Learning Context-Sensitive Word Embeddings with Neural Tensor Skip-Gram Model

Pengfei Liu, Xipeng Qiu[*] and Xuanjing Huang
Shanghai Key Laboratory of Data Science, Fudan University
School of Computer Science, Fudan University
825 Zhangheng Road, Shanghai, China
{pfliu14,xpqiu,xjhuang}@fudan.edu.cn

Abstract

Distributed word representations have a rising interest in NLP community. Most of existing models assume only one vector for each individual word, which ignores polysemy and thus degrades their effectiveness for downstream tasks. To address this problem, some recent work adopts multi-prototype models to learn multiple embeddings per word type. In this paper, we distinguish the different senses of each word by their latent topics. We present a general architecture to learn the word and topic embeddings efficiently, which is an extension to the Skip-Gram model and can model the interaction between words and topics simultaneously. The experiments on the word similarity and text classification tasks show our model outperforms state-of-the-art methods.

1 Introduction

Distributed word representations, also commonly called word embeddings, are to represent words by dense, low-dimensional and real-valued vectors. Each dimension of the embedding represents a latent feature of the word, hopefully capturing useful syntactic and semantic properties. Distributed representations help address the curse of dimensionality and improve generalization because they can group the words having similar semantic and syntactic roles. Therefore, distributed representations are widely used for many natural language process (NLP) tasks, such as syntax [Turian *et al.*, 2010; Collobert *et al.*, 2011; Mnih and Hinton, 2007], semantics[Socher *et al.*, 2012] and morphology [Luong *et al.*, 2013].

However, most of these methods use the same embedding vector to represent a word, which is somehow unreasonable and sometimes it even hurts the model's expression ability because a great deal of words are polysemous. For example, all the occurrences of the word "bank" will have the same embedding, irrespective of whether the context of the word suggests it means "a financial institution" or "a river bank", which results in the word "bank" having an embedding that is approximately the average of its different contextual semantics relating to finance or placement.

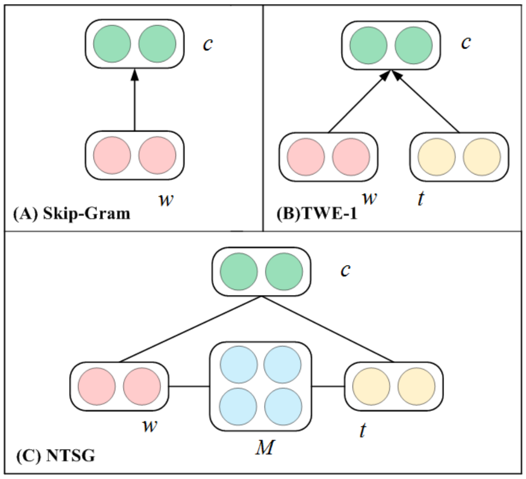

Figure 1: Skip-Gram, TWE-1 and our model(NTSG). The red, yellow and green circles indicate the embeddings of word, topic and the context respectively.

To address this problem, some models [Reisinger and Mooney, 2010; Huang *et al.*, 2012; Tian *et al.*, 2014; Neelakantan *et al.*, 2014] were proposed to learn multi-prototype word embeddings according to the different contexts. These models generate multi-prototype vectors by locally clustering the contexts for each individual word. This locality ignores the correlations among words as well as their contexts. To avoid this limitation, Liu *et al.*[2015] introduced latent topic model [Blei *et al.*, 2003] to globally cluster the words into different topics according to their contexts. They proposed three intuitive models (topical word embeddings, TWE) to enhance the discriminativeness of word embeddings. However, their models do not model clearly the interactions among the words, topics and contexts.

We assume that the single-prototype word embedding can be regarded as a mixture of its different prototypes, while the topic embedding is the averaged vector of all the words under this topic. Thus, the topic and single-prototype word embeddings should be regarded as two kinds of clustering of word senses from different views. The topic embeddings and

[*] Corresponding author

single-prototype word embeddings should have certain relations and should be modeled jointly. Thus, given a word with its topic, a specific sense of the word can be determined by its topic, the context-sensitive word embedding (also called topical word embedding) should be obtained by integrating word vector and topic vector.

In this paper, we propose a neural tensor skip-gram model (NTSG) to learn the distributed representations of words and topics, which is an extension to the Skip-Gram model and replaces the bilinear layer with a tensor layer to capture more interactions between word and topic under different contexts. Figure 1 illustrates the differences among Skip-Gram, TWE and our model. Experiments show qualitative improvements of our model over single-sense Skip-Gram on word neighbors. We also perform empirical comparisons on two tasks, contextual word similarity and text classification, which demonstrate the effectiveness of our model over the other state-of-the-art multi-prototype models.

