Joint Learning of Character and Word Embeddings

Xinxiong Chen[1,2*], **Lei Xu**[1*], **Zhiyuan Liu**[1,2†], **Maosong Sun**[1,2], **Huanbo Luan**[1]
[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, China
[2] Jiangsu Collaborative Innovation Center for Language Ability,
Jiangsu Normal University, Xuzhou 221009 China

Abstract

Most word embedding methods take a word as a basic unit and learn embeddings according to words' external contexts, ignoring the internal structures of words. However, in some languages such as Chinese, a word is usually composed of several characters and contains rich internal information. The semantic meaning of a word is also related to the meanings of its composing characters. Hence, we take Chinese for example, and present a character-enhanced word embedding model (CWE). In order to address the issues of character ambiguity and non-compositional words, we propose multiple-prototype character embeddings and an effective word selection method. We evaluate the effectiveness of CWE on word relatedness computation and analogical reasoning. The results show that CWE outperforms other baseline methods which ignore internal character information. The codes and data can be accessed from `https://github.com/Leonard-Xu/CWE`.

1 Introduction

As the foundation of text representation, word representation aims at representing a word as a vector, which can be used to both compute semantic relatedness between words and feed machine learning systems as word features.

Many NLP tasks conventionally take one-hot word representation, in which each word is represented as a vocabulary-size vector with only one non-zero entry. Due to its simplicity, one-hot representation has been widely adopted in NLP and IR as the basis of bag-of-words (BOW) document models [Manning *et al.*, 2008]. The most critical flaw of one-hot representation is that, it does not take into account any semantic relatedness between words.

Distributed word representation, also known as word embedding, was first proposed in [Rumelhart *et al.*, 1986]. Word embedding encodes the semantic meanings of a word into a real-valued low-dimensional vector. Recent years have witnessed major advances of word embedding, which has been widely used in many NLP tasks including language modeling [Bengio *et al.*, 2003; Mnih and Hinton, 2008], word sense disambiguation [Chen *et al.*, 2014], semantic composition [Zhao *et al.*, 2015], entity recognition and disambiguation [Turian *et al.*, 2010; Collobert *et al.*, 2011], syntactic parsing [Socher *et al.*, 2011; 2013] and knowledge extraction [Lin *et al.*, 2015].

The training process of most previous word embedding models exhibits high computational complexity, which makes them unable to work for large-scale text corpora efficiently. Recently, [Mikolov *et al.*, 2013] proposed two efficient models, continuous bag-of-words model (CBOW) and Skip-Gram model, to learn word embeddings from large-scale text corpora. 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. An example of CBOW is shown in Fig. 1(A), where yellow boxes are word embeddings of context words, which are combined together to get the embedding (the orange box) for the prediction of the target word.

Most methods typically learn word embeddings according to the external contexts of words in large-scale corpora. However, in some languages such as Chinese, a word, usually composed of several characters, contains rich internal information. Take a Chinese word "智能" (intelligence) for example. The semantic meaning of the word "智能" can be learned from its context in text corpora. Meanwhile, we emphasize that its semantic meaning can also be inferred from the meanings of its characters "智" (intelligent) and "能" (ability). Due to the linguistic nature of semantic composition, the semantic meanings of internal characters may also play an important role in modeling semantic meanings of words. Hence an intuitive idea is to take internal characters into account for learning word embeddings.

In this paper, we consider Chinese as a typical language. We take advantages of both internal characters and external contexts, and propose a new model for joint learning of character and word embeddings, named as character-enhanced word embedding model (CWE). In CWE, we learn and maintain both word and character embeddings together. CWE can be easily integrated in word embedding models and one of the frameworks of CWE based on CBOW is shown in Fig. 1(B), where the word embeddings (blue boxes in figure) and character embeddings (green boxes) are composed together to get new embeddings (yellow boxes). The new embeddings

*Indicates equal contribution.
†Corresponding author: Zhiyuan Liu (liuzy@tsinghua.edu.cn).

perform the same role as the word embeddings in CBOW.

