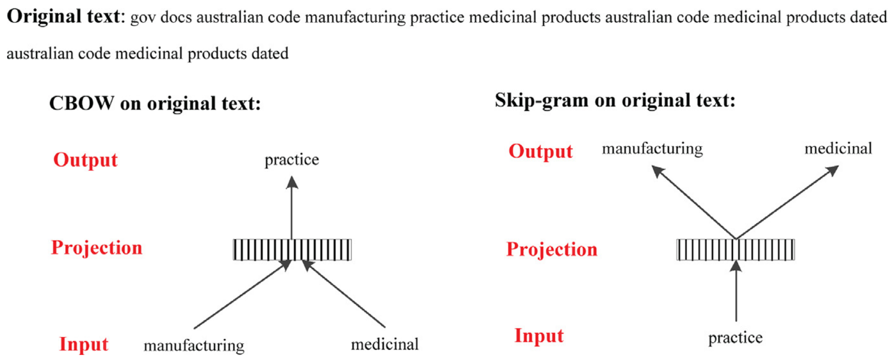

Original text: gov docs australian code manufacturing practice medicinal products australian code medicinal products dated australian code medicinal products dated
CBOW on original text:
Output
practice
Projection
Input
manufacturing
medicinal
Skip-gram on original text:
Output
manufacturing
medicinal
Projection
Input
practice

Fig. 4. The original text and word vector learning with CBOW and skip-gram.

is to maximize the average log probability

$$\frac{1}{T}\sum_{t=1}^{T} logP(w_t|w_{t-s},...,w_{t-1},w_{t+1},...,w_{t+s}), \tag{3}$$

and the prediction probability is computed by soft-max

$$P(w_t|w_{t-s},...,w_{t-1},w_{t+1},...,w_{t+s}) = \frac{e^{y_{w_t}}}{\sum_{i=1}^{W} e^{y_{w_i}}}, \tag{4}$$

where W is the word vocabulary size, y_{w_i} is the un-normalized score for output word w_i and computed as follows, with parameters b and U:

$$y = b + U \cdot h(Seq), \tag{5}$$

where Seq is the sequence of input words and the function h is constructed by an average of the input word vectors. For fast learning, the techniques of hierarchical soft-max and negative sampling are used.

In the skip-gram framework, the current word w_t is used to predict both history and future words. There are also three layers in the skip-gram framework: input, projection and output. The projection layer is the vector of the input word. The objective is to maximize the average log probability

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-s\le c\le s, c\ne 0} logP(w_{t+c}|w_t). \tag{6}$$

The probability $P(w_{t+c}|w_t)$ is also formulated using a soft-max function

$$P(w_{t+c}|w_t) = \frac{e^{V_{w_{t+c}}\cdot V_{w_t}}}{\sum_{j=1}^{W} e^{V_{w_j}\cdot V_{w_t}}}, \tag{7}$$

where V_w is the word vector of word w. The learning of skip-gram is also performed with hierarchical soft-max and negative sampling.

5.2. Our approach

After LDA based topic inference, each word in the text is assigned with a topic which is integrated following the word for data enriching. By data enriching, topics can be viewed as new words. Then traditional word vector learning algorithms can be modified with different objective functions to learn vectors of both words and topics together. Fig. 5 shows examples of the enriched text and word/topic vector learning.

Given a word/topic sequence $w_1, p_1, w_2, p_2, ..., w_T, p_T$, the objective under CBOW model is defined to maximize the following average log probability:

$$\begin{aligned}\frac{1}{T}\sum_{t=1}^{T}\{&logP(w_t|w_{t-s},p_{t-s},...,w_{t-1},p_{t-1},p_t,w_{t+1},p_{t+1},...,w_{t+s},p_{t+s})\\ &+logP(p_t|w_{t-s},p_{t-s},...,w_{t-1},p_{t-1},w_t,w_{t+1},p_{t+1},...,w_{t+s},p_{t+s})\}\\ =&\frac{1}{T}\sum_{t=1}^{T}\{logP(w_t|S_{w_t})+logP(p_t|S_{p_t})\}\\ =&\frac{1}{T}\sum_{t=1}^{T}\sum_{x_t\in\{w_t,p_t\}} logP(x_t|S_{x_t}),\end{aligned} \tag{8}$$

where S_{w_t}, S_{p_t} and S_{x_t} respectively denote the input word/topic sequence to predict w_t, p_t and x_t. Compared with Eq. (3) using context words to predict the current word, the proposed objective also use topics of the current and context words. Besides, topics are viewed as new words and so each topic is also predicted in a similar way.

The prediction probability in Eq. (8) is computed by soft-max

$$P(x_t|S_{x_t}) = \frac{e^{Y(x_t,S_{x_t})}}{\sum_{i=1}^{W} e^{Y(w_i,S_{x_t})} + \sum_{k=1}^{K} e^{Y(p_k,S_{x_t})}} \tag{9}$$

where K is the total topic number, $Y(x_o, S_{x_t})$ is the un-normalized score for output word/topic x_o with S_{x_t} as input computed as follows, with parameters b and U:

$$Y(x_o, S_{x_t}) = b_{x_o} + U_{x_o}\cdot h(S_{x_t}), \tag{10}$$

where the function h is constructed by an average of input word/topic vectors.