The main contributions of this work are as follows.

1. Our model is a general architecture to learn multi-prototype word embeddings, and uses a tensor layer to model the interaction of words and topics. We also show the Skip-Gram and TWE models can be regarded as special cases of our model.

2. To improve the efficiency of the model , we use a low rank tensor factorization approach that factorizes each tensor slice as the product of two low-rank matrices.

2 Neural Models For Word Embeddings

Although there are many methods to learn vector representations for words from a large collection of unlabeled data, here we focus only on the most relevant methods to our model. Bengio *et al.*[2003] represents each word token by a vector for neural language models and estimates the parameters of the neural network and these vectors jointly. Since this model is quite expensive to train, much research has focused on optimizing it, such as C&W embeddings [Collobert and Weston, 2008] and Hierarchical log-linear (HLBL) embeddings [Mnih and Hinton, 2007]. A recent considerable interesting work, word2vec [Mikolov *et al.*, 2013a], uses extremely computationally efficient log-linear models to produce high-quality word embeddings, which includes two models: CBOW and Skip-gram models [Mikolov *et al.*, 2013b].

Skip-Gram is an effective framework for learning word vectors, which aims to predict surrounding words given a target word in a sentence [Mikolov *et al.*, 2013b]. In the Skip-Gram model, $\mathbf{w} \in \mathbb{R}^d$ is the vector representation of the word $w \in \mathcal{V}$, where $\mathcal{V}$ is the vocabulary and d is the dimensionality of word embedding.

Given a pair of words (w, c), the probability that the word c is observed in the context of the target word w is given by

$$Pr(D = 1|w, c) = \frac{1}{1 + \exp(-\mathbf{w}^T \mathbf{c})} \qquad (1)$$

where $\mathbf{w}$ and $\mathbf{c}$ are embedding vectors of w and c respectively.

The probability of not observing word c in the context of w is given by,

$$Pr(D = 0|w, c) = 1 - \frac{1}{1 + \exp(-\mathbf{w}^T \mathbf{c})} \qquad (2)$$

Given a training set $\mathcal{D}$, the word embeddings are learned by maximizing the following objective function:

$$J(\theta) = \sum_{w,c \in \mathcal{D}} Pr(D = 1|w, c) + \sum_{w,c \in \mathcal{D}'} Pr(D = 0|w, c), \qquad (3)$$

where the set $\mathcal{D}'$ is randomly sampled negative examples, assuming they are all incorrect.

3 Neural Tensor Skip-Gram Model

In order to enhance the representation capability of word embeddings, we introduce latent topics and assume that each word has 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)*.

Our goal is to be able to state whether a word w and its topic t can match well under the context c. For instance, $(w, t) = (apple, company)$ matches well under the context $c = iphone$, and (w, t)=$(apple, fruit)$ is a nice match under the context $c = banana$.

In this paper, we extend Skip-Gram model by replacing the bilinear layer with a tensor layer to capture the interactions between the words and topics under different contexts. A tensor is a geometric object that describes relations among vectors, scalars, and other tensors. It can be represented as a multi-dimensional array of numerical values. An advantage of the tensor is that it can explicitly model multiple interactions in data. As a result, tensor-based model have been widely used in a variety of tasks [Socher *et al.*, 2013a; 2013b].

To compute the score of how likely it is that word w and its topic t in a certain context word c, we use the following energy-based function:

$$g(w, c, t) = \mathbf{u}^T f(\mathbf{w}^T \mathbf{M}_c^{[1:k]} \mathbf{t} + \mathbf{V}_c^T (\mathbf{w} \oplus \mathbf{t}) + \mathbf{b}_c), \qquad (4)$$

where $\mathbf{w} \in R^d$,$\mathbf{t} \in R^d$ be the vector representations of the word w and topic t; $\oplus$ is the concatenation operation and $\mathbf{w} \oplus \mathbf{t} = \begin{bmatrix} \mathbf{w} \\ \mathbf{t} \end{bmatrix}$; $\mathbf{M}_c^{[1:k]} \in \mathbb{R}^{d \times d \times k}$ is a tensor, and the bilinear tensor product takes two vectors $\mathbf{w} \in \mathbb{R}^d$ and $\mathbf{t} \in \mathbb{R}^d$ as input, and generates a k-dimensional phrase vector $\mathbf{z}$ as output,

