Topical Word Embeddings

Yang Liu[1], **Zhiyuan Liu**[1*], **Tat-Seng Chua**[2], **Maosong Sun**[1,3]
[1] Department of Computer Science and Technology, State Key Lab on Intelligent Technology and Systems, National Lab for Information Science and Technology, Tsinghua University, Beijing 100084, China
[2] School of Computing, National University of Singapore, Singapore
[3] Jiangsu Collaborative Innovation Center for Language Competence, Jiangsu 221009, China

Abstract

Most word embedding models typically represent each word using a single vector, which makes these models indiscriminative for ubiquitous homonymy and polysemy. In order to enhance discriminativeness, we employ latent topic models to assign topics for each word in the text corpus, and learn topical word embeddings (TWE) based on both words and their topics. In this way, contextual word embeddings can be flexibly obtained to measure contextual word similarity. We can also build document representations, which are more expressive than some widely-used document models such as latent topic models. In the experiments, we evaluate the TWE models on two tasks, contextual word similarity and text classification. The experimental results show that our models outperform typical word embedding models including the multi-prototype version on contextual word similarity, and also exceed latent topic models and other representative document models on text classification. The source code of this paper can be obtained from https://github.com/largelymfs/topical_word_embeddings.

Introduction

Word embedding, also known as word representation, plays an increasingly vital role in building continuous word vectors based on their contexts in a large corpus. Word embedding captures both semantic and syntactic information of words, and can be used to measure word similarities, which are widely used in various IR and NLP tasks.

Most word embedding methods assume each word preserves a single vector, which is problematic due to homonymy and polysemy. Multi-prototype vector space models (Reisinger and Mooney 2010) were proposed to cluster contexts of a word into groups, then generate a distinct prototype vector for each cluster. Following this idea, (Huang et al. 2012) proposed multi-prototype word embeddings based on neural language models (Bengio et al. 2003).

Despite of their usefulness, multi-prototype word embeddings face several challenges: (1) These models generate multi-prototype vectors for each word in isolation, ignoring complicated correlations among words as well as their contexts. (2) In multi-prototype setting, contexts of a word are divided into clusters with no overlaps. In reality, a word's several senses may correlate with each other, and there is not clear semantic boundary between them.

In this paper, we propose a more flexible and powerful framework for multi-prototype word embeddings, namely **topical word embeddings** (TWE), in which **topical word** refers to a word taking a specific topic as context. The basic idea of TWE is that, we allow each word to have different embeddings under different topics. For example, the word *apple* indicates a fruit under the topic *food*, and indicates an IT company under the topic *information technology (IT)*.

We employ the widely used latent Dirichlet allocation (LDA) (Blei, Ng, and Jordan 2003) to obtain word topics, and perform collapsed Gibbs sampling (Griffiths and Steyvers 2004) to iteratively assign latent topics for each word token. In this way, given a sequence of words $D = \{w_1, \ldots, w_M\}$, after LDA converges, each word token w_i will be discriminated into a specific topic z_i, forming a word-topic pair $\langle w_i, z_i \rangle$, which can be used to learn topical word embeddings. We design three TWE models to learn topical word vectors, as shown in Figure 1, where the window size is 1, and w_{i-1} and w_{i+1} are contextual words of w_i.

TWE-1. We regard each topic as a pseudo word, and learn topic embeddings and word embeddings separately. We then build the topical word embedding of $\langle w_i, z_i \rangle$ according to the embeddings of w_i and z_i.

TWE-2. We consider each word-topic pair $\langle w_i, z_i \rangle$ as a pseudo word, and learn topical word embeddings directly.

TWE-3. We reserve distinct embeddings for each word and each topic. We build the embedding of each word-topic pair, by concatenating the corresponding word and topic embeddings, for learning.

Among three TWE models, TWE-1 does not consider the immediate interaction between a word and its assigned topic for learning. TWE-2 considers the inner interaction of a word-topic pair by simply regarding the pair as a pseudo word, but it suffers from the sparsity issue because the occurrences of each word are rigidly discriminated into different topics. as compared to TWE-2, TWE-3 provides trade-off between discrimination and sparsity. But during the learning process of TWE-3, topic embeddings will influence the cor-

*Corresponding author: Zhiyuan Liu (liuzy@tsinghua.edu.cn).

