CSE: Conceptual Sentence Embeddings based on Attention Model

Yashen Wang, Heyan Huang[*], Chong Feng, Qiang Zhou, Jiahui Gu and Xiong Gao
Beijing Engineering Research Center of High Volume Language Information Processing
and Cloud Computing Applications, School of Computer,
Beijing Institute of Technology, Beijing, P. R. China
{yswang,hhy63,fengchong,qzhou,gujh,gaoxiong}@bit.edu.cn

Abstract

Most sentence embedding models typically represent each sentence only using word surface, which makes these models indiscriminative for ubiquitous homonymy and polysemy. In order to enhance representation capability of sentence, we employ conceptualization model to assign associated concepts for each sentence in the text corpus, and then learn conceptual sentence embedding (CSE). Hence, this semantic representation is more expressive than some widely-used text representation models such as latent topic model, especially for short-text. Moreover, we further extend CSE models by utilizing a local attention-based model that select relevant words within the context to make more efficient prediction. In the experiments, we evaluate the CSE models on two tasks, text classification and information retrieval. The experimental results show that the proposed models outperform typical sentence embed-ding models.

1 Introduction

Many natural language processing applications require the input text to be represented as a fixed-length feature, of which sentence representation is very important. Perhaps the most common fixed-length vector representation for texts is the bag-of-words or bag-of-n-grams (Harris, 1970). However, they suffer severely from data sparsity and high dimensionality, and have very little sense about the semantics of words or the distances between the words. Recently, in sentence representation and classification, deep neural network (DNN) approaches have achieved state-of-the-art results (Le and Mikolov, 2014; Liu et al., 2015; Palangi et al., 2015; Wieting et al., 2015). Despite of their usefulness, recent sentence embeddings face several challenges: (i) Most sentence embedding models represent each sentence only using word surface, which makes these models indiscriminative for ubiquitous polysemy; (ii) For short-text, however, neither parsing nor topic modeling works well because there are simply not enough signals in the input; (iii) Setting window size of context words is very difficult. To solve these problems, we must derive more semantic signals from the input sentence, e.g., concepts. Besides, we should assigned different attention for different contextual word, to enhance the influence of words that are relevant for each prediction.

This paper proposed Conceptual Sentence Embedding (CSE), an unsupervised framework that learns continuous distributed vector representations for sentence. Specially, by innovatively introducing concept information, this concept-level vector representations of sentence are learned to predict the surrounding words or target word in contexts. Our research is inspired by the recent work in learning vector representations of words using deep learning strategy (Mikolov et al., 2013a; Le and Mikolov, 2014). More precisely, we first obtain concept distribution of the sentence, and generate corresponding concept vector. Then we concatenate or average the sentence vector, contextual word vectors with concept vector of the sentence, and predict the target word in the given context. All of the sentence vectors and word vectors are trained by the stochastic gradient descent and backpropagation (Rumelhart et al., 1986). At prediction time, sentence vectors are inferred by fixing the word vectors and observed sentence vectors, and training the new sentence vector until convergence.

In parallel, the concept of attention has gained

[*] The contact author.

popularity recently in neural natural language processing researches, which allowing models to learn alignments between different modalities (Bahdanau et al., 2014; Bansal et al., 2014; Rush et al., 2015). In this work, we further propose the extensions to CSE, which adds an attention model that considers contextual words differently depending on the word type and its relative position to the predicted word. The main intuition behind the extended model is that prediction of a word is mainly dependent on certain words surrounding it.

In summary, the basic idea of CSE is that, we allow each word to have different embeddings under different concepts. Taking word apple into consideration, it may indicate a fruit under the concept *food*, and indicate an IT company under the concept *information technology*. Hence, concept information significantly contributes to the discriminative of sentence vector. Moreover, an important advantage of the proposed conceptual sentence embeddings is that they could be learned from unlabeled data. Another advantage is that we take the word order into account, in the same way of n-gram model, while bag-of-n-grams model would create a very high-dimensional representation that tends to generalize poorly.

To summarize, this work contributes on the following aspects: We integrate concepts and attention-based strategy into basic sentence embedding representation, and allow the resulting conceptual sentence embedding to model different meanings of a word under different concept. The experimental results on text classification task and information retrieval task demonstrate that this concept-level sentence representation is robust. The outline of the paper is as follows. Section 2 surveys related researches. Section 3 formally de-scribes the proposed model of conceptual sentence embedding. Corresponding experimental results are shown in Section 4. Finally, we conclude the paper.

2 Related Works

Conventionally, one-hot sentence representation has been widely used as the basis of bag-of-words (BOW) text model. However, it can-not take the semantic information into consideration. Recently, in sentence representation and classification, deep neural network approaches have achieved state-of-the-art results (Le and Mikolov, 2014; Liu et al., 2015; Ma et al., 2015; Palangi et al., 2015; Wieting et al., 2015), most of which are inspired by word embedding (Mikolov et al., 2013a). (Le and Mikolov, 2014) proposed the paragraph vector (PV) that learns fixed-length representations from variable-length pieces of texts. Their model represents each document by a dense vector which is trained to predict words in the document. However, their model depends only on word surface, ignoring semantic information such as topics or concepts. In this paper, we extent PV by introducing concept information.