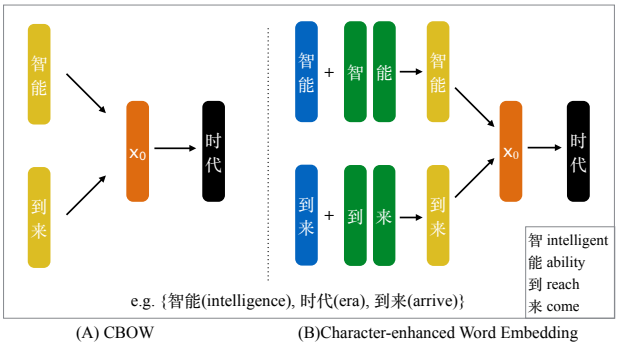

Figure 1: CBOW and CWE.

The framework of CWE seems a simple extension from other word embedding models. However, it faces several difficulties to consider characters into learning word embeddings. (1) Compared with words, Chinese characters are much more ambiguous. A character may play different roles and have various semantic meanings in different words. It will be insufficient to represent one character with only one vector. (2) Not all Chinese words are semantically compositional, such as transliterated words. The consideration of characters in these words will undermine the quality of embeddings for both words and characters.

In this paper, we rise to these challenges with the following methods. (1) We propose multiple-prototype character embeddings. We obtain multiple vectors for a character, corresponding to various meanings of the character. We propose several possible methods for multiple-prototype character embeddings: position-based, cluster-based and nonparametric method. (2) We identify non-compositional words and build a wordlist in advance. Then we treat these words as a whole without considering their characters any more.

In the experiments, we use the tasks of word relatedness and analogical reasoning to evaluate the performance of CWE as well as baselines including CBOW, Skip-Gram and GloVe [Pennington *et al.*, 2014]. The results show that, by successfully enhancing word embeddings with character embeddings, CWE significantly outperforms all baselines.

Note that, our method has great expansibility in two aspects. (1) As shown in this paper, it can be easily integrated in various word embedding methods, including the frameworks of neural network models (CBOW and Skip-Gram) and matrix factorization models (GloVe), and achieve considerable improvements. (2) Our method can also be applied to various languages in which words contain rich internal information and have to deal with the ambiguity issue.

2 Our Model

We will take CBOW for example and demonstrate the framework of CWE based on CBOW.

2.1 CBOW

CBOW aims at predicting the target word, given context words in a sliding window. Formally, given a word sequence $D = \{x_1, \ldots, x_M\}$, the objective of CBOW is to maximize the average log probability

$$\mathcal{L}(D) = \frac{1}{M} \sum_{i=K}^{M-K} \log \Pr(x_i | x_{i-K}, \ldots, x_{i+K}). \quad (1)$$

Here K is the context window size of a target word. CBOW formulates the probability $\Pr(x_i | x_{i-K}, \ldots, x_{i+K})$ using a softmax function as follows

$$\Pr(x_i | x_{i-K}, \ldots, x_{i+K}) = \frac{\exp(\mathbf{x}_o^\top \cdot \mathbf{x}_i)}{\sum_{x'_i \in W} \exp(\mathbf{x}_o^\top \cdot \mathbf{x}'_i)}, \quad (2)$$

where W is the word vocabulary, $\mathbf{x}_i$ is the vector representation of the target word x_i, and $\mathbf{x}_o$ is the average of all context word vectors

$$\mathbf{x}_o = \frac{1}{2K} \sum_{j=i-K,\ldots,i+K, j \neq i} \mathbf{x}_j. \quad (3)$$

In order to make the model efficient for learning, hierarchical softmax and negative sampling are used when learning CBOW [Mikolov *et al.*, 2013].

2.2 Character-Enhanced Word Embedding

CWE considers character embeddings in an effort to improve word embeddings. We denote the Chinese character set as C and the Chinese word vocabulary as W. Each character $c_i \in C$ is represented by vector $\mathbf{c}_i$, and each word $w_i \in W$ is represented by vector $\mathbf{w}_i$.

As we learn to maximize the average log probability in Equation (1) with a word sequence $D = \{x_1, \ldots, x_M\}$, we represent context words with both character embeddings and word embeddings to predict target words. Formally, a context word x_j is represented as

$$\mathbf{x}_j = \mathbf{w}_j \oplus \frac{1}{N_j} \sum_{k=1}^{N_j} \mathbf{c}_k, \quad (4)$$

where $\mathbf{w}_j$ is the word embedding of x_j, N_j is the number of characters in x_j, $\mathbf{c}_k$ is the embedding of the k-th character c_k in x_j, and $\oplus$ is the composition operation.