$$\mathbf{z} = \mathbf{w}^T \mathbf{M}_c^{[1:k]} \mathbf{t}, \qquad (5)$$

where each entry of $\mathbf{z}$ is computed by one slice $i = 1, \cdots, k$ of the tensor:

$$\mathbf{z}_i = \mathbf{w} \mathbf{M}_c^{[i]} \mathbf{t}. \qquad (6)$$

The other parameters in Eq. (4) are the standard form of a neural network: $\mathbf{u} \in \mathbb{R}^k$, $\mathbf{V}_c \in \mathbb{R}^{k \times (2d)}$ and $\mathbf{b}_c \in \mathbb{R}^k$. f is a standard nonlinearity applied element-wise, which is set to $f(t) = \frac{1}{1+\exp(-t)}$, same with Skip-Gram.

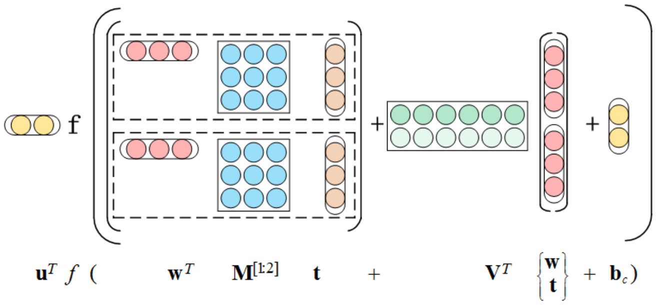
f
+
+
$\mathbf{u}^T\ f\ (\quad \mathbf{w}^T\ \mathbf{M}^{[1:2]}\ \mathbf{t}\ +\ \mathbf{V}^T \begin{Bmatrix}\mathbf{w}\\ \mathbf{t}\end{Bmatrix} + \mathbf{b}_c)$

Figure 2: Visualization of the Neural Tensor Network.

In Eq. (4), the tensor $\mathbf{M}_c^{[1:k]}$ depends on the context c. It is infeasible to assign a tensor to each context word c, therefore, we use the same tensor $\mathbf{M}^{[1:k]}$ for all contexts. Therefore, we rewrite Eq. (4) as

$$g(w, c, t) = \mathbf{u}^T f(\mathbf{w}^T \mathbf{M}^{[1:k]} \mathbf{t} + \mathbf{V}_c^T (\mathbf{w} \oplus \mathbf{t}) + \mathbf{b}_c). \quad (7)$$

Figure 2 shows a visualization of this model. The main advantage is that it models the latent relations among the words, topics and contexts jointly. Intuitively, the introduced tensor can incorporate the interaction between words and topics.

3.1 Tensor Factorization

Despite tensor-based transformation being effective for capturing the interactions, introducing tensor-based transformation into neural network models is time prohibitive since the tensor product operation drastically slows down the model. Without considering matrix optimization algorithms, the tensor operation complexity in Eq. (7) is $O(d^2k)$. Moreover, the additional tensor could bring millions of parameters to the model which makes the model suffer from the risk of overfitting. To remedy this, we propose a tensor factorization approach that factorizes each tensor slice as the product of two low-rank matrices. Formally, each tensor slice $\mathbf{M}^{[i]} \in \mathbb{R}^{d \times d}$ is factorized into two low rank matrix $\mathbf{P}^{[i]} \in \mathbb{R}^{d \times r}$ and $\mathbf{Q}^{[i]} \in \mathbb{R}^{r \times d}$:

$$\mathbf{M}^{[i]} = \mathbf{P}^{[i]} \mathbf{Q}^{[i]}, 1 \le i \le k \quad (8)$$

where $r \ll d$ is the number of factors.

$$g(w, c, t) = \mathbf{u}^T f(\mathbf{w}^T \mathbf{P}^{[1:k]} \mathbf{Q}^{[1:k]} \mathbf{t} + \mathbf{V}_c^T (\mathbf{w} \oplus \mathbf{t}) + \mathbf{b}_c), \quad (9)$$

The complexity of the tensor operation is now $O(rdk)$. As long as r is small enough, the factorized tensor operation would be much faster than the un-factorized one and the number of free parameters would also be much smaller, which prevents the model from overfitting.

3.2 Related Models and Special Cases

We now introduce several related models in increasing order of expressiveness and complexity. Each model assigns a score to triplet using a function g measuring how likely the word w is assigned to topic t under the context c.