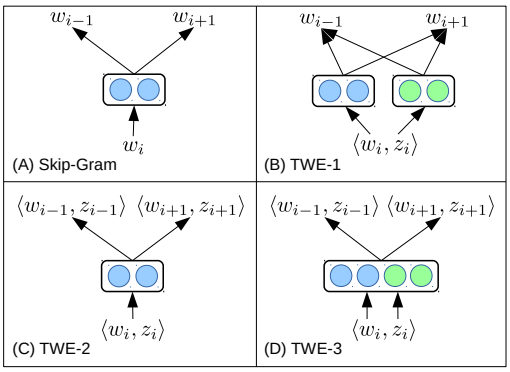

Figure 1: Skip-Gram and TWE models. Blue circles indicate word embeddings and green circles indicate topic embeddings. Since TWE-2 does not reserve stand-alone word / topic embeddings, we simply represent topical word embeddings in TWE-2 using blue circles.

responding word embeddings, which may make those words in the same topic less discriminative.

We extend Skip-Gram (Mikolov et al. 2013), the state-of-the-art word embedding model, to implement our TWE models. The three TWE models will be introduced in detail in next section. TWE models can be used to compute the contextual word embedding for a word given its context, and can also be used for representing a document aggregated from topical word embeddings of all words in it.

We tested our proposed models in two tasks, contextual word similarity and text classification, to evaluate our models. The experimental results demonstrate that our models outperform conventional and other multi-prototype word embedding models on contextual word similarity, and also exceed widely-used topic-based or embedding-based document models on text classification.

The main contribution of this work is that, we integrate topics into basic word embedding representation and allow the resulting topical word embeddings to model different meanings of a word under different context. As compared to multi-prototype word embedding models which build multi-prototypes of each word separately, our models employ topic models to take advantages of all words as well as their context together to learn topical word embeddings.

Our Models

Skip-Gram

Skip-Gram is a well-known framework for learning word vectors (Mikolov et al. 2013), as shown in Fig. 1(A). Skip-Gram aims to predict context words given a target word in a sliding window. In this framework, each word corresponds to a unique vector. The vector of target word is used as features to predict the context words.

Given a word sequence $D = \{w_1, \ldots, w_M\}$, the objective of Skip-Gram is to maximize the average log probability

$$\mathcal{L}(D) = \frac{1}{M} \sum_{i=1}^{M} \sum_{-k \le c \le k, c \neq 0} \log \Pr(w_{i+c}|w_i). \quad (1)$$

Here k is the context size of a target word. Skip-Gram formulates the probability $\Pr(w_c|w_i)$ using a softmax function as follows

$$\Pr(w_c|w_i) = \frac{\exp(\mathbf{w}_c \cdot \mathbf{w}_i)}{\sum_{w_i \in W} \exp(\mathbf{w}_c \cdot \mathbf{w}_i)}, \quad (2)$$

where $\mathbf{w}_i$ and $\mathbf{w}_c$ are respectively the vector representations of target word w_i and context word w_c, and W is the word vocabulary. In order to make the model efficient for learning, the techniques of hierarchical softmax and negative sampling are used when learning Skip-Gram (Mikolov et al. 2013).

Word vectors learned with Skip-Gram can be used for computing word similarities. The similarity of two words w_i and w_j can simply be measured with inner product of their word vectors, namely $S(w_i, w_j) = \mathbf{w}_i \cdot \mathbf{w}_j$. In previous work, the task of word similarity is always used to evaluate the performance of word embedding methods (Mikolov et al. 2013; Baroni, Dinu, and Kruszewski 2014).

As discussed in the Introduction section, we want to enhance representation capability of word embeddings by introducing latent topic models. With the favor of latent Dirichlet allocation (LDA), we assign a latent topic $z_i \in T$ for each word w_i, according to the probability $\Pr(z_i|w_i, d) \propto \Pr(w_i|z_i) \Pr(z_i|d)$. Afterwards, we propose three models of topical word embeddings (TWE).

TWE-1

TWE-1 aims to learn vector representations for words and topics separately and simultaneously. For each target word with its topic $\langle w_i, z_i \rangle$, we propose TWE-1 as follows. The objective of TWE-1 is defined to maximize the following average log probability

$$\mathcal{L}(D) = \frac{1}{M} \sum_{i=1}^{M} \sum_{-k \le c \le k, c \neq 0} \log \Pr(w_{i+c}|w_i) + \log \Pr(w_{i+c}|z_i). \quad (3)$$

Compared with only using the target word w_i to predict context words in Skip-Gram, TWE-1 also uses the topic z_i of target word to predict context words. The basic idea of TWE-1 is to regard each topic as a pseudo word that appears in all positions of words assigned with this topic. Hence, the vector of a topic will represent the collective semantics of words under this topic.