We have two options for the operation $\oplus$, addition and concatenation. For the addition operation, we require the dimensions of word embeddings and character embeddings to be equal (i.e., $|\mathbf{w}_j| = |\mathbf{c}_k|$). We simply add the word embedding with the average of character embeddings to obtain $\mathbf{x}_j$. On the other hand, we can also concatenate the word embedding and the average of character embeddings into the embedding $\mathbf{x}_j$ with a dimension of $|\mathbf{w}_j| + |\mathbf{c}_k|$. In this case, the dimension of word embeddings is not necessarily equal to that of character embeddings. In the experiments, we find the concatenation operation, although being more time consuming, does not outperform the addition operation significantly, hence we only consider the addition operation for simplicity in this paper. Technically, we use

$$\mathbf{x}_j = \frac{1}{2}(\mathbf{w}_j + \frac{1}{N_j} \sum_{k=1}^{N_j} \mathbf{c}_k). \quad (5)$$

Note that multipling $\frac{1}{2}$ is crucial because it maintains similar length between embeddings of compositional and non-compositional words. Moreover, we ignore the character embeddings on the side of target words in negative sampling and hierarchical softmax for simplicity.

The pivotal idea of CWE is to replace the stored vectors $\mathbf{x}$ in CBOW with real-time compositions of $\mathbf{w}$ and $\mathbf{c}$, but shares the same objective in Equation (1). As a result, the represent of word x_i will change due to the change of character embeddings $\mathbf{c}$ even when the word is not inside the context window.

2.3 Multiple-Prototype Character Embeddings

Chinese characters are highly ambiguous. Here we propose multiple-prototype character embeddings to address this issue. The idea is that, we keep multiple vectors for one character, each corresponding to one of the meanings.

We propose several methods for multiple-prototype character embeddings: (1) Position-based character embeddings; (2) Cluster-based character embeddings; and (3) Nonparametric cluster-based character embeddings.

Position-based Character Embeddings

In Chinese, a character usually plays different roles when it is in different positions within a word. Hence, we keep three embeddings for each character c, $(\mathbf{c}^B, \mathbf{c}^M, \mathbf{c}^E)$, corresponding to its three types of positions in a word, i.e., `Begin`, `Middle` and `End`.

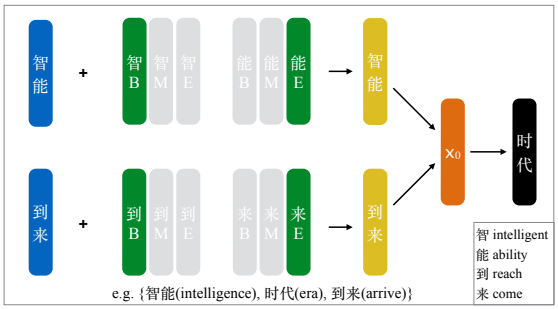

Figure 2: Position-based character embeddings for CWE.

As demonstrated in Fig. 2, we take a context word and its characters, $x_j = \{c_1, \ldots, c_{N_j}\}$, for example. We will take different embeddings of a character according to its position within x_j. That is, when building the embedding $\mathbf{x}_j$, we will take the embedding $\mathbf{c}_1^B$ for the beginning character c_1 of the word x_j, take the embeddings $\mathbf{c}_k^M$ for the middle characters $\{c_k | k = 2, \ldots, N_j - 1\}$, and take the embedding $\mathbf{c}_{N_j}^E$ for the last character c_{N_j}. Hence, Equation (4) can be rewritten as

$$\mathbf{x}_j = \frac{1}{2}\Big(\mathbf{w}_j + \frac{1}{N_j}(\mathbf{c}_1^B + \sum_{k=2}^{N_j-1} \mathbf{c}_k^M + \mathbf{c}_{N_j}^E)\Big), \tag{6}$$

which can be further used to obtain $\mathbf{x}_0$ using Equation (3) for optimization.

In the position-based CWE, various embeddings of each character are differentiated by the character position in the word, and the embedding assignment for a specific character in a word can be automatically determined by the character position. However, the exact meaning of a character is not only related to its position in a word. Motivated by multiple-prototype methods for word embeddings, we propose cluster-based character embeddings for CWE.

Cluster-based Character Embeddings

Following the method of multiple-prototype word embeddings [Huang *et al.*, 2012], we can also simply cluster all occurrences of a character according to its context and form multiple prototypes of the character. For each character c, we may cluster all its occurrences into N_c clusters, and build one embedding for each cluster.