Aiming at enhancing discriminativeness for ubiquitous polysemy, (Liu et al., 2015) employed latent topic models to assign topics for each word in the text corpus, and learn topical word embeddings (TWE) and sentence embeddings based on both words and their topics. Besides, to combine deep learning with linguistic structures, many syntax-based embedding algorithms have been proposed (Severyn et al., 2014; Wang et al., 2015b) to utilize long-distance dependencies. However, short-texts usually do not observe the syntax of a written language, nor do they contain enough signals for statistical inference (e.g., topic model). Therefore, neither parsing nor topic modeling works well because there are simply not enough signals in the input, and we must derive more semantic signals from the input, e.g., concepts, which have been demonstrated effective in knowledge representation (Wang et al., 2015c; Song et al., 2015). Shot-text conceptualization, is an interesting task to infer the most likely concepts for terms in the short-text, which could help better make sense of text data, and extend the texts with categorical or topical information (Song et al., 2011). Therefore, our models utilize short-text conceptualization algorithm to discriminate concept-level sentence senses and provide a good performance on short-texts.

Recently, attention model has been used to improve many neural natural language pro-cessing researches by selectively focusing on parts of the source data (Bahdanau et al., 2014; Bansal et al., 2014; Wang et al., 2015a). To the best of our knowledge, there has not been any other work exploring the use of attentional mechanism for sentence embeddings.

3 Conceptual Sentence Embedding

This paper proposes four conceptual sentence embedding models. The first one is based on continuous bag-of-word model (denoted as **CSE-1**) which have not taken word order into consideration. To overcome this drawback, its extension model (denoted as **CSE-2**), which is based on Skip-Gram model, is proposed. Based on the basic conceptual sentence embedding models above, we obtain their variants (**aCSE-1** and **aCSE-2**) by introducing attention model.

3.1 CBOW Model & Skip-Gram Model

As inspiration of the proposed conceptual sentence embedding models, we start by discussing previous models for learning word vectors (Mikolov et al., 2013a; Mikolov et al., 2013b) firstly.

Let us overview the framework of Continuous Bag-of-Words (CBOW) firstly, which is shown in Figure 1(a). Each word is typically mapped to an unique vector, represented by a column in a word matrix $\mathbf{W} \in \Re^{d*|V|}$. Wherein, V denotes the word vocabulary and d is embedding dimension of word. The column is indexed by position of the word in V. The concatenation or average of the vectors, the context vector w_t, is then used as features for predicting the target word in the current context. Formally, Given a sentence $S = \{w_1, w_2, \ldots, w_l\}$, the objective of CBOW is to maximize the average log probability:

$$\mathcal{L}(S)=\tfrac{1}{(l-2k-2)}\sum_{t=k+1}^{l-k}\log Pr(w_t|w_{t-k},\cdots,w_{t+k}) \quad (1)$$

Wherein, k is the context windows size of target word w_t. The prediction task is typically done via a softmax function, as follows:

$$Pr(w_t|w_{t-k},\cdots,w_{t+k}) = \frac{e^{y_{w_t}}}{\sum_{w_i \in V} e^{y_{w_i}}} \quad (2)$$

Each of $y_(w_t)$ is an un-normalized log-probability for each target word w_t, as follows:

$$y_{w_t} = \mathbf{U}h(w_{t-k},\ldots,w_{t+k}); \mathbf{W}) + b \quad (3)$$

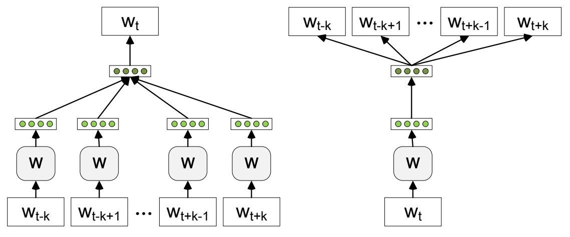
Wherein, $\mathbf{U}$ and b are softmax parameters. And $h(\cdot)$ is constructed by a concatenation or average of word vectors $\{\mathbf{w}_{t-k},\ldots,\mathbf{w}_{t+k}\}$ extracted from word matrix $\mathbf{W}$ according to $\{w_{t-k},\ldots,w_{t+k}\}$. For illustration purposes, we utilize average here. On the condition of average, the context vector c_t is obtained by averaging the embeddings of each word, as follows:

$$\mathbf{c}_t = \frac{1}{2k}\sum_{-k\le c\le k, c\ne 0}\mathbf{w}_{t+c} \quad (4)$$

The framework of Skip-Gram (Figure 1(b)) aims to predict context words given a target word w_t in a sliding window, instead of predicting the current word based on its context. Formally, given a sentence $S = \{w_1, w_2, \ldots, w_l\}$, the objective of Skip-Gram is to maximize the following average log probability:

$$\mathcal{L}(S)=\tfrac{1}{(l-2k)}\sum_{t=k+1}^{l-k}\sum_{-k\le c\le k,c\ne 0}\log Pr(w_{t+c}|w_t) \quad (5)$$