In TWE-1, we get topical word embedding of a word w in topic z by concatenating the embedding of w and z, i.e., $\mathbf{w}^z = \mathbf{w} \oplus \mathbf{z}$, where $\oplus$ is the concatenation operation, and the length of $\mathbf{w}^z$ is double of $\mathbf{w}$ or $\mathbf{z}$.

Contextual Word Embedding TWE-1 can be used for contextual word embedding. For each word w with its context c, TWE-1 will first infer the topic distribution $\Pr(z|w, c)$ by regarding c as a document, namely $\Pr(z|w, c) \propto \Pr(w|z) \Pr(z|c)$. With the topic distribution, we can further obtain the contextual word embedding of w in c as

$$\mathbf{w}^c = \sum_{z \in T} \Pr(z|w, c) \mathbf{w}^z, \quad (4)$$

where $\mathbf{w}^z$ is the embedding of word w under topic z, obtained by concatenating word vector $\mathbf{w}$ and topic vector $\mathbf{z}$.

Contextual word embedding will be used for computing contextual word similarity. Given a pair of words with their contexts, namely (w_i, c_i) and (w_j, c_j), contextual word similarity aims to measure the similarity between the two words, which can be formalized as follows $S(w_i, c_i, w_j, c_j) = \left(\mathbf{w}_i^{c_i} \cdot \mathbf{w}_j^{c_j}\right)$, which can also be rewritten as

$$\sum_{z \in T} \sum_{z' \in T} \Pr(z|w_i, c_i) \Pr(z'|w_j, c_j) S(\mathbf{w}^z, \mathbf{w}^{z'}), \quad (5)$$

where $S(\mathbf{w}^z, \mathbf{w}^{z'})$ is the similarity between $\mathbf{w}^z$ and $\mathbf{w}^{z'}$, which is calculated using cosine similarity in this paper. The similarity function in Eq. (5) is named as `AvgSimC` following (Reisinger and Mooney 2010).

We also employ the idea in (Reisinger and Mooney 2010) to define `MaxSimC` as an alterative to `AvgSimC`. `MaxSimC` selects the corresponding topical word embedding $\mathbf{w}^z$ of the most probable topic z inferred using w in context c as the contextual word embedding. Following this idea, we define the contextual word embedding of w in c as

$$\mathbf{w}^c = \mathbf{w}^z, \quad z = \arg\max_z \Pr(z|w, c), \quad (6)$$

and the contextual word similarity is defined as

$$S(w_i, c_i, w_j, c_j) = \mathbf{w}_i^z \cdot \mathbf{w}_j^{z'}, \quad (7)$$

where

$$z = \arg\max_z \Pr(z|w_i, c_i), \quad z' = \arg\max_z \Pr(z|w_j, c_j).$$

Document Embedding In TWE-1, we represent the semantics of a document by aggregating over all topical word embeddings of each word in the document, i.e., $\mathbf{d} = \sum_{w \in d} \Pr(w|d)\mathbf{w}^z$, where $\Pr(w|d)$ can be weighted with TFIDF scores of words in d.

TWE-2

Topic models group words into various topics according to their semantic meanings. In other words, a word in different topics may correspond to different meanings. This makes sense because many words have multiple senses, and these senses should have discriminative vectors in semantic space.

Hence, in TWE-2 we regard each word-topic pair $\langle w, z\rangle$ as a pseudo word and learn a unique vector $\mathbf{w}^z$. The objective of TWE-2 is to maximize the average log probability

$$\mathcal{L}(D) = \frac{1}{M} \sum_{i=1}^{M} \sum_{-k \le c \le k, c \neq 0} \log \Pr(\langle w_{i+c}, z_{i+c}\rangle | \langle w_i, z_i\rangle), \quad (8)$$

where $\Pr(\langle w_c, z_c\rangle|\langle w_i, z_i\rangle)$ is also a softmax function

$$\Pr(\langle w_c, z_c\rangle|\langle w_i, z_i\rangle) = \frac{\exp(\mathbf{w}_c^{z_c} \cdot \mathbf{w}_i^{z_i})}{\sum_{\langle w_c, z_c\rangle \in \langle W, T\rangle} \exp(\mathbf{w}_c^{z_c} \cdot \mathbf{w}_i^{z_i})}. \quad (9)$$

In this way, TWE-2 naturally divides the representation of a word w into T parts, having $\mathbf{w} = \sum_{z \in T} \Pr(z|w)\mathbf{w}_z$, where $\Pr(z|w)$ can be obtained from LDA.