For learning vectors of both words and topics, the objective under skip-gram model is defined to maximize the following average log probability:

$$\begin{aligned}&\frac{1}{T}\sum_{t=1}^{T}\left\{\sum_{-s\le c\le s,c\ne 0}(logP(w_{t+c}|w_t)+logP(p_{t+c}|w_t))+logP(p_t|w_t)\right.\\ &\left.+\sum_{-s\le c\le s,c\ne 0}(logP(w_{t+c}|p_t)+logP(p_{t+c}|p_t))+logP(w_t|p_t)\right\}\\ &=\frac{1}{T}\sum_{t=1}^{T}\sum_{-s\le c\le s}\sum_{\substack{x_{in}\in\{x_t,\,p_t\},\\ x_o\in\{w_{t+c},\,p_{t+c}\},\\ x_{in}\ne x_o}} logP(x_o|x_{in}),\end{aligned} \tag{11}$$

where x_{in} and x_o are input and output words or topics. Compared with Eq. (6) using the current word to predict context words, the proposed objective also predicts topics of the current and context words. Besides, topics are viewed as new words and so each topic is also used to predict context words and topics together with the current word.

The probability in Eq. (11) is also formulated using the soft-max function:

$$P(x_o|x_{in}) = \frac{e^{V'_{x_o}\cdot V_{x_{in}}}}{\sum_{i=1}^{W} e^{V'_{w_i}\cdot V_{x_{in}}} + \sum_{j=1}^{K} e^{V'_{p_j}\cdot V_{x_{in}}}}. \tag{12}$$

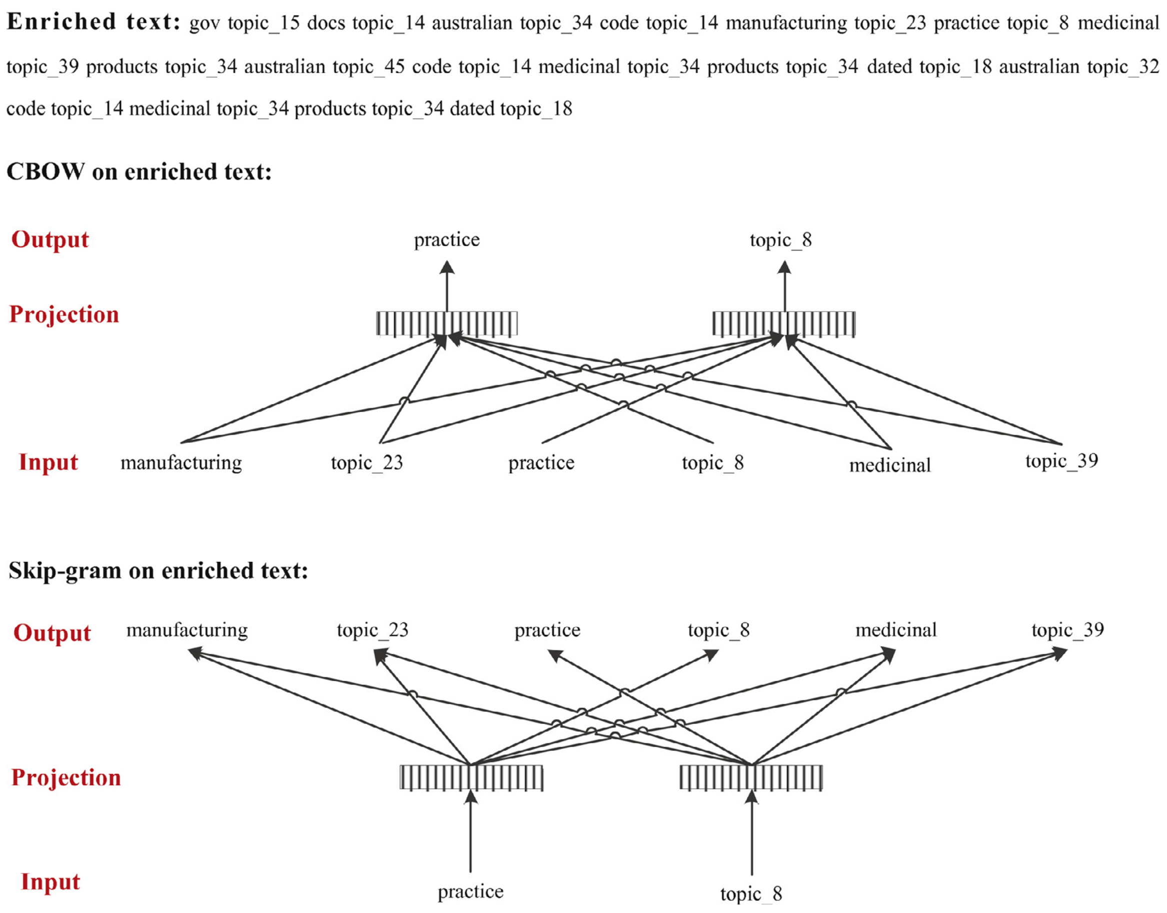

Enriched text: gov topic_15 docs topic_14 australian topic_34 code topic_14 manufacturing topic_23 practice topic_8 medicinal topic_39 products topic_34 australian topic_45 code topic_14 medicinal topic_34 products topic_34 dated topic_18 australian topic_32 code topic_14 medicinal topic_34 products topic_34 dated topic_18

Fig. 5. The enriched text and word/topic vector learning with CBOW and skip-gram.