Skip-Gram Skip-Gram is a well-know framework for learning word vector [Mikolov *et al.*, 2013b], as show in Figure 1(A). Skip-Gram aims to predict context words given a target word in a sliding window. Given a pair of words (w_i, c), we denote $Pr(c|w_i)$ as the probability that the word c is observed in the context of the target word w_i.

With negative-sampling approach, skip-Gram formulates the probability $Pr(c|w_i)$ as follows:

$$\begin{aligned} Pr(c|w_i) &\approx Pr(D = 1|\mathbf{w}_i, \mathbf{c}) && (10)\\ &= \frac{1}{1 + \exp(-\mathbf{w}_i^T \mathbf{c})} && (11)\\ &= f(-\mathbf{w}_i^T \mathbf{c}) && (12) \end{aligned}$$

where $Pr(D = 1|\mathbf{w}_i, \mathbf{c})$ is the probability that (w_i, c) came from the corpus data.

This model is a special case of our neural tensor model we set $f(t) = \frac{1}{1+\exp(-t)}$, $k = 1$, $\mathbf{M} = 0$, $\mathbf{b}_c = 0$ and $\mathbf{V}_c = \mathbf{c}$.

Topical Word Embeddings Liu *et al.*[2015] trained a similar model to learn topical word embeddings (TWE), as show in Figure 1(B), which uses the topic t_i of target word to predict context words compared with only using the target word w_i to predict context words in Skip-Gram [Mikolov *et al.*, 2013b]. They proposed three models with different combinations of word and topic. Here we just use their first model TWE-1 for comparison since TWE-1 achieves best results. TWE-1 regards each topic as a pseudo word that appears in all positions of words assigned with this topic.

$$\begin{aligned} Pr(c|w_i, t_i) &\approx Pr(c|w_i) Pr(c|t_i) && (13)\\ &\approx f\left((\mathbf{c} \oplus \mathbf{c})^T (\mathbf{w} \oplus \mathbf{t})\right). && (14) \end{aligned}$$

From Eq. (14), we can see that TWE-1 is also a special case of the neural tensor model if $k = 1$ and $\mathbf{M} = 0$, $\mathbf{b}_c = 0$ and $\mathbf{V}_c = (\mathbf{c} \oplus \mathbf{c})$. While this is an improvement over the Skip-Gram, the main problem with this model is that the parameters of the vector $\mathbf{w}$ and $\mathbf{t}$ do not interact with each other, and they are independently mapped to a common space.

Our Model Different with Skip-Gram and TWE, our model gives a more general framework to model the ternary relations among words, topics and contexts, as show in Figure 1(C). Skip-Gram and TWE can be regarded as special cases of our model. Our model incorporates the interaction of vector $\mathbf{w}$ and $\mathbf{t}$ in a simple and efficient way.

To get a different representations of a word type w in different contexts, we first get its topic t with LDA and get the context-sensitive representation by combining the embeddings of w and t. The simplest way is to concatenate the word and its topic embeddings, $\mathbf{w}^t = \mathbf{w} \oplus \mathbf{t}$.

3.3 Training

We use the contrastive max-margin criterion [Bordes *et al.*, 2013; Socher *et al.*, 2013a] to train our model. Intuitively, the max-margin criterion provides an alternative to probabilistic, likelihood-based estimation methods by concentrating directly on the robustness of the decision boundary of a model [Taskar *et al.*, 2005]. The main idea is that each triplet $\langle w, t, c\rangle$ coming from the training corpus should receives a higher score than a triplet in which one of the elements is replaced with a random elements. Let the set of all parameters be Ω, we minimize the following objective:

$$J(\Omega) = \sum_{\langle w,t,c\rangle \in \mathcal{D}} \sum_{\langle w,\hat{t},\hat{c}\rangle \in \hat{\mathcal{D}}} \max(0, 1- \tag{15}$$

$$g(w, t, c) + g(w, \hat{t}, \hat{c})\ ) + \lambda\|\Omega\|_2^2, \tag{16}$$

where $\mathcal{D}$ is the set of triplets from training corpus and we score the correct triplet higher than its corrupted one up to margin of 1. For each correct triplet we sample P random corrupted triplets. We used standard L_2 regularization of all the parameters, weighted by the hyperparameter λ. We have the following derivative for the j'th slice of the full tensor:

$$\frac{\partial g(w, c, t)}{\partial \mathbf{M}^{[j]}} = \mathbf{u}_j f'(\mathbf{z}_j)\mathbf{w}\mathbf{t}^T \tag{17}$$

where $\mathbf{z}_j = \mathbf{w}\mathbf{M}^{[j]}\mathbf{t} + \mathbf{V}_j^T(\mathbf{w} \oplus \mathbf{t})) + \mathbf{b}_j$, $\mathbf{V}_j$ is the jth row of the matrix $\mathbf{V}$ and we defined $\mathbf{z}_j$ as the jth element of the k-dimensional hidden tensor layer. We use SGD for optimization which converges to a local optimum of our non-convex objective function.