In TWE-2, we follow Eq. (4) to obtain the contextual word embedding of a word with its context, which will be further used to compute contextual word similarity. For document embedding, we follow the same idea of TWE-1 and obtain a document embedding by $\mathbf{d} = \sum_{\langle w, z\rangle \in d} \Pr(\langle w, z\rangle|d)\mathbf{w}^z$.

TWE-3

Since TWE-2 divides the occurrences of each word into multiple topics, the learning of embeddings may suffer from more sparsity issue as compared to Skip-Gram. In order to alleviate the problem, we propose TWE-3 to provide a trade-off between discrimination and sparsity.

Similar to TWE-1, TWE-3 has vectors of both words and topics. For each word-topic pair $\langle w, z\rangle$, TWE-3 concatenates the vectors $\mathbf{w}$ and $\mathbf{z}$ to build the vector $\mathbf{w}^z$, with $|\mathbf{w}^z| = |\mathbf{w}| + |\mathbf{z}|$. Note that, the length of word vectors $|\mathbf{w}|$ is not necessarily the same as the length of topic vectors $|\mathbf{z}|$.

The objective of TWE-3 is identical to TWE-2, as shown in Eq. (8), and the probability $\Pr(\langle w_c, z_c\rangle|\langle w_i, z_i\rangle)$ is formalized as in Eq. (9). In TWE-3, the parameters of each word embedding $\mathbf{w}$ and topic embedding $\mathbf{z}$ is shared over all word-topic pairs $\langle w, z\rangle$; while in TWE-2, each word-topic pair $\langle w, z\rangle$ have their own parameters. Hence, in TWE-2 two word-topic pairs $\langle w, z\rangle$ and $\langle w, z'\rangle$ will have distinct parameters, but in TWE-3 they share the same word embeddings $\mathbf{w}$.

In TWE-3, the construction of contextual word embeddings and document embeddings will follow that of TWE-1.

Optimization and Parameter Estimation

Learning TWE models follows the similar optimization scheme as that of Skip-Gram used in (Mikolov et al. 2013). We use stochastic gradient descent (SGD) for optimization, and gradients are calculated using the back-propagation algorithm.

Initialization is important for learning TWE models. In TWE-1, we first learn word embeddings using Skip-Gram. Afterwards, we initialize each topic vector with the average over all words assigned to this topics, and learn topic embeddings while keeping word embeddings unchanged. In TWE-2, we initialize the vector of each topic-word pair with the corresponding word vector from Skip-Gram, and learn TWE models. In TWE-3, we initialize word vectors using those from Skip-Gram, and topic vectors using those from TWE-1, and learn TWE models.

Complexity Analysis

In Table 1, we show the complexity of various models, including the number of model parameters and computational complexity. In this table, the topic number is T, the vocabulary size is W, the window size is C, the corpus length is M, and the vector lengths of words and topics are K_W and K_T, respectively. In Skip-Gram, TWE-1 and TWE-2, we have $K_W = K_T = K$. Note that, we do not count parameters of topic models into model parameters of TWE, whose number is about $O(WT)$.

Table 1: Model complexities.

Model	Model Parameters	Computational Complexity
Skip-Gram	WK	$CM(1 + \log W)$
TWE-1	$(W + T)K$	$IM + 2CM(1 + \log W)$
TWE-2	WTK	$IM + CM(1 + \log WT)$
TWE-3	$WK_W + TK_T$	$IM + CM(1 + \log WT)$

In computational complexity, $O(ID)$ indicates the computational complexity of learning topic models, where I is the iteration numbers; the second term is the complexity of learning topical word embeddings with hierarchical softmax.

From the complexity analysis, we can see that, compared with Skip-Gram, TWE models require more parameters to record more discriminative information for word embeddings, but the computational complexity does not increase too much relatively, especially when many fast algorithms of topic models are available (Porteous et al. 2008; Smola and Narayanamurthy 2010; Ahmed et al. 2012).

Experiments

In this section, we evaluate related models on two tasks empirically, including contextual word similarity and text classification, then we present some examples of topical word embeddings for intuitive comprehension.

Contextual Word Similarity

Traditional task for evaluating word embeddings is word similarity computation ignoring context information using standard datasets such as WordSim353 (Finkelstein et al. 2001). Nevertheless, semantic meanings of most words depend highly on their context. Hence we use the task of contextual word similarity to demonstrate the effectiveness of topical word embeddings.

Datasets and Experiment Setting We use the dataset released by (Huang et al. 2012) for evaluation, named as SCWS in this paper following (Luong, Socher, and Manning 2013). In SCWS, there are $2,003$ pairs of words and sentences containing these words; and human labeled word similarity scores are based on the word meanings in the context. We compute the Spearman correlation between similarity scores from a system and the human judgements in the dataset for comparison. For TWE models, the similarity between two words are computed using Eq. (5).