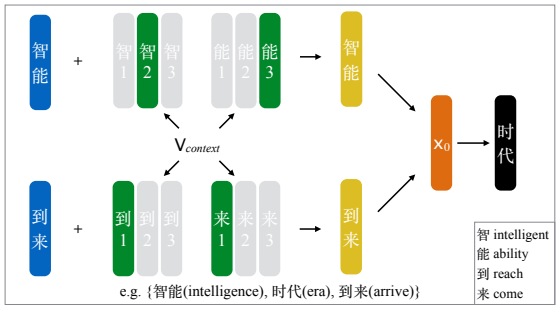

Figure 3: Cluster-based character embeddings for CWE.

As demonstrated in Fig. 3, take context word $x_j = \{c_1, \ldots, c_N\}$ for example, $\mathbf{c}_k^{r_k^{\max}}$ will be used to get $\mathbf{x}_j$. Define $S()$ as cosine similarity, then

$$r_k^{\max} = \arg\max_{r_k} S(\mathbf{c}_k^{r_k}, \mathbf{v}_{context}), \tag{7}$$

where

$$\mathbf{v}_{context} = \sum_{t=j-K}^{j+K} \mathbf{x}_t = \sum_{t=j-K}^{j+K} \frac{1}{2}(\mathbf{w}_t + \frac{1}{N_t} \sum_{c_u \in x_t} \mathbf{c}_u^{\text{most}}). \tag{8}$$

$\mathbf{c}_u^{\text{most}}$ is the character embedding most frequently chosen by x_t in the previous training. After obtaining the optimal cluster assignment collection $R = \{r_1^{\max}, \ldots, r_{N_j}^{\max}\}$, we can get the embedding $\mathbf{x}_j$ of x_j as

$$\mathbf{x}_j = \frac{1}{2}(\mathbf{w}_j + \frac{1}{N_j} \sum_{k=1}^{N_j} \mathbf{c}_k^{r_k^{\max}}), \tag{9}$$

and correspondingly get the embedding of $\mathbf{x}_0$ according to Equation (3) for optimization.

Note that, we can also apply the idea of clustering to position-based character embeddings. That is, for each position of a character (B, M, E), we learn multiple embeddings to solve the possible ambiguity issue confronted in this position. This may be named as position-cluster-based character embeddings.

Nonparametric Cluster-based Character Embeddings
The above hard cluster assignment is similar to the k-means clustering algorithm, which learns a fixed number of clusters for each character. Here we propose a nonparametric version of cluster-based character embeddings, which learns a varying number of clusters for each character. Following the idea of online nonparametric clustering algorithm [Neelakantan *et al.*, 2014], the number of clusters for a character is unknown, and is learned during training.

Suppose N_{c_k} is the number of clusters associated with the character c_k. For the character c_k in a word x_j, the cluster assignment r_k is given by

$$r_k = \begin{cases} N_{c_k}+1, & \text{if } S(\mathbf{c}_k^{r_k}, \mathbf{v}_{context}) < \lambda \text{ for all } r_k. \\ r_k^{\max}, & \text{otherwise.} \end{cases} \quad (10)$$

2.4 Word Selection for Learning

There are many words in Chinese which do not exhibit semantic compositions from their characters. These words include: (1) single-morpheme multi-character words, such as "琵琶" (lute), "徘徊" (wander), where these characters are hardly used in other words; (2) transliterated words, such as "沙发" (sofa), "巧克力" (chocolate), which shows mainly phonetic compositions; and (3) many entity names such as person names, location names and organization names.

To prevent the interference of non-compositional words, we propose not to consider characters when learning these words, and learn both word and character embeddings for other words. We simply build a word list about transliterated words manually, and perform Chinese POS tagging to identify all entity names. Single-morpheme words almost do not influence modeling because their characters usually appear only in these words, which are not specially dealt with.

2.5 Initialization and Optimization

Following the similar optimization scheme as that of CBOW used in [Mikolov *et al.*, 2013], we use stochastic gradient descent (SGD) to optimize CWE models. Gradients are calculated using the back-propagation algorithm.

We can initialize both word and character embeddings at random like CBOW, Skip-Gram and GloVe. Initialization with pre-trained character embeddings may achieve a slightly better result. We can obtain pre-trained character embeddings by simply regarding each character in the corpora as an individual word and learning character embeddings with word embedding models.