5.3. Parameter estimation

The parameter optimization process is conducted by stochastic gradient descent which can be defined as

$$\theta \leftarrow \theta + \alpha \frac{\partial L}{\partial \theta}, \tag{13}$$

where L is the objective function to maximize and θ is the set of parameters. In the proposed CBOW model, at current word w_t and topic p_t, parameters θ_{cbow} can be updated by stochastic gradient descent as follows:

$$\theta_{cbow} \leftarrow \theta_{cbow} + \alpha \frac{\partial (L_t^{cbow})}{\partial \theta_{cbow}}, \tag{14}$$

where $L_t^{cbow} = \sum_{x_t \in \{w_t, p_t\}} logP(x_t|S_{x_t})$. In the proposed skip-gram model, at current word w_t and topic p_t, parameters θ_{skg} can be updated by stochastic gradient descent as follows:

$$\theta_{skg} \leftarrow \theta_{skg} + \alpha \frac{\partial (L_t^{skg})}{\partial \theta_{skg}}, \tag{15}$$

where

$$L_t^{skg} = \sum_{-s \le c \le s} \sum_{\substack{x_{in} \in \{x_t, p_t\}, \\ x_o \in \{w_{t+c}, p_{t+c}\}, \\ x_{in} \neq x_o}} logP(x_o|x_{in}). \tag{16}$$

From Eqs. (9) and (12), it is easy to see that derivative computation of model parameters in Eqs. (14) and (15) is proportional to the sum of topic and vocabulary size, which is often very large. To solve the computational complexity of derivatives, we use hierarchical softmax and negative sampling as efficient approximation of the full softmax for fast learning.

5.3.1. Hierarchical softmax

In this paper, the hierarchical softmax uses a binary Huffman tree representation of the output layer with the W words and K topics as its leaves and explicitly represents the relative probabilities of the child nodes for each node. On the binary tree, each word or topic x can be reached by an appropriate path from the root of the tree. Let $n(x, i)$ be the i-th node on the path with length $L(x)$ from the root to x. Then $n(x, 1)$ is the root and $n(x, L(x))$ is x. In addition, let $rch(n)$ is the right child node of node n.

For the CBOW model, the new prediction probabilities in Eq. (8) can be computed by hierarchical softmax as follows:

$$P(x_t|S_{x_t}) = \prod_{i=1}^{L(x_t)-1} \sigma([[n(x_t, i+1) = rch(n(x_t, i))]] \times (U_{n(x_t,i)} h(S_{x_t}) + b_{n(x_t,i)})), \tag{17}$$

where $\sigma(\chi) = \frac{1}{1+exp(-\chi)}$, and $[[\chi]]$ is 1 if χ is true or -1. It is easy to see that the cost of prediction probabilities and their derivatives is proportional to $L(w_t) + L(p_t)$ which is not greater than $log(W + K)$. Also, unlike the standard softmax formulation of the CBOW model where the hidden nodes connect to the words/topics, the hierarchical softmax formulation has connections from the hidden nodes to the inner nodes on the binary tree with parameters $\theta_{cbow} = \{U_n, b_n, V_x\}$. Then derivatives in Eq. (14) can be computed as

$$\frac{\partial (L_t^{cbow})}{\partial U_{n(x_t,i)}} = (1 - f_{n(x_t,i)})[[n(x_t, i+1) = rch(n(x_t, i))]] \cdot h(S_{x_t}), \tag{18}$$

$$\frac{\partial (L_t^{cbow})}{\partial b_{n(x_t,i)}} = (1 - f_{n(x_t,i)})[[n(x_t, i+1) = rch(n(x_t, i))]], \tag{19}$$

$$\frac{\partial (L_t^{cbow})}{\partial V_{x_{in}}} = \sum_{\substack{x_t \in \{w_t, p_t\},\\ x_t \neq x_{in}}} \left\{ \sum_{i=1}^{L(x_t)-1} (1 - f_{n(x_t,i)}) \times [[n(x_t, i+1) = rch(n(x_t, i))]] U_{n(x_t,i)} \right\}, \tag{20}$$

where x_{in} is any input word or topic, and $f_{n(x_t,i)} = \sigma([[n(x_t, i+1) = rch(n(x_t, i))]](U_{n(x_t,i)} h(S_{x_t}) + b_{n(x_t,i)}))$.

For the skip-gram model, the new prediction probabilities in Eq. (11) can be computed by hierarchical softmax as follows:

$$P(x_o|x_{in}) = \prod_{i=1}^{L(x_o)-1} \sigma([[n(x_o, i+1) = rch(n(x_o, i))]](V'_{n(x_o,i)} \cdot V_{x_{in}})), \tag{21}$$

it can be seen that the cost of prediction probabilities and their derivatives is proportional to $L(w_t) + L(p_t)$ which is not greater than $log(W + K)$. Besides, unlike the standard softmax formulation of the skip-gram which assigns word representations V_x to each word or topic x, the hierarchical softmax formulation also has one representation V_n for each inner node n of the binary tree resulting in $\theta_{skg} = \{V_x, V_n\}$. Then derivatives in Eq. (15) can be computed as

$$\frac{\partial L_t^{skg}}{\partial V_{n(x_o,i)}} = \sum_{\substack{x_{in} \in \{x_t, p_t\},\\ x_{in} \neq x_o}} (1 - f_{n(x_o,i)}) \times [[n(x_o, i+1) = rch(n(x_o, i))]] V_{x_{in}}, \tag{22}$$

$$\frac{\partial L_t^{skg}}{\partial V_{x_{in}}} = \sum_{-s \le c \le s} \sum_{\substack{x_o \in \{w_{t+c}, p_{t+c}\},\\ x_o \neq x_{in}}} \sum_{i=1}^{L(x_o)-1} (1 - f_{n(x_o,i)}) \times [[n(x_o, i+1) = rch(n(x_o, i))]] V_{n(x_o,i)}, \tag{23}$$

where $f_{n(x_o,i)} = \sigma([[n(x_o, i+1) = rch(n(x_o, i))]](V'_{n(x_o,i)} \cdot V_{x_{in}}))$.