We select Wikipedia, the largest online knowledge base, to learn topical word embeddings for this task. We adopt the April 2010 dump also used by (Huang et al. 2012) [1]. When learning topic assignments with LDA, we set $T = 400$ and $I = 50$. When learning Skip-Gram and TWE models, we set window size as 5 and the dimensions of both word embeddings and topic embeddings as $K = 400$. For TWE-1 and TWE-3, we obtain topical word embeddings via concatenation over corresponding word embeddings and topic embeddings; and for TWE-2, topical word embeddings are off-the-shelf.

In experiments, we compare our models with baselines including C&W model (Collobert and Weston 2008), TFIDF (including Pruned TFIDF and Pruned TFIDF-M) (Reisinger and Mooney 2010), Huang's model (Huang et al. 2012), Tian's model (Tian et al. 2014), LDA and Skip-Gram. Since we evaluate on the same data set as (Huang et al. 2012; Tian et al. 2014), we simply report the evaluation results of C&W, TFIDF and Huang's model from (Huang et al. 2012) and the results of Tian's model from (Tian et al. 2014). Among these baselines, C&W, TFIDF, Pruned TFIDF, LDA-S, LDA-C and Skip-Gram are single-prototype models, while Pruned TFIDF-M, Huang's model and Tian's model are multi-prototype models.

We give a brief introduction to these baseline methods. C&W model is evaluated using word embeddings provided by (Collobert and Weston 2008), ignoring context information. The TFIDF methods represent words using context words within 10-word windows, weighted by TFIDF. (Reisinger and Mooney 2010) found pruning the context words of low TFIDF scores can improve performance, and (Huang et al. 2012) reported the result of the pruned method by removing all but the top 200 words in each word vector. For LDA, we have two methods for computing contextual word similarity. **LDA-S** represents each word using its topic distribution $\Pr(z|w) \propto \Pr(w|z)\Pr(z)$ ignoring context, where $\Pr(w|z)$ and $\Pr(z)$ can be obtained from the LDA model. **LDA-C** regards the context sentence c of the word w as a document, and represents the word with its contextual topic distribution $\Pr(z|w, c)$ for computing word similarities. It is obvious that LDA-S ignores context and LDA-C is context-aware. Here we simply report the results of the LDA model used in TWE models. In Skip-Gram, we simply compute similarities using word embeddings ignoring context. Here the dimension of word embeddings in Skip-Gram is $K = 400$. Tian's model proposed a probabilistic model for learning multi-prototype word embeddings, which achieves comparable performance but is more efficient as compared to Huang's model.

Evaluation Result In Table 2, we show the evaluation results of various models on the SCWS dataset. For all multi-prototype models and the TWE models, we report the evaluation results using both `AvgSimC` and `MaxSimC`. We have the following observations from Table 2:

Table 2: Spearman correlation $\rho \times 100$ of contextual word similarity on the SCWS data set.

Model	$\rho \times 100$	
C&W	57.0	
TFIDF	26.3	
Pruned TFIDF	62.5	
LDA-S	56.9	
LDA-C	50.4	
Skip-Gram	65.7	
	`AvgSimC`	`MaxSimC`
Pruned TFIDF-M	60.5	60.4
Tian	65.4	63.6
Huang	65.3	58.6
TWE-1	**68.1**	**67.3**
TWE-2	67.9	63.6
TWE-3	67.1	65.5

The TWE models outperform all baselines, and TWE-1 achieves the best performance. This indicates that the TWE models can effectively take advantages of latent topic models for topical word embeddings.

[1] The dataset and the vocabulary file can be downloaded from http://www.socher.org/.

For multi-prototype models, the TWE models are significantly better than other baselines when computing with MaxSimC. The TWE models can represent the semantics of a contextual word more precisely by picking the most probable topical word embedding. This suggests the following two advantages of the TWE models as compared to other multi-prototype models: (1) In most multi-prototype models, each word usually have limited number of prototypes (e.g., Huang's model set the number as 10). The TWE models, by taking advantages of latent topic models, can discriminate more details of word semantics in the topic space, which usually has hundreds of dimension (e.g., we set the number of topics $T = 400$ in this paper). (2) Most multi-prototype models build multi-prototypes for each word separately, ignoring rich interactions between words as well as their contexts. By adopting topic models, we are able to discriminate word semantics by considering words and their contexts all together. The adoption of topic models also provides us a more principled way to select the most appropriate topical word embedding for a word under specific context.