2.6 Complexity Analysis

We take CBOW and the corresponding CWE models for example to analyze model complexities. For CWE, we denote CWE with position-based character embeddings as CWE+P, and CWE with cluster-based character embeddings as CWE+L, CWE with nonparametric cluster-based character embeddings as CWE+N, and CWE with position-cluster-based character embeddings as CWE+LP. The complexity of each model is shown in Table 1.

Model Parameters. The table shows the complexity of model parameters in each model. In the table, the dimension of representation vectors is T, the word vocabulary size is $|W|$, the character vocabulary size is $|C|$, the number of character positions in a word is $P = 3$, the number of clusters for each character is L, and the average number of nonparametric clusters for each character is $\hat{L}$.

Table 1: Model complexities.

Model	Model Parameters	Computational Complexity
CBOW	$\|W\|T$	$2KMF_0$
CWE	$(\|W\|+\|C\|)T$	$2KM(F_0+\hat{N})$
CWE+P	$(\|W\|+P\|C\|)T$	$2KM(F_0+\hat{N})$
CWE+L	$(\|W\|+L\|C\|)T$	$2KM(F_0+\hat{N}+L\hat{N})$
CWE+N	$(\|W\|+\hat{L}\|C\|)T$	$2KM(F_0+\hat{N}+\hat{L}\hat{N})$
CWE+LP	$(\|W\|+LP\|C\|)T$	$2KM(F_0+\hat{N}+L\hat{N})$

Computational Complexity. In the table, the CBOW window size is $2K$, the corpus size is M, the average number of characters of each word is $\hat{N}$, and the computational complexity of negative sampling and hierarchical softmax for each target word is F_0. In computational complexity, $O(2KMF_0)$ indicates the computational complexity of learning word representations with CBOW. CWE and its extensions have additional complexities of computing character embeddings $O(2KM\hat{N})$. CWE+L, CWE+N and CWE+LP also have to perform cluster selections, either $O(L\hat{N})$ or $O(\hat{L}\hat{N})$.

From the complexity analysis, we can observe that, compared with CBOW, the computational complexity of CWE does not increase much, although CWE models require more parameters to account for character embeddings.

3 Experiments and Analysis

3.1 Datasets and Experiment Settings

We select a human-annotated corpus with news articles from *The People's Daily* for embedding learning. The corpus has 31 million words. The word vocabulary size is 105 thousand and the character vocabulary size is 6 thousand (covering 96% characters in national standard charset GB2312). We set vector dimension as 200 and context window size as 5. For optimization, we use both hierarchical softmax and 10-word negative sampling. We perform word selection for CWE and use pre-trained character embeddings as well. We introduce CBOW, Skip-Gram and GloVe as baseline methods, using the same vector dimension and default parameters. We evaluate the effectiveness of CWE on word relatedness computation and analogical reasoning.

3.2 Word Relatedness Computation

In this task, each model is required to compute semantic relatedness of given word pairs. The correlations between results of models and human judgements are reported as the model performance. In this paper, we select two datasets, wordsim-240 and wordsim-296 for evaluation. In wordsim-240, there are 240 pairs of Chinese words and human-labeled relatedness scores. Of the 240 word pairs, the words in 233 word pairs have appeared in the learning corpus and there are new words in the left 7 word pairs. In wordsim-296, the words in

280 word pairs have appeared in the learning corpus and the left 16 pairs have new words.

We compute the Spearman correlation ρ between relatedness scores from a model and the human judgements for comparison. For CWE and other baseline embedding methods, the relatedness score of two words are computed via cosine similarity of word embeddings. Note that, CWE here is implemented based on CBOW and obtains word embeddings via Equation (4). For a word pair with new words, we assume its similarity is 0 in baseline methods since we can do nothing more, while CWE can generate embeddings for these new words from their character embeddings for relatedness computation. The evaluation results of CWE and baseline methods on wordsim-240 and wordsim-296 are shown in Table 2.

Table 2: Evaluation results on wordsim-240 and wordsim-296 ($\rho \times 100$).