5.3.2. Negative sampling

As a simple alternative to the hierarchical softmax, negative sampling is an extremely simple training method with $Neg(x)$ negative samples for each positive sample x.

For the CBOW model, the predicting probability with negative sampling is defined as

$$P(x_t|S_{x_t}) = \sigma(U_{x_t} h(S_{x_t}) + b_{x_t}) \prod_{i=1}^{Neg(x_t)} \sigma(-U_{x_i} h(S_{x_t}) - b_{x_i}), \tag{24}$$

and derivatives in Eq. (14) can be computed as

$$\frac{\partial (L_t^{cbow})}{\partial U_{x_t}} = (1 - f_{x_t}) h(S_{x_t}), \tag{25}$$

$$\frac{\partial (L_t^{cbow})}{\partial b_{x_t}} = 1 - f_{x_t}, \tag{26}$$

$$\frac{\partial (L_t^{cbow})}{\partial U_{x_i}} = (f_{x_i} - 1) h(S_{x_t}), \tag{27}$$

$$\frac{\partial (L_t^{cbow})}{\partial b_{x_i}} = f_{x_i} - 1, \tag{28}$$

$$\frac{\partial (L_t^{cbow})}{\partial V_{x_{in}}} = \sum_{\substack{x_t \in \{w_t, p_t\},\\ x_t \neq x_{in}}} \left\{ (1 - f_{x_t}) U_{x_t} + \sum_{i=1}^{Neg(x_t)} (f_{x_i} - 1) U_{x_i} \right\}, \tag{29}$$

where x_{in} is any input word or topic and $f_x = \sigma(U_x h(S_{x_t}) + b_x)$.

For the skip-gram model, the predicting probability with negative sampling is defined as

$$P(x_o|x_{in}) = \sigma(V'_{x_o} \cdot V_{x_{in}}) \prod_{j=1}^{Neg(x_o)} \sigma(-V'_{x_j} \cdot V_{x_{in}}) \tag{30}$$

and then derivatives in Eq. (15) can be computed as

$$\frac{\partial L_t^{skg}}{\partial V_{x_o}} = \sum_{\substack{x_{in} \in \{x_t, p_t\},\\ x_{in} \neq x_o}} (1 - f'_{x_o}) V_{x_{in}}, \tag{31}$$

$$\frac{\partial L_t^{skg}}{\partial V_{x_j}} = \sum_{\substack{x_{in} \in \{x_t, p_t\},\\ x_{in} \neq x_o}} (f'_{x_j} - 1) V_{x_{in}}, \tag{32}$$

$$\frac{\partial L_t^{skg}}{\partial V_{x_{in}}} = \sum_{-s \le c \le s} \sum_{\substack{x_o \in \{w_{t+c}, p_{t+c}\},\\ x_o \neq x_{in}}} \left\{ \sum_{j=1}^{Neg(x_o)} (f'_{x_j} - 1) V_{x_j} + (1 - f'_{x_o}) V_{x_o} \right\}, \tag{33}$$

where $f'_x = \sigma(V'_x \cdot V_{x_{in}})$.

5.4. Discussions with TWE

The idea of learning both words and topics under skip-gram model is similar to the topical word vectors named TWE [42] shown in Fig. 6. The TWE-1 uses both the word and its assigned topic to predict context words in skip-gram, but not considering the immediate interaction between a word and its assigned topic for learning. The TWE-2 considers the inner interaction by viewing the word/topic pair as a pseudo word and learning the pseudo word vectors (topical word embeddings) directly, but it suffers from the sparsity issue. TWE-3 builds the vector of each word-topic pair by concatenating the corresponding word and topic vectors and each word-topic pair is predicted by the contextual word-topic pairs. The TWE-3 provides trade-off between discrimination and sparsity, but topic embeddings will influence the corresponding word embeddings during the learning process and may make those words in the same topic less discriminative.

Here, we regard each topic as a new word and insert the topics into the corpus. The vectors of both words and topics are learned on the enriched texts together. Compared to TWE-1, each word is predicted by not only the contextual words together with topics but also the topic assigned to the output word in topic learning. TWE-1 only predicts words but our method regards topics as new words which are also added to the output layer for vector learning. TWE-1 learns embeddings of words and topics separately, while

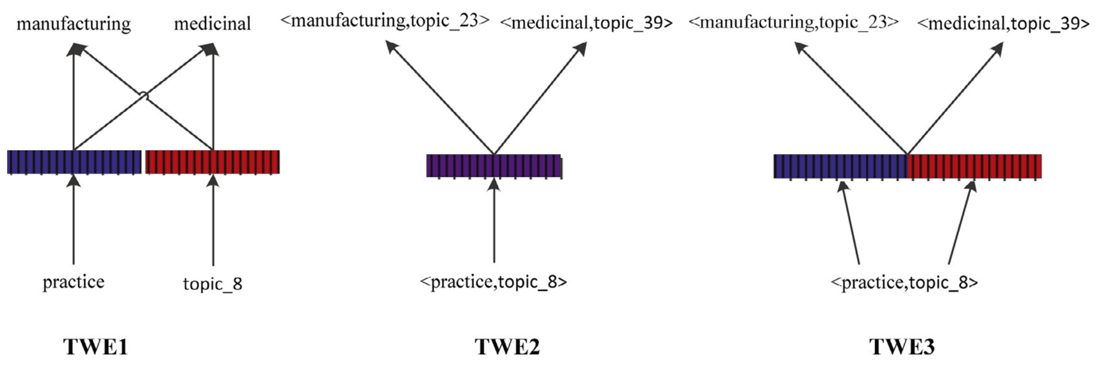


Fig. 6. Three kinds of TWE models.