The performance of three TWE models reflect their characteristics. (1) TWE-2 is comparable to TWE-1 when using AvgSimC, but comparably under-performs when using MaxSimC. We guess the reason is, when using MaxSimC, the sparsity issue of TWE-2, caused by learning distinct embeddings for each word-topic pair, is more significant. (2) The performance of TWE-2 is between TWE-1 and TWE-3. The reason is probable that during learning of TWE-3, topic embeddings will influence the word embeddings and make the topical word embeddings within one topic less discriminative.

Text Classification

Here we investigate the effectiveness of TWE models for document modeling using multi-class text classification.

Datasets and Experiment Setting Multi-class text classification is a well studied problem in NLP and IR. In this paper, we run the experiments on the dataset 20NewsGroup [2]. 20NewsGroup consists of about $20,000$ documents from 20 different newsgroups. We report macro-averaging precision, recall and F-measure for comparison.

For TWE models, we learn topic models using LDA on the training set by setting the number of topics $T = 80$. We further learn topical word embeddings using the training set, then generate document embeddings for both training set and test set. Afterwards, we regard document embedding vectors as document features and train a linear classifier using Liblinear (Fan et al. 2008). We set the dimensions of both word and topic embeddings as $K = 400$.

We consider the following baselines, bag-of-words (BOW) model, LDA, Skip-Gram, and Paragraph Vector (PV) models (Le and Mikolov 2014). The BOW model represents each document as a bag of words and the weighting scheme is TFIDF. For the TFIDF method, we select top $50,000$ words according to TFIDF scores as features. LDA represents each document as its inferred topic distribution. Here we use the LDA model in TWE models. In Skip-Gram, we build the embedding vector of a document by simply averaging over all word embedding vectors in this document. The dimension of word embeddings in Skip-Gram is also $K = 400$. Paragraph Vector models are document embedding models proposed most recently, including the distributed memory model (PV-DM) and the distributed bag-of-words model (PV-DBOW). PV models are reported to achieve the state-of-the-art performance on sentiment classification (Le and Mikolov 2014). (Le and Mikolov 2014) have not released source codes of PV models. There is an implementation doc2vec available online [3]. In the experiments, we find the performance of doc2vec is comparably worse than our implementation, hence we only report the results of our implementation for comparison.

Evaluation Results Table 3 shows the evaluation results of text classification on 20NewsGroup. We can observe that TWE-1 outperforms all baselines significantly, especially for topic models and embedding models. This indicates that our model can capture more precise semantic information of documents as compared to topic models and embedding models. Moreover, as compared to the BOW model, the TWE models manage to reduce the document feature space by 99.2 percent in this case.

Table 3: Evaluation results of multi-class text classification.

Model	Accuracy	Precision	Recall	F-measure
BOW	79.7	79.5	79.0	79.0
LDA	72.2	70.8	70.7	70.0
Skip-Gram	75.4	75.1	74.3	74.2
PV-DM	72.4	72.1	71.5	71.5
PV-DBOW	75.4	74.9	74.3	74.3
TWE-1	**81.5**	**81.2**	**80.6**	**80.6**
TWE-2	79.0	78.6	77.9	77.9
TWE-3	77.4	77.2	76.2	76.1

Among three TWE models, it is amazing that the simplest TWE-1 model again achieves the best performance. As inspired by the anonymous reviewer, word and topic embeddings are learned independently in TWE-1 but are built interactively in TWE-2 and TWE-3. The independence assumption in TWE-1 may be the reason of better performance. Moreover, the size of 20NewsGroup is to some extent small, we guess TWE-2 and TWE-3 may achieve better performance give more data for learning. In future, we will conduct more experiments to explore genuine reasons why the simpler model achieved the best performance.

Examples of Topical Word Embeddings

In order to demonstrate the characteristics of TWE models, we selected several example words and used TWE models to find the most similar words of these words in different topics. For comparison, we also used Skip-Gram to find similar words of these example words.

[2] http://qwone.com/~jason/20Newsgroups/.

[3] It can be downloaded from https://github.com/ccri/gensim, which will be released in the package gensim.

In Table 4, we show the most similar words of three example words, *bank*, *left* and *apple*, which are typical polysemous words. For each example word w, we first show the result obtained from Skip-Gram, i.e., the first line of each example word; then we list the results under two representative topics of the example word obtained from TWE-2, denoted as $w\#1$ and $w\#2$.

Table 4: Nearest neighbor words by TWE-2 and Skip-Gram.