Dataset	wordsim-240		wordsim-296	
Method	233 Pairs	240 Pairs	280 Pairs	296 Pairs
CBOW	55.69	55.85	61.81	55.75
Skip-Gram	56.27	56.12	58.79	51.71
GloVe	47.72	48.22	48.22	43.06
CWE	56.90	57.56	64.02	63.57
CWE+P	56.34	57.30	62.39	62.41
CWE+L	**59.00**	59.53	**64.53**	**63.58**
CWE+LP	57.98	58.84	63.63	63.01
CWE+N	58.81	**59.64**	62.89	61.08

From the evaluation results on wordsim-240, we observe that: (1) CWE and its extensions all significantly outperform baseline methods on both 233 word pairs and 240 word pairs. (2) Cluster-based extensions including +P, +LP and +N perform better than CWE, which indicate that modeling multiple senses of characters is important for character embeddings and position information is not adequate in addressing ambiguity. (3) The addition of 7 word pairs with new words does not cause significant change of correlations for both baselines and CWE methods. The reason is that, the 7 word pairs are mostly unrelated. The default setting of 0 in baseline methods is basically consistent with the fact.

From the evaluation results on wordsim-296, we observe that: The performance of baseline methods drop dramatically when adding 16 word pairs of new words, while the performance of CWE and its extensions keeps stable. The reason is that the baseline methods cannot handle these new words appropriately. For example, "老虎" (tiger) and "美洲虎" (jaguar) are semantically relevant, but the relatedness is set to 0 in baseline methods simply because "美洲虎" does not appear in the corpus, resulting in all baseline methods putting the word pair much lower than where it should be [1]. In contrast, CWE and its extensions compute the semantic relatedness of these word pairs much closer to human judgements. Since it is more often to see a new word in Chinese than a new character, CWE can easily cover all Chinese characters in these new words and provide useful information about their semantic meanings for computing the relatedness.

There is a side effect when considering character embeddings. That is, CWE methods will tend to misjudge the relatedness of two words with common characters. For example, the relatedness of word pair "肥皂剧" (soap opera) and "歌剧" (opera) and the word pair "电话" (telephone) and "回话" (reply) are overestimated by CWE methods in this task due to having common characters (i.e., "剧" and "话", respectively). In the future, we may take the importance of characters in a word into consideration for CWE methods.

3.3 Analogical Reasoning

This task consists of analogies such as " 男人 (man) : 女人 (woman) :: 父亲 (father) : ?". Embedding methods are expected to find a word x such that its vector $\mathbf{x}$ is closest to vec(女人) - vec(男人) + vec(父亲) according to the cosine similarity. If the word "母亲" (mother) is found, the model is considered having answered the problem correctly. Since there is no existing Chinese analogical reasoning dataset, we manually build a Chinese dataset consisting of $1,125$ analogies [2]. It contains 3 analogy types: (1) capitals of countries (687 groups); (2) states/provinces of cities (175 groups); and (3) family words (240 groups). The learning corpus covers more than 97% of all the testing words.

As we have mentioned, the idea of CWE can be easily adopted in many existing word embedding models. In this section, we implement CWE models based on CBOW, Skip-Gram and GloVe, and show their evaluation results on analogical reasoning in Table 3. Here we only report the results of CWE and CWE+P for their stability of performance when adopting to all three word embedding models.

Table 3: Evaluation accuracies ($\%$) on analogical reasoning.

Method	Total	Capital	State	Family
CBOW	54.85	51.40	66.29	62.92
+CWE	58.24	53.32	66.29	70.00
+CWE+P	60.07	54.36	66.29	73.75
Skip-Gram	69.14	62.78	82.29	80.83
+CWE	68.04	63.66	81.14	78.75
+CWE+P	72.07	65.44	**84.00**	**84.58**
GloVe	67.44	69.22	58.05	69.25
+CWE	70.42	70.01	64.00	76.25
+CWE+P	**72.99**	**73.26**	65.71	81.25

From Table 3, we observe that: (1) For CBOW, Skip-Gram and GloVe, most of their CWE versions consistently outperform the original model. This indicates the necessity of considering character embeddings for word embeddings. (2) Our CWE models can improve the embedding quality of all words, not only those words whose characters are considered for learning. For example, in the type of capitals of countries, all the words are entity names whose characters are not used for learning. CWE model can still make an improvement on this type as compared to baseline models. (3) As reported in [Mikolov *et al.*, 2013;

[1] The trick of counting common characters won't help much because there are many relevant words do not share common words, e.g., "狮子" (lion) and " 美洲虎" (jaguar).

[2] The dataset can be accessed from `https://github.com/Leonard-Xu/CWE`.

Pennington *et al.*, 2014], Skip-Gram and GloVe perform better on analogical reasoning than CBOW. By simply integrating the idea of CWE to Skip-Gram and GloVe, we achieve an encouraging increase of 3% to 5%. This indicates the generality of effectiveness of CWE.