4 Experiments

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

In our experiments, we use four different settings of tensor $\mathbf{M}$ in the Eq. (7) as follows.

- NTSG-1 : We set $k = 1$ and $M^{[1]}$ is an identity matrix.
- NTSG-2 : We set $k = 1$ and $M^{[1]}$ is a full matrix.
- NTSG-3 : We set $k = 2$ and each tensor slice $M^{[i]}$ is factorized with two low rank matrices of $r = 50$.
- NTSG-4 : We set $k = 5$ and each tensor slice $M^{[i]}$ is factorized with two low rank matrices of $r = 50$.

4.1 Nearest Neighbors

Table 1 shows qualitatively the results of discovering multiple senses by presenting the nearest neighbors associated with various embeddings. For each word, we first show its nearest neighbors by the embeddings of Skip-Gram (the first line); the rest lines are the neighbor words under some representative topics, which are obtained by the topic and word embeddings of our model (NTSG-2 is used). The neighbor words returned by Skip-Gram are a mixture of multiple senses of the example word, which indicates that Skip-Gram combines multiple senses of a polysemous words into a unique embedding vector. In contrast, our model can successfully discriminate word senses into multiple topics by integrating the word and topic embeddings.

Words	Similar Words
bank	depositor, fdicinsured, river, idbi
bank:1	river, flood, road, hilltop
bank:2	finance, investment, stock, share
left	right, pass, leftside, front
left:1	leave, throw, put, go
left:2	right, back, front, forward
apple	blackberry, ipod, pear, macworld
apple:1	macintosh, iphone, inc, mirco
apple:2	cherry, peach, berry, orange
fox	wsvn, abc, urocyon, kttv
fox:1	wttg, kold-tv, wapt, wben-tv
fox:2	ferrell, watkin, eamonn, flanagans
fox:3	wolf, deer, beaver, boar
orange	citrus, yellow, yelloworang, lemon
orange:1	blue, maroon, brown, yellow
orange:2	pineapple, mango, grove, peach
run	wsvn, start, operate, pass
run:1	walk, go, chase, move
run:2	operate, running, driver, driven
plant	nonflowering, factory, flowering, nonwoody
plant:1	factory, distillate, subdepot, refinery
plant:2	warmseason, intercropped, seedling, highyield

Table 1: Nearest neighbor words by our model and Skip-Gram. The first line in each block is the results of Skip-Gram; and the rest lines are the results of our model.

In Figure 3, we present a visualization of high-dimensional topical word embeddings [1]. The left subfigure shows most of the words are clustered in different groups according to their topics. The right subfigure shows the two topical embeddings of the word *apple* and their neighbor words. We can see that our model can effectively discriminate the multiple senses of a word.

4.2 Contextual Word Similarity

We evaluate our embeddings on Stanford's Contextual Word Similarities (SCWS) dataset, developed by Huang *et al.*[2012]. There are 2003 word pairs in SCWS dataset, which includes 1328 noun-noun pairs, 399 verb-verb pairs, 140 verb-noun, 97 adjective-adjective, 30 noun-adjective, 9 verb-adjective, and 241 same-word pairs. The sentences containing these words are also provided. The human labeled similarity scores between words are based on the word meanings in the context. We compute the Spearman correlation between similarity scores from different models and the human judgements in the dataset for comparison.

We select Wikipedia, the largest online knowledge base, to learn topical word embeddings for this task. We adopt the April 2010 dump, which is also used by [Huang *et al.*, 2012].

The widely used collapsed Gibbs sampling LDA [Blei *et al.*, 2003; Griffiths and Steyvers, 2004] is used to obtain word topics. 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 t_i, forming a word-topic pair (w_i, t_i), which can be used to learn our model.