Wherein, $\mathbf{w}_t$ and $\mathbf{w}_c$ are respectively the vector representations of the target word w_t and the context word w_c. Usually, during the training stage of CBOW and Skip-Gram: (i) in order to make the models efficient for learning, the techniques of hierarchical softmax and negative sampling are used to ensure the models efficient for learning (Morin and Bengio, 2005; Mikolov et al., 2013a); (ii) the word vectors are trained by using stochastic gradient descent where the gradient is obtained via backpropagation (Rumelhart et al., 1986). After the training stage converges, words with similar meaning are mapped to a similar position in the semantic vector space. e.g., 'powerful' and 'strong' are close to each other.



(a) (b)

Figure 1: (a) CBOW model and (b) Skip-Gram model.

3.2 CSE based on CBOW Model

Intuitively, the proposed (attention-based) conceptual sentence embedding model for learning sentence vectors, is inspired by the methods for learning the word vectors. The inspiration is that, in researches of word embeddings: (i) The word vectors are asked to contribute to a prediction task about the target word or the surrounding words in the context; (ii) The word representation vectors are initialized randomly, however they could finally capture precise semantics as an indirect result. Therefore, we will utilize this idea in our sentence vectors in a similar manner: The concept-associated sentence vectors are also asked to contribute to the prediction task of the target word or surrounding words in given contextual text windows. Furthermore, attention model will attribute different influence value to different contextual words.

We describe the first conceptual sentence embedding model, denoted as **CSE-1**, which is based on CBOW. In the framework of **CSE-1** (Figure 2(a)), each sentence, denoted by sentence ID, is mapped to a unique vector **s**, represented by a column in matrix **S**. And its concept distribution θ_C are generated from a knowledge-based text conceptualization algorithm (Wang et al., 2015c). Moreover, similar to word embedding methods, each word w_i is also mapped to a unique vector $\mathbf{w}_i$, represented by a column in matrix W. The surrounding words in contextual text window $\{w_{t-k}, \ldots, w_{t+k}\}$, sentence ID and concept distribution θ_C corresponding to this sentence are the inputs. Besides, **C** is a fixed linear operator similar to the one used in (Huang et al., 2013) that converts the concept distribution θ_C to a concept vector, denoted as **c**. Note that, this makes our model very different from (Le and Mikolov, 2014) where no concept information is used, and experimental results demonstrate the efficiency of introducing concept information. It is clear that **CSE-1** also does not take word order into consideration just like CBOW.

Afterward, the sentence vector **s**, surrounding word vectors $\{\mathbf{w}_{t-k}, \ldots, \mathbf{w}_{t+k}\}$ and the concept vector **c** are concatenated or averaged to predict the target word w_t in current context. In reality, the only change in this model compared to the word embedding method is in Eq. 3, where $h(\cdot)$ is constructed from not only **W** but also **C** and **S**. Note that, the sentence vector is shared across all contexts generated from the same sentence but not across sentences. Wherein, the contexts are fixed-length (length is $2k$) and sampled from a sliding window over the current sentence. However, the word matrix **W** is shared across sentences.

In summary, the procedure of **CSE-1** itself is described as follows. A probabilistic conceptualization algorithm (Wang et al., 2015c) is employed here to obtain the corresponding concepts about given sentence: Firstly, we preprosess and segment the given sentence into a set of words; Then, based on a probabilistic lexical knowledge-base Probase (Wu et al., 2012), the heterogeneous semantic graph for these words and their corresponding concepts are constructed (Figure 3 shows an example); Finally, we utilize a simple iterative process to identify the most likely mapping from words to concepts. After efforts above, we could conceptualize words in given sentence, and access the concepts and corresponding probabilities, which is the concept distribution θ_C mentioned before. Note that, the concept distribution yields an important influence on the entire framework of conceptual sentence embedding, by contributing greatly to the semantic representation.

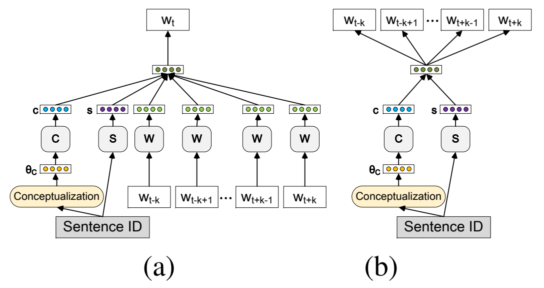


Figure 2: **CSE-1** model (a) and **CSE-2** model (b). Green circles indicate word embeddings, blue circles indicate concept embeddings, and purple circles indicate sentence embeddings. Besides, orange circles indicate concept distribution θ_C generated by knowledge-based text conceptualization algorithm.