Words	Similar Words
bank	citibank, investment, river
bank#1	insurance, stock, investor
bank#2	river, edge, coast
left	right, leave, quit
left#1	moved, arrived, leave
left#2	right, bottom, hand
apple	macintosh, ios, juice
apple#1	peach, juice, strawberry
apple#2	mac, ipod, android

From Table 4, we can observe that, similar words returned by Skip-Gram contain similar words of multiple senses of example words. This indicates that Skip-Gram combines multiple senses of a polysemous word into a unique embedding vector. In contrast, with TWE models, we can successfully discriminate word senses into multiple topics by topical word embeddings.

Related Work

The success of IR and NLP tasks crucially depend on text representation, of which word representation is the foundation. Conventionally, NLP tasks usually take one-hot word representation, with each word being represented as a W-length vector with only one non-zero entry. The one-hot representation is simple and has been widely used as the basis of bag-of-words (BOW) document models (Manning, Raghavan, and Schütze 2008). However, it suffers from several challenges, the most critical one of which is it cannot take the relationship between words into consideration, while in fact many words share high semantic or syntactic relations.

Word embeddings, first proposed in (Rumelhart, Hintont, and Williams 1986), have been successfully used in language models (Bengio et al. 2006; Mnih and Hinton 2008) and many NLP tasks, such as named entity recognition (Turian, Ratinov, and Bengio 2010), disambiguation (Collobert et al. 2011) and parsing (Socher et al. 2011; 2013). Word embeddings are useful because they can encode both syntactic and semantic information of words into continuous vectors and similar words are close in vector space.

Previous word embedding models are time consuming due to high computational complexity. Recently, (Mikolov et al. 2013) proposed two efficient models, Skip-Gram and continuous bag-of-words model (CBOW), to learn word embeddings from a large-scale text corpus. The training objective of CBOW is to combine the embeddings of context words to predict the target word; while Skip-Gram is to use the embedding of each target word to predict its context words (Mikolov et al. 2013). In this paper, we base on Skip-Gram to extend our models.

In most previous word embedding models, one word owns a unique vector, which is problematic because many words have multiple senses. Hence, researchers propose multi-prototype models. (Reisinger and Mooney 2010) proposed a multi-prototype vector space model, which cluster contexts of each target word into groups, and build context vectors for each cluster. Following this idea, (Huang et al. 2012) also clustered contexts, and each cluster generated a distinct prototype embedding. Besides, probabilistic models (Tian et al. 2014), bilingual resources (Guo et al. 2014) and non-parametric models (Neelakantan et al. 2014) have been explored for multi-prototype word embeddings. Most of these methods perform multi-prototype modeling for each word in isolation. On the contrary, TWE models use latent topic models to discriminate word senses by considering all words and their contexts together. TWE models are also applicable for document embeddings. Moreover, multi-prototype models can be incorporated in TWE models easily, which will be left as future work.

Conclusion and Future Work

In this paper, we propose three topical word embedding models, which can be expressively used for contextual word embeddings and document embeddings. We evaluate our TWE models on two tasks including contextual word similarity and text classification. The experimental results show that our models especially the simplest TWE-1 outperform state-of-the-art word embedding models for contextual word similarity and are also competitive for text classification.

We consider the following future research directions:

- In LDA the topic number must be pre-defined. Cross-validation can be used to find appropriate topic number but are time-consuming and impractical for large-scale data. We will explore non-parametric topic models (Teh et al. 2006) for topical word embeddings.
- There are many knowledge bases available, such as WordNet (Miller 1995), containing rich linguistic knowledge of homonymy and polysemy. We may explore techniques to incorporate these prior linguistic knowledge into topical word embeddings.
- Documents usually contain additional information such as categorical labels, hyperlinks and timestamps. We may take these information for learning more representative topic models (Mcauliffe and Blei 2008; Zhu, Ahmed, and Xing 2009; Lacoste-Julien, Sha, and Jordan 2009) and enhance topical word embeddings.

Acknowledgments

This research is supported by the 973 Program (No. 2014CB340501), the National Natural Science Foundation of China (NSFC No. 61133012 and 61170196), and the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Office. We thank Shaohua Li for helpful discussions. We thank all anonymous reviewers for their constructive comments.

References

Ahmed, A.; Aly, M.; Gonzalez, J.; Narayanamurthy, S.; and Smola, A. J. 2012. Scalable inference in latent variable models. In *Proceedings of WSDM*, 123–132.

Baroni, M.; Dinu, G.; and Kruszewski, G. 2014. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of ACL*, 238–247.