To make this a fair comparison, the partial parameters are set to same with [Liu *et al.*, 2015]. We set the number of topic $T = 400$ and iteration number $I = 50$. When learning Skip-Gram and our models, we set window size as 5 and the

[1] We use the t-SNE toolkit for visualization. http://lvdmaaten.github.io/software/

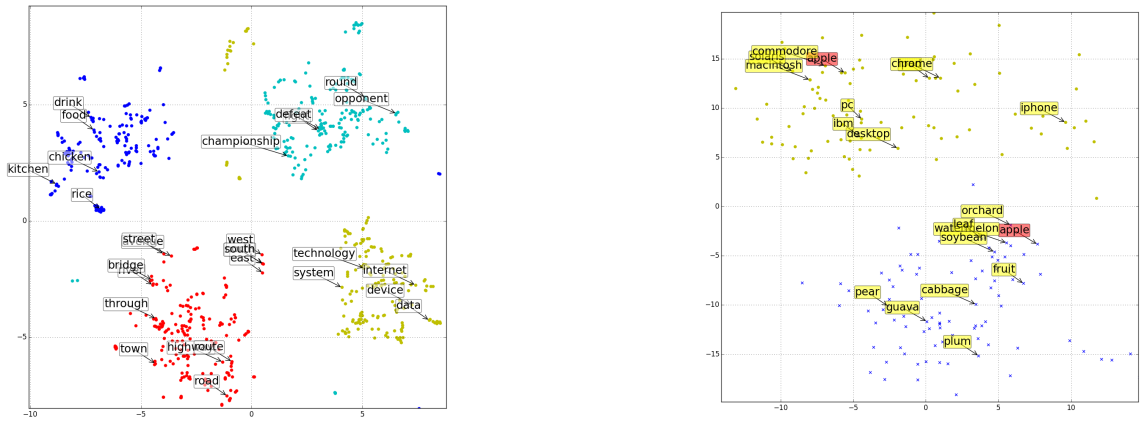

Figure 3: 2-D topical word embeddings of NTSG-2. The left one shows the topical word representations in four different topics. The right one shows two topical embeddings of $apple$ and their neighbor words.

dimensionality of both word embeddings and topic embeddings as $K = 400$.

We use two similarity scores $AvgSimC$ and $MaxSimC$ following [Reisinger and Mooney, 2010; Liu *et al.*, 2015].

For each word w with its context c, we will first infer the topic distribution $Pr(t|w, c)$ by regarding c as a document. Given a pair of words with their contexts, namely (w_i, c_i) and (w_j, c_j), $AvgSimC$ aims to measure the averaged similarity between the two words under different topics:

$$AvgSimC = \sum_{t,t' \in T} Pr(t|w_i, c_i) Pr(t'|w_j, c_j) S(\mathbf{w'}_i, \mathbf{w}'_j), \tag{18}$$

where $\mathbf{w}'$ is the embedding of word w under its topic t, obtained by concatenating word and topic embeddings $\mathbf{w}' = \mathbf{w} \oplus \mathbf{t}$; $S(\mathbf{w'}_i, \mathbf{w}'_j)$ represents cosine similarity in this paper.

$MaxSimC$ selects the corresponding topical word embedding $\mathbf{w}'$ of the most probable topic t inferred using w in context c as the contextual word embedding. and the contextual word similarity is defined as

$$MaxSimC = S(\mathbf{w}_i^t, \mathbf{w}_j^{t'}), \tag{19}$$

where $t = \arg\max_t Pr(t|w_i, c_i)$ $t' = \arg\max_t Pr(t|w_j, c_j)$.

Finally, we show the evaluation results of various models in Table 2. Since we evaluate on the same data set as the other multi-prototype models [Huang *et al.*, 2012; Tian *et al.*, 2014; Neelakantan *et al.*, 2014; Liu *et al.*, 2015], we simply report the evaluation results from their papers. For the baseline Skip-Gram, we simply compute similarities using word embeddings ignoring context. Here the dimensionality of word embeddings in Skip-Gram is $K = 400$. C&W model is evaluated using word embeddings provided by [Collobert *et al.*, 2011], ignoring context information. The TFIDF methods represent words using context words within 10-word windows, weighted by TFIDF.

For all multi-prototype models and our models, we report the evaluation results using both $AvgSimC$ and $MaxSimC$. Table 2 shows the NTSG-2 model outperforms the other methods. The previous state-of-art model [Neelakantan *et al.*, 2014] on this task achieves 69.3% using the avgSimC measure, while the NTSG-2 achieves the best score of 69.5% on this task. The results on the other metrics are similar. By introducing topic embedding, the model can distinguish the different senses of each word more effectively. Moreover, the two model NTSG-1,2, which incorporates the interaction between words and topics, also get a better performance as compared to the [Liu *et al.*, 2015] model. As for the four NTSG models, we find that NTSG-2 outperforms the others. The reasons may be as follows: for NTSG-1, it models the interactions between words and topics using an inner product operation directly, which make the model less expressive; As for NTSG-3,4 the operation of tensor factorization degrades the performance while speeds up the training process, which just a trad-off between the performance and training speed.