During the training stage, we aim at obtaining word matrix **W**, sentence matrix **S**, and softmax weights $\{\mathbf{U}, b\}$ on already observed sentences. The techniques of hierarchical softmax and negative sampling are used to make the model efficient for learning. **W** and **S** are trained using stochastic gradient descent: At each step of stochastic gradient descent, we sample a fixed-length context from the given sentence, compute the error gradient which is obtained via backpropagation, and then use the gradient to update the parameters. During the inferring stage, we get sentence vectors for new sentences (unobserved before) by adding more columns in **S** and gradient descending on **S** while holding **W**, **U** and b fixed. Finally, we use **S** to make a prediction about multi-labels by using a standard classifier in output layer.

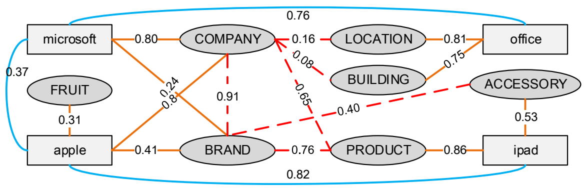


Figure 3: Semantic graph of example sentence microsoft unveils office for apples ipad. Rectangles indicate terms occurred in given sentence, and ellipses indicate concept defined in knowledge-base (e.g., Probase). Bule solid links indicate *isA* relationship between terms and concepts, and red dashed lines indicate correlation relationship between two concepts. Numerical values on the line is corresponding probabilities.

3.3 CSE based on Skip-Gram Model

The above method considers the combination of the sentence vector with the surrounding word vectors and concept vector to predict the target word in given text window. However, it loss information about word order somehow, just like CBOW. In fact, there exists another for modeling the prediction procedure: we could ignore the context words in the input, but force the model to predict words randomly sampled from the fix-length contexts in the output. As is shown in Figure 2 (b), only sentence vector **s** and concept vector **c** are used to predict the next word in a text window. That means, contextual words are no longer used as inputs, whereas they become what the output layer predict. Hence, this model is similar to the Skip-Gram model in word embedding (Mikolov et al., 2013b). In reality, what this means is that at each iteration of stochastic gradient descent, we sample a text window $\{w_{t-k}, \ldots, w_{t+k}\}$, then sample a random word from this text window and form a classification task given the sentence vector **s** and corresponding concept vector **c**.

We denote this sort of conceptual sentence embedding model as **CSE-2**. The scheme of **CSE-2** is similar to that of **CSE-1** as described above. In addition to being conceptually simple, **CSE-2** requires to store less data. We only need to store $\{\mathbf{U}, b, \mathbf{S}\}$ as opposed to $\{\mathbf{U}, b, \mathbf{S}, \mathbf{W}\}$ in **CSE-1**.

3.4 CSE based on Attention Model

As mentioned above, setting a good value for contextual window size k is difficult. Because a larger value of k may introduce a degenerative behavior in the model, and more effort is spent predicting words that are conditioned on unrelated words, while a smaller value of k may lead to cases where the window size is not large enough include words that are semantically related (Bansal et al., 2014; Wang et al., 2015a). To solve these problems , we extend the proposed models by introducing attention model (Bahdanau et al., 2014; Rush et al., 2015), by allowing it to consider contextual words within the window in a non-uniform way. For illustration purposes, we extend **CSE-1** here with attention model. Following (Wang et al., 2015a), we rewrite Eq.(4) as follows:

$$\mathbf{c}_t = \frac{1}{2k} \sum_{-k \le c \le k, c \ne 0} a_{t+c}(w_{t+c}) \mathbf{w}_{t+c} \qquad (6)$$

Wherein we replace the average of the surrounding word vectors in Eq.(4) with a weighted sum of the these vectors. That means, each contextual word w_{t+c} is attributed a different attention level, representing how much the attention model believes whether it is important to look at in order to predict the target word w_t. The attention factor $a_i(w_i)$ for word w_i in position i is formulated as a softmax function over contextual words (Bahdanau et al., 2014), as follows:

$$a_i(w) = \frac{e^{d_{w,i}} + r_i}{\sum_{-k \le c \le k, c \ne 0} e^{d_{w,c}} + r_c} \qquad (7)$$

Wherein, $d_{w,i}$ is an element of matrix $\mathbf{D} \in \Re^{|V|*2k}$, which is a set of parameters determining the importance of each word type in each relative position i (distance to the left/right of target word w_t). Moreover, r_i, an element of $\mathbf{R} \in \Re^{2k}$, is a bias, which is conditioned only on the relative position i. Note that, attention models have been reported expensive for large tables in terms of storage and performance (Bahdanau et al., 2014; Wang et al., 2015a). Nevertheless the computation consumption here is simple, and compute the attention of all words in the input requires $2k$ operations, as it simply requires retrieving on value from the lookup-matrix **D** for each word and one value from the bias vector **R** for each word in the context. Although this strategy may not be the best approach and there exist more elaborate attention models (Bahdanau et al., 2014; Luong et al., 2015), the proposed attention model is a proper balance of computational efficiency and complexity.

Thus, besides $\{\mathbf{W}, \mathbf{C}, \mathbf{S}\}$ in CSE models, **D** and **R** are added into parameter set which relates to gradients of the loss function Eq.(1). All parameters are computed with backpropagation and updated after each training instance using a fixed learning rate. We denote the attention-based **CSE-1** model above as **aCSE-1**. With limitation of space, attention variant of **CSE-2**, denoted as **aCSE-2**, is not described here, however the principle is similar to **aCSE-1**.