Table 1
Model complexity comparison with TWE.

Model	Model Parameters	Computational Complexity
TWE1	(W + K)L	2sMlogW
TWE2	WKL	sMlog(WK)
TWE3	(W + K)L	sMlog(WK)
Ours	(W + K)L	2sMlog(W + K)

Table 2
Statistics of snippets as training and testing data.

Domain	Training data	Testing data
Business	1200	300
Computers	1200	300
Culture-Arts-Ent.	1880	330
Education-Science	2360	300
Engineering	220	150
Health	880	300
Politics-Society	1200	300
Sports	1120	300
Total	10060	2280

our method learns words and topics together. So our method considers the inner interaction between a word and its assigned topic for learning. TWE-2 divides occurrences of each word into multiple topics, the learning of embeddings may suffer from more sparsity issue. In our learning framework, parameters of each word and the assigned topic are shared over all word-topic pairs compared with TWE-2 where each word-topic pair has their own parameters. So our method alleviates the sparsity problem over TWE-2. TWE-3 concatenates embeddings of the word and the assigned topic for learning, but may make those words in the same topic less discriminative compared with our method.

In Table 1, we show the complexity of our method and the three TWE models, including the number of model parameters and computational complexity. In this table, vector lengths of words and topics are both L, orignal corpus length M and window size s. We can see that, compared with TWE models, our method requires no more parameters and the computational complexity does not increase too much.

6. Experimental results

6.1. Data sets

We evaluated the performance of the proposed short text classification method on a database of web search snippets [12] which are very short, sparse, noisy and less topic-focused. The data set consists of two subsets, named short texts and Corpus. We want to classify short texts, drawing support from the Corpus for topic modeling and word/topic vector learning. The corpus was collected by crawling pages with topic-related seed words so that it is topic-consistent with the short texts. We report macro-averaging precision, recall and F1 as the performance metrics.

6.1.1. Search snippets

The search snippets were selected from results of web search transaction using predefined phrases of different domains. For each query phrase put into Google search engine, the top 20 or 30 ranked web search snippets were collected. Then the class label of the collected search snippets was assigned as the same as that of the issued phrase. Some basic statistics of these search snippets are summarized in Table 2.

6.1.2. Corpus collection

To collect the topic-consistent corpus, some seed keywords were prepared coming from different domains. For each seed keyword, JWikiDocs tool [40] was run to download the corresponding Wikipedia page and crawl relevant pages by following outgoing hyperlinks. Each crawling transaction was limited by the total number of download pages or the maximum depth (4 in the experiment) of hyperlinks. The seed keywords and the statistics of the crawled Wikipedia data are shown as follows:

Topic-oriented keywords for crawling Wikipedia

Arts architecture, fine, art, dancing, fashion, film, music ...
Business advertising, e-commerce, finance, investment ...
Computers hardware, software, database, multimedia ...
Education course, graduate, professor, university ...
Engineering automobile, telecommunication, civil, eng. ...
Entertainment book, music, movie, painting, photos ...
Health diet, therapy, healthcare, treatment, nutrition ...
Mass-media news, newspaper, journal, television ...
Politics government, legislation, party, regime, military ...
Science biology, physics, chemistry, ecology, laboratory ...
Sports baseball, cricket, football, tennis, olympic games ...
Misc. association, development, environment ...

Statistics of the crawled Wikipedia data

Raw data: 3.5GB; |docs| = 471,177
Preprocessing: removing duplicate docs, HTML tags, navigation links, stop and rare words
Final data: 240MB; |docs| = 71,986; |paragraphs| =882,376; |vocabulary| = 60,649; |total words| = 30,492,305

6.2. Effectiveness of the topic-consistent corpus

The more consistent between the corpus and short texts to be classified, the more effective is the corpus for topic modeling and word vector learning. To verify the effectiveness of the topic-consistent corpus, we use the latest Wikipedia dump as the general corpus for comparison. For simplification, only word vectors are learned on the two corpora with skip-gram and CBOW. Word

Table 3
Effectiveness of the topic-consistent corpus

Model	gr-CBOW	tc-CBOW	gr-skip-gram	tc-skip-gram
P (%)	83.12	84.33	84.01	85.46
R (%)	82.94	84.06	84.25	85.00
F1 (%)	83.02	84.19	84.13	85.23

Table 4
Classification results changing according to the number of topics

K	30	40	50	60	70	80	90	100
P (%)	82.01	81.56	82.87	81.37	81.31	80.12	80.25	79.93
R (%)	81.79	81.27	82.68	81.42	81.06	79.66	80.08	79.85
F1 (%)	81.90	81.41	82.77	81.39	81.18	79.89	80.16	79.89

Table 5
Effectiveness of different learning methods for classification

Model	ns-CBOW	hs-CBOW	ns-skip-gram	hs-skip-gram
P (%)	85.33	86.23	86.01	87.37
R (%)	85.12	86.39	86.26	87.48
F1 (%)	85.22	86.31	86.13	87.42

vectors are learned by a particular tool named word2vec [16]. The text feature is the mean representation vector and the classifier is linear SVM. The classification results are shown in Table 3 and we can see that the topic-consistent corpus (tc-) is more effective than the general (gr-).