Bengio, Y.; Ducharme, R.; Vincent, P.; and Jauvin, C. 2003. A neural probabilistic language model. *JMLR* 3:1137–1155.

Bengio, Y.; Schwenk, H.; Senécal, J.-S.; Morin, F.; and Gauvain, J.-L. 2006. Neural probabilistic language models. In *Innovations in Machine Learning*. 137–186.

Blei, D. M.; Ng, A. Y.; and Jordan, M. I. 2003. Latent dirichlet allocation. *JMLR* 3:993–1022.

Collobert, R., and Weston, J. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of ICML*, 160–167. ACM.

Collobert, R.; Weston, J.; Bottou, L.; Karlen, M.; Kavukcuoglu, K.; and Kuksa, P. 2011. Natural language processing (almost) from scratch. *JMLR* 12:2493–2537.

Fan, R.-E.; Chang, K.-W.; Hsieh, C.-J.; Wang, X.-R.; and Lin, C.-J. 2008. Liblinear: A library for large linear classification. *JMLR* 9:1871–1874.

Finkelstein, L.; Gabrilovich, E.; Matias, Y.; Rivlin, E.; Solan, Z.; Wolfman, G.; and Ruppin, E. 2001. Placing search in context: The concept revisited. In *Proceedings of WWW*, 406–414.

Griffiths, T. L., and Steyvers, M. 2004. Finding scientific topics. *PNAS* 101:5228–5235.

Guo, J.; Che, W.; Wang, H.; and Liu, T. 2014. Learning sense-specific word embeddings by exploiting bilingual resources. In *Proceedings of COLING*, 497–507.

Huang, E. H.; Socher, R.; Manning, C. D.; and Ng, A. Y. 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of ACL*, 873–882.

Lacoste-Julien, S.; Sha, F.; and Jordan, M. I. 2009. Disclda: Discriminative learning for dimensionality reduction and classification. In *Proceedings of NIPS*, 897–904.

Le, Q. V., and Mikolov, T. 2014. Distributed representations of sentences and documents. 873–882.

Luong, M.-T.; Socher, R.; and Manning, C. 2013. Better word representations with recursive neural networks for morphology. In *Proceedings of CoNLL*.

Manning, C. D.; Raghavan, P.; and Schütze, H. 2008. *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge.

Mcauliffe, J. D., and Blei, D. M. 2008. Supervised topic models. In *Proceedings of NIPS*, 121–128.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of NIPS*, 3111–3119.

Miller, G. A. 1995. Wordnet: a lexical database for english. *Communications of the ACM* 38(11):39–41.

Mnih, A., and Hinton, G. E. 2008. A scalable hierarchical distributed language model. In *Proceedings of NIPS*, 1081–1088.

Neelakantan, A.; Shankar, J.; Passos, A.; and McCallum, A. 2014. Efficient non-parametric estimation of multiple embeddings per word in vector space. In *Proceedings of EMNLP*.

Porteous, I.; Newman, D.; Ihler, A.; Asuncion, A.; Smyth, P.; and Welling, M. 2008. Fast collapsed gibbs sampling for latent dirichlet allocation. In *Proceedings of KDD*, 569–577.

Reisinger, J., and Mooney, R. J. 2010. Multi-prototype vector-space models of word meaning. In *Proceedings of HLT-NAACL*, 109–117.

Rumelhart, D. E.; Hintont, G. E.; and Williams, R. J. 1986. Learning representations by back-propagating errors. *Nature* 323(6088):533–536.

Smola, A., and Narayanamurthy, S. 2010. An architecture for parallel topic models. *Proceedings of VLDB* 3(1-2):703–710.

Socher, R.; Lin, C. C.; Ng, A.; and Manning, C. 2011. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of ICML*, 129–136.

Socher, R.; Bauer, J.; Manning, C. D.; and Ng, A. Y. 2013. Parsing with compositional vector grammars. In *Proceedings of ACL*, 455–465.

Teh, Y. W.; Jordan, M. I.; Beal, M. J.; and Blei, D. M. 2006. Hierarchical dirichlet processes. *JMLR* 101(476).

Tian, F.; Dai, H.; Bian, J.; Gao, B.; Zhang, R.; Chen, E.; and Liu, T.-Y. 2014. A probabilistic model for learning multi-prototype word embeddings. In *Proceedings of COLING*, 151–160.

Turian, J.; Ratinov, L.; and Bengio, Y. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of ACL*, 384–394.

Zhu, J.; Ahmed, A.; and Xing, E. P. 2009. Medlda: maximum margin supervised topic models for regression and classification. In *Proceedings of ICML*, 1257–1264.