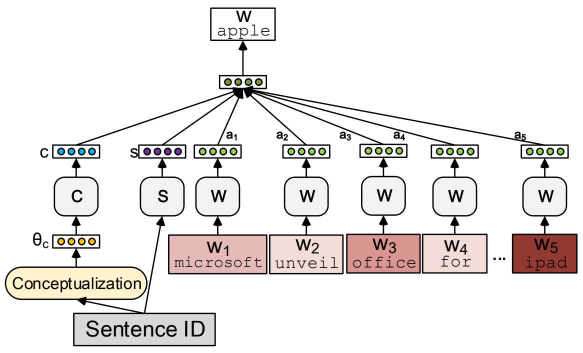


Figure 4: **aCSE-1** model. The illustration of example sentence 'mcrosoft unveils office for apple's ipad' for predicting word 'apple'.

Taking example 'microsoft unveils office for apple's ipad' into consideration. The prediction of the polysemy word 'apple' by **CSE-1** is shown in Figure 4, and darker cycle cell indicate higher attention value. We could observe that preposition word 'for' tend to be attributed very low attention, while context words, especially noun-words which contribute much to conceptualization (such as 'ipad', 'office', and 'microsoft') are attributed higher weights as these word own more predictive power. Wherein, 'ipad' is assigned the highest attention value as it close to the predicted word and co-occurs with it more frequently.

As described before, concept distribution θ_C yields a considerable influence on conceptual sentence embedding. This is because, each dimensionality of this distribution denotes the probability of the concept (topic or category) this sentence is respect to. In other words, the concept distribution is a solid semantic representation of the sentence. Nevertheless, the information in each dimensionality of sentence (or word) vector makes no sense. Hence, there exist a linear operator in **CSE-1**, **CSE-2**, **aCSE-1**, and **aCSE-2**, which transmit the concept distribution into word vector and sentence vector, as shown in Figure 2 and Figure 3.

4 Experiments and Results

In this section, we show experiments on two text understanding problems, text classification and information retrieval, to evaluate related models in several aspects. These tasks are always used to evaluate the performance of sentence embedding methods (Liu et al., 2015; Le and Mikolov, 2014). The source codes and datasets of this paper are publicly available[1].

4.1 Datasets

We utilize four datasets for training and evaluating. For text classification task, we use three datasets: **NewsTile**, **TREC** and **Twitter**. Dataset **Tweet11** is used for evaluation in information retrieval task. Moreover, we construct dataset **Wiki** to fully train topic model-based models.

NewsTitle: The news articles are extracted from a large news corpus, which contains about one million articles searched from Web pages. We organize volunteers to classify these news articles manually into topics according its article content (Song et al., 2015), and we select six topics: company, health, entertainment, food, politician, and sports. We randomly select 3,000 news articles in each topic, and only keep its title and its first one line of article. The average word count of titles is 9.41.

TREC: It is the corpus for question classification on TREC (Li and Roth, 2002), which is widely used as benchmark in text classification task. There are 5,952 sentences in the entire dataset, classified into the 6 categories as follows: person, abbreviation, entity, description, location and numeric.

Tweet11: This is the official tweet collections used in TREC Microblog Task 2011 and 2012 (Ounis et al., 2011; Soboroff et al., 2012). Using the official API, we crawled a set of local copies of the corpus. Our local Tweets11 collection has a sample of about 16 million tweets, and a set of 49 (TMB2011) and 60 (TMB2012) timestamped topics.

Twitter: This dataset is constructed by manually labeling the previous dataset Tweet11. Similar to dataset NewsTitle, we ask our volunteers to label these tweets. After manually labeling, the dataset contains 12,456 tweets which are in four

[1]`http://hlipca.org/index.php/2014-12-09-02-55-58/2014-12-09-02-56-24/58-acse`

categories: company, country, entertainment, and device. The average length of the tweets is 13.16 words. Because of its noise and sparsity, this social media dataset is very challenging for the comparative models.

Moreover, we also construct a Wikipedia dataset (denoted as **Wiki**) for training. We preprocess the Wikipedia articles[2] with the following rules. First, we remove the articles less than 100 words, as well as the articles less than 10 links. Then we remove all the category pages and disambiguation pages. Moreover, we move the content to the right redirection pages. Finally we obtain about 3.74 million Wikipedia articles for indexing and training.

4.2 Alternative Algorithms

We compare the proposed models with the following comparative algorithms.

BOW: It is a simple baseline which represents each sentence as bag-of-words, and uses TF-IDF scores (Salton and Mcgill, 1986) as features to generate sentence vector.

LDA: It represents each sentence as its topic distribution inferred by latent dirichlet allocation (Blei et al., 2003). We train this model in two ways: (i) on both Wikipedia articles and the evaluation datasets above, and (ii) only on the evaluation datasets. We report the better of the two.