6.3. Topic modeling

We estimate topic models based on LDA (implemented with the package named GibbsLDA++ [12]) through Gibbs Sampling on the collected corpus. We set $\alpha = 50/K$ and $\beta = 0.01$. The number of topics ranges from 30, 40, 50...to 80, 90, and 100. To estimate parameters of each model, we run 2000 Gibbs Sampling iterations, and save the estimated model at every 200 iterations. After topic modeling, we perform topic inference with 50 Gibbs Sampling iterations on both training and testing data, using topic distributions as features and building SVM classifiers for training and classification. The results are shown in Table 4 and we can see the best results are achieved at $K = 50$. So we set topic number as 50 to learn the topic model for further experiments.

6.4. Word and topic vector learning

6.4.1. Comparison of different learning methods

With hierarchical softmax and negative sampling proposed in Section 5.3, we learn vector representations of both words and topics by CBOW and skip-gram models. Four sets of vectors are resulted in i.e. CBOW with hierarchical softmax (hs-CBOW), CBOW with negative sampling (ns-CBOW), skip-gram with hierarchical softmax (hs-skip-gram), skip-gram with negative sampling (ns-skip-gram). To verify the effectiveness of hierarchical softmax and negative sampling, we respectively use the four sets of vectors to perform short text classification. The classification results are shown in Table 5. It can be seen that the best results are achieved with vectors learned by skip-gram with hierarchical softmax, which are used to perform the resulting experiments.

6.4.2. Effectiness of topic vectors

To verify the effectiveness of learning word and topic vectors together, we perform short text classification with three sets of vectors i.e. only word vectors, only topic vectors, both word and topic vectors. The word vectors are learned on the original

Table 6
Effectiveness of representing both words and topics as vectors for classification

Vectors	only words	only topics	both words and topics
P (%)	85.46	84.33	87.37
R (%)	85.00	83.82	87.48
F1 (%)	85.23	84.07	87.42

corpus using traditional skip-gram with hierarchical softmax. The topic vectors are produced by removing the word vectors from the word/topic vectors which are learned by the proposed skip-gram with hierarchical softmax. The classification results are shown in Table 6. We can see that learning word and topics vectors together can improve the classification results compared with using word or topic only.

6.5. Comparison with baselines

We consider the following baselines which have been proposed on the snippet dataset in other literatures:

BOW

The BOW model represents each document as a bag of words and the term weighting scheme is TF-IDF. The classifier training method is the linear SVM.

LDA+MaxEnt [12]

LDA is used to discover a set of hidden topics from the corpus. These topics are used as features to enrich the representation of short text and the classifier is Maximum Entropy (MaxEnt) which is very fast in both training and inference.

Multi-Topics [38]

LDA is run on the corpus to generate topic models with respect to different topic number. Then, a subset of all the generated topic models is chosen to form the topic space of multiple granularity. The features of the multi-granularity topics are used to generate features for short text classification.

Classifier integration [43]

The short texts are enriched with LDA based topics and respectively learned with SVM and MaxEnt. The results of the two classifiers are integrated for a comprehensive prediction.

Link analysis [45]

The BOW feature is enriched with the most topic related words based on the link analysis of the topic-word graph. The enriched feature is used for classifier training and text classification with SVM.

CNN [44]

The word vectors are first clustered to discover semantic cliques. Then, multi-scale semantic units are detected under the supervision of semantic cliques, which introduce useful external knowledge for short texts. These meaningful semantic units are combined and fed into convolutional neural networks (CNN), followed by max-pooling operation.

The comparisons are shown in Table 7 and we can see the superiority of our method. The BOW feature has very little sense about the semantics of the words and so performs poorly on short texts which provide insufficient word co-occurrence. The methods of LDA+MaxEnt, Multi-topics and the classifier integration (Cls-integration in Table 7) try to discover the LDA based topics for semantic enriching and use the topic features for classification. But the emphasis in LDA is on modeling topics not word meanings with no guarantee that the representations are sensible as points in the feature space. The link analysis method tries to enrich the short text with topic related words and use BOW model for classification. But the expanded words are based on the link analysis on the word-topic graph derived from LDA still not modeling word meanings. The CNN based classification method uses only the word

Table 7
Comparison with baselines

Model	Proposed	BOW	LDA	Multi-topics	Cls-integration	Link analysis	CNN
P (%)	87.37	66.23	82.87	85.12	86.33	86.22	84.68
R (%)	87.48	66.19	82.68	84.86	86.48	86.04	84.45
F1 (%)	87.42	66.21	82.77	84.99	86.40	86.13	84.56

Table 8
Paired *t*-test results for baselines

Model	Proposed	BOW	LDA	Multi-topics	Cls-integration	Link analysis	CNN
AM	87.97	66.83	83.24	85.33	86.95	86.65	85.11
STD	0.061	0.041	0.148	0.032	0.045	0.037	0.054
TR	N	Y	Y	Y	Y	Y	Y

Table 9
Comparison with TWE models

Model	Proposed	TWE1	TWE2	TWE3
P (%)	87.37	86.22	85.88	85.46
R (%)	87.48	86.05	85.49	85.17
F1 (%)	87.42	86.13	85.68	85.31

Table 10
Paired t-test results for TWE models

Model	Proposed	TWE1	TWE2	TWE3
AM	87.97	86.78	86.24	86.01
STD	0.061	0.035	0.038	0.043
TR	N	Y	Y	Y

vetors without the embeddings of topics and the word vectors are learned on the general corpus not topic-consistent with the short texts. Our method follows the benefit of enriching texts with topics and further learn vectors of both words and topics on the topic-consistent corpus for text representation and classification.

To evaluate the significance of the performance differences between the proposed method and the baselines, we conduct the paired t-test with $\alpha = 0.05$ on the results of 5-fold cross validation. The corresponding accuracy means (AM), standard deviations (STD) and test results (TR) are shown in Table 8. The test results are denoted by Y or N. Y means the performance difference between the proposed and the compared methods is significant and N not. We can see that the proposed method performed significantly better than the compared methods.