4.3 Text Classification

We also investigate the effectiveness of our model for text classification. We use the popular dataset 20NewsGroup, which consists of about 20,000 documents from 20 different newsgroups. We report macro-averaging precision, recall and F1-measure for comparison.

For our model, we first learn topic models using LDA on the training and test set by setting the number of topics $T = 80$, which is the same as in [Liu *et al.*, 2015]. Then we learn word and topic embeddings on the training set with the dimensions of both word and topic embeddings $d = 400$. For each word and its topic in a document, we generate its contextual word embeddings by concatenating the word and topic embeddings. Further, a document q is also represented as a vector by averaging the contextual word embeddings of all words in the document, i.e.,$\mathbf{q} = \sum_{w \in q} Pr(w|q)\mathbf{w}^t$, where $\mathbf{w}^t = \mathbf{w} \oplus \mathbf{t}$ and $Pr(w|q)$ can be weighted with TFIDF scores of words in q. Afterwards, we regard document embedding vectors as document features and train a linear classifier using

Model	$\rho \times 100$	
TFIDF	26.3	
Pruned TFIDF	62.5	
Skip-Gram	65.7	
C&W	57.0	
	AvgSimC	**MaxSimC**
multi-prototypes		
[Huang *et al.*, 2012]	65.4	63.6
[Tian *et al.*, 2014]	65.3	58.6
[Neelakantan *et al.*, 2014]	69.3	-
[Liu *et al.*, 2015]	68.1	67.3
our models		
NTSG-1	68.2	67.3
NTSG-2	**69.5**	**67.9**
NTSG-3	68.5	67.2
NTSG-4	67.1	65.7

Table 2: Spearman correlation ρ 100 of contextual word similarity on the SCWS data set.

Model	Acc.	Prec.	Rec.	F1
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
multi-prototypes				
[Liu *et al.*, 2015]	81.5	81.2	80.6	80.6
our models				
NTSG-1	**82.6**	82.5	81.9	81.2
NTSG-2	82.5	**83.7**	**82.8**	**82.4**
NTSG-3	81.9	83.0	81.7	81.1
NTSG-4	79.8	80.7	78.8	78.8

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

Liblinear [Fan *et al.*, 2008].

We consider the following baselines, bag-of-words (BOW) model, LDA, Skip-Gram and TWE. 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. 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 $K = 400$.

Table 3 shows the evaluation results of text classification on 20NewsGroup. NTSG-1,2,3 outperform all baselines significantly, especially NTSG-2 achieves the best performance. For TWE model [Liu *et al.*, 2015], during the learning process, topic embeddings will influence the corresponding word embeddings, which may make those words in the same topic less discriminative, which was also mentioned by [Liu *et al.*, 2015]. Our model solves this problem in some degree. Similar to the previous task, NTSG-2 is superior to the other NTSG models. In future, we will conduct more experiments to explore the trad-off between model's performance and training speed.

5 Related Works

Recently, it has gained lots of interests to learn multi-prototype word embeddings. Reisinger and Mooney[2010] introduced a method for constructing multiple sparse, high-dimensional vector representations of words. Huang *et al.*[2012] extended this approach incorporating global document context to learn multiple dense, low-dimensional embeddings by using neural networks. Both the methods perform word sense discrimination as a preprocessing step by clustering contexts for each word type, making training more expensive. Tian *et al.*[2014] proposed to model word polysemy from a probabilistic perspective and integrate it with the highly efficient continuous Skip-Gram model. Neelakantan *et al.*[2014] porposed multiple-sense Skip-Gram to jointly perform word sense discrimination and embedding learning. Most of these models generate multi-prototype vectors with locally clustering the contexts for each word, which ignores complicated correlations among words as well as their contexts. To avoid this limitation, Liu *et al.*[2015] introduced latent topic model to globally cluster the words into different topics according to their contexts. They proposed three intuitive models to enhance the discriminativeness of word embeddings. However, their models do not model clearly the interactions among the word, topic and contexts. As mentioned in section 3.2, TWE can be regarded as a special case of our model.