PV: Paragraph Vector models are variable-length text embedding models, including the distributed memory model (**PV-DM**) and the distributed bag-of-words model (**PV-DBOW**). It has been reported to achieve the state-of-the-art performance on task of sentiment classification (Le and Mikolov, 2014), however it only utilizes word surface.

TWE: By taking advantage of topic model, it overcome ambiguity to some extent (Liu et al., 2015). Typically, TWE learn topic models on training set. It further learn topical word embeddings using the training set, then generate sentence embeddings for both training set and testing set. (Liu et al., 2015) proposed three models for topical word embedding, and we present the best results here. Besides, We also train TWE in two ways like **LDA**.

[2]`http://en.wikipedia.org/wiki/Wikipedia:Databasedown-load`

4.3 Experiment Setup

The details about parameter settings of the comparative algorithms are described in this section, respectively. For **TWE**, **CSE-1**, **CSE-2** and their attention variants **aCSE-1**, and **aCSE-2**, the structure of the hierarchical softmax is a binary Huffman tree (Mikolov et al., 2013a; Mikolov et al., 2013b), where short codes are assigned to frequent words. This is a good speedup trick because common words are accessed quickly (Le and Mikolov, 2014).We set the dimensions of sentence, word, topic and concept embeddings as 5,000, which is like the number of concept clusters in Probase (Wu et al., 2012; Wang et al., 2015c). Meanwhile, we have done many experiments on choosing the context window size (k). We perform experiments on increasing windows size from 3 to 11, and different size works differently on different dataset with different average length of short-texts. And we choose the result of windows size of 5 present here, because it performs best in almost datasets. Usually, in project layer, the sentence vector, the context vector and the concept vectors could be averaged or concatenated for combination to predict the next word in a context. We perform experiments following these two strategies respectively, and report the better of the two. In fact, the concatenation performs better since averaging different types of vectors may cause loss of information somehow.

For **BOW** and **LDA**, we remove stop words by using InQuery stop-word list. For **BOW**, we select top 50,000 words according to TF-IDF scores as features. For both **LDA** and **TWE**, in the text classification task, we set the topic number to be the cluster number or twice, and report the better of the two; while in the information retrieval task, we experimented with a varying number of topics from 100 to 500, which gives similar performance, and we report the final results of using 500 topics.

In summary, we use the sentence vectors generated by each algorithm as features and run a linear classifier using Liblinear (Fan et al., 2010) for evaluation.

4.4 Text Classification

In this section, we run the multi-class text classification experiments on the dataset **NewsTitle**, **Twitter**, and **TREC**. We report precision, recall and F-measure for comparison (as shown in Table 1). Statistical t-test are employed here. To de-

	NewsTitle			Twitter			TREC		
Model	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
BOW	0.782	0.791	0.786	0.437	0.429	0.433	0.892	0.891	0.891
LDA	0.717	0.705	0.711	0.342	0.308	0.324	0.813	0.809	0.811
PV-DBOW	0.725	0.719	0.722	0.413	0.408	0.410	0.824	0.819	0.821
PV-DM	0.748	0.740	0.744	0.426	0.424	0.425	0.836	0.825	0.830
TWE	0.811^{β}	0.803^{β}	0.807^{β}	0.459^{β}	0.438	0.448^{β}	0.898^{β}	0.886^{β}	0.892^{β}
CSE-1	0.815	0.809	0.812	0.461	**0.449**	0.454	0.896	0.890	0.893
CSE-2	**0.827**	**0.817**	**0.822**	**0.475**	0.447	**0.462**	**0.901**	**0.895**	**0.898**
aCSE-1	0.824	0.818	0.821	0.471	**0.454**	0.462	0.901	0.897	0.899
aCSE-2	$\mathbf{0.831}^{\alpha\beta}$	$\mathbf{0.820}^{\alpha\beta}$	$\mathbf{0.825}^{\alpha\beta}$	$\mathbf{0.477}^{\alpha\beta}$	$0.450^{\alpha\beta}$	$\mathbf{0.463}^{\alpha\beta}$	$\mathbf{0.909}^{\alpha\beta}$	$\mathbf{0.904}^{\alpha\beta}$	$\mathbf{0.906}^{\alpha\beta}$

Table 1: Evaluation results of multi-class text classification task.

cide whether the improvement by method A over method B is significant, the t-test calculates a value p based on the performance of A and B. The smaller p is, the more significant is the improvement. If the p is small enough ($p < 0.05$), we conclude that the improvement is statistically significant. In Table 1, the superscript α and β respectively denote statistically significant improvements over **TWE** and **PV-DM**.

Without regard to attention-based model firstly, we could conclude that **CSE-2** outperforms all the baselines significantly (expect for recall in **Twitter**). This fully indicates that the proposed model could capture more precise semantic information of sentence as compared to topic model-based models and other embedding models. Because the concepts we obtained contribute significantly to the semantic representation of sentence, meanwhile suffer slightly from texts noisy and sparsity. Moreover, as compared to **BOW**, **CSE-1** and **CSE-2** manage to reduce the feature space by 90 percent, while among them, **CSE-2** needs to store less data comparing with **CSE-1**. By introducing attention model, performances of CSE models are entirely promoted, as compared **aCSE-2** with original **CSE-2**, which demonstrates the advantage of attention model.