6.6. Comparison with topical word e mbeddings

Our method is similar to the topical word embeddings [42] in learning both word and topic vectors. Because TWE models have not been used for short text classification, we compare TWE models with our method specially not together with other baselines. Table 9 shows the results of the comparison between our method and the three TWE models. The TWE-1 model achieves the best performance among the three TWE models and our method performs better than all the TWE models.

TWE-1 uses the assigned topic to predict the contextual words and so does not consider the immediate interaction between the word and its assigned topic in learning. TWE-2 regards the word-topic pair as a pseudo word and the inner interaction of the pair is considered, but it suffers from the sparsity issue because the occurrences of each word are rigidly discriminated into different topics. TWE-3 provides trade-off between discrimination and sparsity by concatenating the embeddings of the word and the assigned topic for learning, but topic embeddings will influence the corresponding word embeddings, which may make those words in the same topic less discriminative. Our method regards the topics as the new words in the text, and learns the vectors of both words and vectors together by integrating the topics with the current and the contextual words. So our learning method benefits more interactions between the words and the topics than TWE-1 while avoids the sparse problem compared with TWE-2 and does not make the words in the same topic less discriminative compared with TWE-3.

To evaluate the significance of the performance differences between the proposed method and TWE models, we conduct the paired *t*-test with $\alpha = 0.05$ on the results of 5-fold cross validation. The corresponding accuracy means (AM), standard deviations (STD) and test results (TR) are shown in Table 10. We can see that the proposed method performed significantly better than the three TWE methods.

7. Conclusions and future work

In this paper, we improve the task of short text classification by learning vector representations of not only words but also their topics. We estimate the topic model with LDA on the corpus and enrich texts with the word topics. On the enriched corpus, viewing topics as new words, vectors of both words and topics are learned together. For short text classification, we do topic inference on each text and also enrich the text with topics. Then features of short texts are represented based on vectors of both words and topics for training and classification. The experimental results demonstrate the superiority of our methods over various baselines. There are some other issues that are missing in this paper. The future work will explore more tasks in our framework such as topic learning for short texts, discriminative representations of short texts based on vectors of words and topics, collecting corpus more topic-consistent with the classified short texts.

Acknowledgment

This work has been supported by the National Natural Science Foundation of China (NSFC) under grants No. 61403385 and No. 61403353, the Open Project Program of the National Laboratory of Pattern Recognition (NLPR) and the Fundamental Research Funds for the Central Universities of China. We also give great thanks to Peng Wang for his useful discussions.

References

[1] B. Sriram, D. Fuhry, E. Demir, H. Ferhatosmanoglu, M. Demirbas, Short text classification in twitter to improve information filtering, in: Proceeding of ACM SIGIR, ACM, 2010, pp. 841–842.