6 Conclusion

We introduce a general architecture, Neural Tensor Skip-Gram model, to learn multi-prototype word embeddings. By combining the embeddings of the word and its topics under different contexts, we can obtain context-sensitive word embeddings per word type. Our model achieves better results than the other state-of-the-art multi-prototype models in the contextual word similarity task and the text classification task.

We consider the following future research directions: (1) There are some neural models for topic modelling, such as neural autoregressive topic model [Larochelle and Lauly, 2012]. We wish to integrate these ideas and design a united architecture to learn the latent topics jointly. (2) We would like to explore a more sophisticated combination of word and topic to get the context-sensitive word embeddings instead of the simple concatenation. (3) By learning word and topic embedding jointly, we find the words under the same topic can also be subdivided to several more specific topic, which provides us a clue that we can learn topic embedding hierarchically.

Acknowledgments

We would like to thank the anonymous reviewers for their valuable comments. This work was partially funded by National Natural Science Foundation of China (61472088, 61473092), The National High Technology Research and Development Program of China (2015AA015408), Shanghai Science and Technology Development Funds (13dz2260200, 13511504300, 14ZR1403200), Shanghai Leading Academic Discipline Project (B114).

References

[Bengio *et al.*, 2003] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *The Journal of Machine Learning Research*, 3:1137–1155, 2003.

[Blei *et al.*, 2003] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March 2003.

[Bordes *et al.*, 2013] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *NIPS*, 2013.

[Collobert and Weston, 2008] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of ICML*, 2008.

[Collobert *et al.*, 2011] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537, 2011.

[Fan *et al.*, 2008] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. Liblinear: A library for large linear classification. *The Journal of Machine Learning Research*, 9:1871–1874, 2008.

[Griffiths and Steyvers, 2004] Thomas L Griffiths and Mark Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101(suppl 1):5228–5235, 2004.

[Huang *et al.*, 2012] Eric Huang, Richard Socher, Christopher Manning, and Andrew Ng. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 873–882, Jeju Island, Korea, July 2012. Association for Computational Linguistics.

[Larochelle and Lauly, 2012] Hugo Larochelle and Stanislas Lauly. A neural autoregressive topic model. In F. Pereira, C.J.C. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 2708–2716. Curran Associates, Inc., 2012.

[Liu *et al.*, 2015] Yang Liu, Zhiyuan Liu, Tat-Seng Chua, and Maosong Sun. Topical word embeddings. In *AAAI*, 2015.

[Luong *et al.*, 2013] Minh-Thang Luong, Richard Socher, and C Manning. Better word representations with recursive neural networks for morphology. *CoNLL-2013*, 104, 2013.

[Mikolov *et al.*, 2013a] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[Mikolov *et al.*, 2013b] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*. 2013.

[Mnih and Hinton, 2007] Andriy Mnih and Geoffrey Hinton. Three new graphical models for statistical language modelling. In *Proceedings of ICML*, 2007.

[Neelakantan *et al.*, 2014] Arvind Neelakantan, Jeevan Shankar, Alexandre Passos, and Andrew McCallum. Efficient non-parametric estimation of multiple embeddings per word in vector space. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014.

[Reisinger and Mooney, 2010] Joseph Reisinger and Raymond J Mooney. Multi-prototype vector-space models of word meaning. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 109–117. Association for Computational Linguistics, 2010.

[Socher *et al.*, 2012] Richard Socher, Brody Huval, Christopher D Manning, and Andrew Y Ng. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1201–1211. Association for Computational Linguistics, 2012.

[Socher *et al.*, 2013a] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. Reasoning with neural tensor networks for knowledge base completion. In *Advances in Neural Information Processing Systems*, 2013.

[Socher *et al.*, 2013b] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*, 2013.

[Taskar *et al.*, 2005] Ben Taskar, Vassil Chatalbashev, Daphne Koller, and Carlos Guestrin. Learning structured prediction models: A large margin approach. In *Proceedings of the international conference on Machine learning*, 2005.

[Tian *et al.*, 2014] Fei Tian, Hanjun Dai, Jiang Bian, Bin Gao, Rui Zhang, Enhong Chen, and Tie-Yan Liu. A probabilistic model for learning multi-prototype word embeddings. In *Proceedings of COLING*, pages 151–160, 2014.

[Turian *et al.*, 2010] Joseph Turian, Lev Ratinov, and Yoshua Bengio. Word representations: a simple and general method for semi-supervised learning. In *ACL*, 2010.