PV-DM and **PV-DBOW** are reported as the state-of-the-art model for sentence embedding. From the results we can also see that, the proposed model **CSE-2** and **aCSE-2** significantly outperforms **PV-DBOW**. As expected, **LDA** performs worst, even worse than **BOW**, because it is trained on very sparse short-texts (i.e., question and social media text), where there is no enough statistical information to infer word co-occurrence and word topics, and latent topic model suffer extremely from the sparsity of the short-text. Besides, the number of topics slightly impacts the performance of **LDA**. In future, we may conduct more experiments to explore genuine reasons. As described in section 3, **aCSE-2** (**CSE-2**) performs better than **aCSE-1** (**CSE-1**), because the former one take word order into consideration. Based on Skip-Gram similarly, **CSE-2** outperforms **TWE**. Although **TWE** aims at enhancing sentence representation by using topic model, neither parsing nor topic modeling would work well because short-texts lack enough signals for inference. Whats more, sentence embeddings are generated by simple aggregating over all topical word embeddings of each word in this sentence in **TWE**, which limits its capability of semantic representation.

Overall, nearly all the alternative algorithms perform worse on **Twitter**, especially **LDA** and **TWE**. This is mainly because that data in **Twitter** are more challenging for topic model as short-texts are noisy, sparse, and ambiguous. Although the training on larger corpus, i.e., way (i), contributes greatly to improving the performance of these topic-model based algorithms, they only have similar performance to **CSE-1** and could not transcend the attention-based variants. Certainly, we could also train **TWE** (even **LDA**) on a very larger corpus, and could expect a letter better results. However, training latent topic model on very large dataset is very slow, although many fast algorithms of topic models are available (Smola and Narayanamurthy, 2010; Ahmed et al., 2012). Whats more, from the complexity analysis, we could conclude that, compared with **PV**, **CSE** only need a little more space to store look-ups matrix **D** and **R**; while compared with **CSE** and **PV**, **TWE** require more parameters to store more discriminative information for word embedding.

4.5 Information Retrieval

The information retrieval task is also utilized to evaluate the proposed models, and we want to examine whether a sentence should be retrieved given a query. Specially, we mainly focus on short-text retrieval by utilizing official tweet collection **Tweet11**, which is the benchmark dataset for microblog retrieval. We index all tweets in this collection by using Indri toolkit, and then perform a general relevance-pseudo feedback procedure, as follows: (i) Given a query, we firstly obtain associated tweets, which are before query issue time, via preliminary retrieval as feedback tweets. (ii) We generate the sentence representation vector of both original query and these feedback tweets by the alternative algorithms above. (iii) With efforts above, we compute cosine scores between query vector and each tweet vector to measure the semantic similarity between the query and candidate tweets, and then re-rank the feedback tweets with descending cosine scores.

We utilize the official metric for the TREC Microblog track, i.e., Precision at 30 (P@30), and Mean Average Precision (MAP), for evaluating the ranking performance of different algorithms. Experimental results for this task are shown in Table 2. Besides, we also operate a query-by-query analysis and conduct t-test to demonstrate the improvements on both metrics are statistically significant. In Table 2, the superscript α and β respectively denote statistically significant improvements over **TWE** and **PV-DM** ($p < 0.05$).

As shown in Table 2, the **CSE-2** significantly outperforms all these models, and exceeds the best baseline model (**TWE**) by 11.9% in MAP and 4.5% in P@30, which is a statistically significant improvement. Without regard to attention-based model firstly, such an improvement comes from the **CSE-2**'s ability to embed the contextual and semantic information of the sentences into a finite dimension vector. Topic model based algorithms (e.g., **LDA** and **TWE**) suffer extremely from the sparsity and noise of tweet collection. For the twitter data, since we are not able to find appropriate long texts, latent topic models are not performed.

	TMB2011		**TMB2012**	
Model	**MAP**	**P@30**	**MAP**	**P@30**
BOW	0.304	0.412	0.321	0.494
LDA	0.281	0.409	0.311	0.486
PV-DBOW	0.285	0.412	0.324	0.491
PV-DM	0.327	0.431	0.340	0.524
TWE	0.331	0.446^{β}	0.347^{β}	0.511
CSE-1	0.337	0.451	0.344	0.512
CSE-2	**0.367**	**0.461**	**0.360**	**0.517**
aCSE-1	0.342	0.459	0.351	0.516
aCSE-2	**0.370**$^{\alpha\beta}$	**0.464**$^{\alpha\beta}$	**0.364**$^{\alpha\beta}$	**0.522**$^{\alpha\beta}$

Table 2: Results of information retrieval.

We could observe that attention-based CSE model (**aCSE-1** and **aCSE-2**) improves over original CSE model (**CSE-1** and **CSE-2**). However, attention model promotes **CSE-1** significantly, while **aCSE-2** obtain similar results compared to **CSE-2**, indicating that attention model leads to small improvement for Skip-Gram based CSE model. We argue that it is because Skip-Gram itself gives less weight to the distant words by sampling less from those words, which is essentially similar to attention model somehow.