[2] A. Sun, Short text classification using very few words, in: Proceeding of ACM SIGIR, ACM, 2012, pp. 1145–1146.
[3] S. Busemann, S. Schmeier, R.-G. Arens, Message classification in thecall center, in: Proceedings of the sixth conference on Applied natural language processing, Morgan Kaufmann, 2000, pp. 158–165.
[4] F. Sebastiani, Machine learning in automated text categorization, ACM Comput. Surv. 34 (1) (2002) 1–47.
[5] M. Lan, C.-L. Tan, H.-B. Low, S.-Y. Sung, A comprehensive comparative study on term weighting schemes for text categorization with support vector machines, in: Proceeding of WWW, ACM, 2005, pp. 1032–1033.
[6] H.-F. Yu, C.-H. Ho, Y.-C. Juan, C.-J. Lin, LibShortText: a library for short-text classification and analysis, Technical report 2013, LibShortText software.
[7] S. Zelikovitz, H. Hirsh, Improving short-text classification using unlabeled background knowledge to assess document similarity, in: Proceedings of the seventeenth international conference on machine learning, Morgan Kaufmann, 2000, pp. 1183–1190.
[8] X.-H. Hu, X.-D. Zhang, C.-M. Lu, E.-K. Park, X.-H. Zhou, Exploiting wikipedia as external knowledge fordocument clustering, in: Proceeding of KDD'09, ACM, 2009, pp. 389–396.
[9] H.-H. Cao, D.-H. Hu, D. Shen, D.-X. Jiang, J.-T. Sun, E.-H. Chen, Q. Yang, Context-aware queryclassification, in: Proceeding of SIGIR'09, ACM, 2009, pp. 3–10.
[10] K.-Y. He, Y.-S. Chang, W.-H. Lu, Improving identification of latent user-goals through search-result snippet classification, in: Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence, IEEE Computer Society, 2007, pp. 683–686.
[11] I. Dhillon, D. Modha, Concept decompositions for large sparse text data using clustering, Mach. Learn. 29 (2–3) (2001) 103–130.
[12] X.-H. Phan, C.-T. Nguyen, D.-T. Le, A hidden topic-based framework toward building applications with short web documents, IEEE Trans. Knowl. Data Eng. 23 (7) (2011) 961–976.
[13] D.-T. Vo, C.-Y. Ock, Learning to classify short text from scientific documents using topic models with various types of knowledge, Expert Syst. Appl. 42 (3) (2015) 1684–1698.
[14] A.-L. Maas, A.-Y. Ng, A probabilistic model for semanticword vectors, in: NIPS Workshop on Deep Learning and Unsupervised Feature Learning, ACM, 2010.
[15] A.-L. Maas, R.-E. Daly, P.-T. Pham, Learning word vectors for sentimentanalysis, in: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1. Association for Computational Linguistics, Association for Computational Linguistics, 2011, pp. 142–150.
[16] T. Mikolov, K. Chen, G. Corrado, Efficient estimation of word representations in vector space, 2013, arXiv preprint arXiv:1301.3781
[17] Q. Le, T. Mikolov, Distributed representations of sentences and documents, in: Proceedings of the 31th International Conference on Machine Learning, JMLR.org, 2014.
[18] Y. Bengio, R. Ducharme, P. Vincent, C. Jauvin, A neural probabilistic language model, J. Mach. Learn. Res. 3 (6) (2003) 1137–1155.
[19] R. Collobert, J. Weston, A unified architecture for natural language processing: Deep neural networks with multitask learning, in: Proceedings of the 25th international conference on Machine learning, ACM, 2008, pp. 160–167.
[20] A. Mnih, G.-E. Hinton, A scalable hierarchical distributed language model, in: Advances in neural information processing systems, Curran Associates, Inc., 2009, pp. 1081–1088.
[21] J. Turian, L. Ratinov, Y. Bengio, Word representations: a simple and general method for semi-supervised learning, in: Proceedings of the 48th annual meeting of the association for computational linguistics. Association for Computational Linguistics, The Association for Computer Linguistics, 2010, pp. 384–394.
[22] R. Socher, C.-C. Lin, C. Manning, Parsing natural scenes and natural language with recursive neural networks, Proceedings of the 28th international conference on machine learning, 2011, 129–136.
[23] Z.-S. Harris, Distributional structure, Word 10 (1954) 146–162.
[24] G. Salton, C. Buckley, Term-weighting approaches in automatic text retrieval, Inf. Process. Manage. 24 (5) (1988) 513–523.
[25] M. Kan, C.-L. Tan, H.-B. Low, Proposing a new term weighting scheme for text categorization, in: AAAI, 6, AAAI Press, 2006, pp. 763–768.
[26] J. Martineau, T. Finin, Delta TFIDF: an improved feature space forsentiment analysis, in: ICWSM, The AAAI Press, 2009.
[27] Y. Ko, A study of term weighting schemes using class information for text classification, in: Proc. ACM SIGIR, ACM, 2012, pp. 1029–1030.
[28] M. Sahami, T. Heilman, A web-based kernel function for measuring the similarityof short text snippets, in: Proceeding of WWW, ACM, 2006, pp. 377–386.
[29] D. Bollegala, Y. Matsuo, M. Ishizuka, Measuring semantic similarity between words using web search engines, in: Proceeding of WWW, 7, ACM, 2007, pp. 757–766.
[30] W. Yih, C. Meek, Improving similarity measures for short segments of text, in: Proceeding of AAAI, AAAI Press, 2007, pp. 1489–1494.
[31] S. Banerjee, K. Ramanathan, A. Gupta, Clustering short texts using wikipedia, in: Proceeding of SIGIR, ACM, 2007, pp. 787–788.
[32] L. Zhang, C. Li, J. Liu, H. Wang, Graph-based text similarity measurement by exploiting wikipedia as background knowledge, World Acad. Sci. Eng. Technol. 59 (2011) 1548–1553.
[33] Y. Zhu, L. Li, L. Luo, Learning to classify short text with topic model and external knowledge, Knowledge Science, Engineering and Management, Springer Berlin Heidelberg, 2013, pp. 493–503.
[34] H. Andreas, S. Staab, G. Stumme, Ontologies improve text document clustering, Data Mining (2003) 541–544.
[35] X. Hu, N. Sun, C. Zhang, T.-S. Chua, Exploiting internal and external semantics for the clustering of short texts using world knowledge, in: Proceeding of CIKM, ACM, 2009, pp. 919–928.
[36] Y.-Q. Song, H.-X. Wang, Z.-Y. Wang, H.-S. Li, W.-Z. Chen, Short text conceptualization using a probabilisticknowledgebase, in: Proceeding of IJCAI, IJCAI/AAAI, 2011, pp. 2330–2336.
[37] D.-M. Blei, A.-Y. Ng, M.-I. Jordan, Latent dirichlet allocation, J. Mach. Learn. Res. 3 (2003) 993–1022.
[38] M.-G. Chen, X.-M. Jin, D. Shen, Short text classification improved by learning multi-granularity topics, in: Proceeding of IJCAI, IJCAI/AAAI, 2011, pp. 1776–1781.
[39] X.-Q. Cheng, X.-H. Yan, Y.-Y. Lan, J.-F. Guo, BTM: topic modeling over short texts, IEEE Trans. Knowl. Data. Eng. 26 (12) (2014) 2928–2941.
[40] X.-H. Phan, JWikidocs: A java-based wikipedia crawling toolkit, jwebpro.sourceforge.net (2007).
[41] G. Heinrich, Parameter estimation for text analysis, Technical report (2005).
[42] Y. Liu, Z. Liu, T.-S. Chua, M. Sun, Topical word embeddings, in: AAAI, 2015.
[43] P. Wang, H. Zhang, Y.-F. Wu, A robust framework for short text categorization based on topic model andintegrated classifier, in: IJCNN, IEEE, 2014, pp. 3534–3539.
[44] P. Wang, J. Xu, Word embedding clustering and convolutional neural network for short text categorization, in: The 53nd Annual Meeting of the Association for Computational Linguistics, The Association for Computer Linguistics, 2015.
[45] P. Wang, H. Zhang, Short text feature enrichment using link analysis on topic-keyword graph, in: Natural Language Processing and Chinese Computing, Springer, 2014, pp. 79–90.