5 Conclusion

By inducing concept information, the proposed conceptual sentence embedding maintains and enhances the semantic information of sentence embedding. Furthermore, we extend the proposed models by introducing attention model, which allows it to consider contextual words within the window in a non-uniform way while maintaining the efficiency. We compare them with different algorithms, including bag-of-word models, topic model-based model and other state-of-the-art sentence embedding models. The experimental results demonstrate that the proposed method performs the best and shows improvement over the compared methods, especially for short-texts.

Acknowledgments

The work was supported by National Natural Science Foundation of China (Grant No. 61132009), National Basic Research Program of China (973 Program, Grant No. 2013CB329303), and National Hi-Tech Research & Development Program (863 Pro-gram, Grant No. 2015AA015404).

References

Amr Ahmed, Moahmed Aly, Joseph Gonzalez, Shravan Narayanamurthy, and Alexander J. Smola. 2012. Scalable inference in latent variable models. In *International Conference on Web Search and Web Data Mining, WSDM 2012, Seattle, Wa, Usa, February*, pages 123–132.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *Eprint Arxiv*.

Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2014. Tailoring continuous word representations for dependency parsing. In *Meeting of the Association for Computational Linguistics*, pages 809–815.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Rongen Fan, Kaiwei Chang, Cho Jui Hsieh, Xiangrui Wang, and Chih Jen Lin. 2010. Liblinear: A library for large linear classification. *Journal of Machine Learning Research*, 9(12):1871–1874.

Zellig S. Harris. 1970. *Distributional Structure*. Springer Netherlands.

Posen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *ACM International Conference on Conference on Information and Knowledge Management*, pages 2333–2338.

Quoc V. Le and Tomas. Mikolov. 2014. Distributed representations of sentences and documents. *Eprint Arxiv*, 4:1188–1196.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING*.

Yang Liu, Zhiyuan Liu, Tat-Seng Chua, and Maosong Sun. 2015. Topical word embeddings. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *EMNLP*.

Mingbo Ma, Liang Huang, Bing Xiang, and Bowen Zhou. 2015. Dependency-based convolutional neural networks for sentence embedding. In *Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *Computer Science*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems*, 26:3111–3119.

Frederic Morin and Yoshua Bengio. 2005. Hierarchical probabilistic neural network language model. *Aistats*.

Iadh Ounis, Craig MacDonald, Jimmy Lin, and Ian Soboroff. 2011. Overview of the trec-2011 microblog track.

H Palangi, L Deng, Y Shen, J Gao, X He, J Chen, X Song, and R Ward. 2015. Deep sentence embedding using the long short term memory network: Analysis and application to information retrieval. *Arxiv*, 24(4):694–707.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1986. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*.

Gerard Salton and Michael J. Mcgill. 1986. *Introduction to modern information retrieval*. McGraw-Hill,.

Aliaksei Severyn, Alessandro Moschitti, Manos Tsagkias, Richard Berendsen, and Maarten De Rijke. 2014. A syntax-aware re-ranker for microblog retrieval. In *SIGIR*, pages 1067–1070.

Alexander Smola and Shravan Narayanamurthy. 2010. An architecture for parallel topic models. *Proceedings of the Vldb Endowment*, 3(1):703–710.

Ian Soboroff, Iadh Ounis, Craig MacDonald, and Jimmy Lin. 2012. Overview of the trec-2012 microblog track. In *TREC*.

Yangqiu Song, Haixun Wang, Zhongyuan Wang, Hongsong Li, and Weizhu Chen. 2011. Short text conceptualization using a probabilistic knowledgebase. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence - Volume Volume Three*, pages 2330–2336.

Yangqiu Song, Shusen Wang, and Haixun Wang. 2015. Open domain short text conceptualization: a generative + descriptive modeling approach. In *Proceedings of the 24th International Conference on Artificial Intelligence*.

Ling Wang, Tsvetkov Yulia, Amir Silvio, Fermandez Ramon, Dyer Chris, Black Alan W, Trancoso Isabel, and Lin Chu-Cheng. 2015a. Not all contexts are created equal: Better word representations with variable attention. In *Conference on Empirical Methods in Natural Language Processing*, pages 1367–1372.

Mingxuan Wang, Zhengdong Lu, Hang Li, and Qun Liu. 2015b. Syntax-based deep matching of short texts. *Computer Science*.

Zhongyuan Wang, Kejun Zhao, Haixun Wang, Xiaofeng Meng, and Ji-Rong Wen. 2015c. Query understanding through knowledge-based conceptualization. In *Proceedings of the 24th International Conference on Artificial Intelligence*.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2015. Towards universal paraphrastic sentence embeddings. *Computer Science*.

Wentao Wu, Hongsong Li, Haixun Wang, and Kenny Q. Zhu. 2012. Probase: a probabilistic taxonomy for text understanding. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, pages 481–492.