3.4 Influence of Learning Corpus Size

We take the task of word relatedness computation for example to investigate the influence of corpus size for word embeddings. As shown in Fig. 4, We list the results of CBOW and CWE on wordsim-240 and wordsim-296 with various corpus size from 3MB to 180MB (whole corpus). The figure shows that, CWE can quickly achieve much better performance than CBOW when the learning corpus is still relatively small (e.g., 7MB and 15MB).

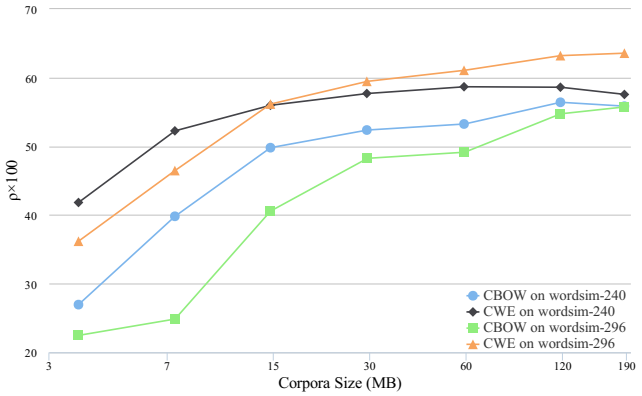

Figure 4: Results on wordsim task with different corpora size.

3.5 Case Study

Table 4 shows the quality of multiple-prototype character embeddings with their nearest words, using the results of CWE+P and CWE+L with 2 clusters for each character (marked with I and II in the table). For each embedding of a character, we list the words with the maximum cosine similarity among all words (including those which do not contain the character). Note that we use $\mathbf{x}_j$ in Equation (4) as the word embedding.

As shown in the table, the words containing the given character are successfully picked up as top-related words, which indicates the joint learning of character and word embeddings is reasonable. In most cases, both position- and cluster-based character embeddings can effectively distinguish different meanings of a character.

Examples of position-based character embeddings show that, position-based CWE works well while sometimes not. For the position-based character "道", the nearest words to "道-B" are closely related to *Taoist*, and the nearest words to "道-E" are about *road or path*. Meanwhile, for the character "法", whenever it is at the beginning or end of a word, its meaning can always be *law*. Hence, both "法-B" and "法-E" are learned related to *law*. On the other hand, cluster-based character embedding works generally well. For example, it successfully differentiates two different meanings of "法": *law* and *method*. But it may suffer from noise in some cases.

Table 4: Nearest words of each sense of example characters.

法-B	法政 (law and politics), 法例 (rule), 法律 (law), 法理 (principle), 法号 (religious name), 法书 (calligraphy)
法-E	懂法 (understand the law), 法律 (law), 消法 (elimination), 正法 (execute death)
法-I	法律 (law), 法例 (rule), 法政 (law and politics), 正法 (execute death), 法官 (judge)
法-II	道法 (an oracular rule), 求法 (solution), 实验法 (experimental method), 取法 (follow the method)
道-B	道行 (attainments of a Taoist priest), 道经 (Taoist scriptures), 道法 (an oracular rule), 道人 (Taoist)
道-E	直道 (straight way), 近道 (shortcut), 便道 (sidewalk), 半道 (halfway), 大道 (revenue), 车道 (traffic lane)
道-I	直道 (straight way), 就道 (get on the way), 便道 (sidewalk), 巡道 (inspect the road), 大道 (revenue)
道-II	道行 (attainments of a Taoist priest), 邪道 (evil ways), 道法 (an oracular rule), 论道 (talk about methods)

4 Related Work

Although a lot of neural network models have been proposed to train word embeddings, very little work has been done to explore sub-word units and how they can be used to compose word embeddings. [Collobert *et al.*, 2011] used extra features such as capitalization to enhance their word vectors, which can not generate high-quality word embeddings for rare words.

Some work tries to reveal morphological compositionality. [Alexandrescu and Kirchhoff, 2006] proposed a factored neural language model where each word is viewed as a vector of factors. [Lazaridou *et al.*, 2013] explored the application of compositional distributional semantic models, originally designed to learn phrase meanings, for derivational morphology. [Luong *et al.*, 2013] proposed a recursive neural network (RNN) to model morphological structure of words. [Botha and Blunsom, 2014] proposed a scalable method for integrating compositional morphological representations into a log-bilinear language model. These models are mostly sophisticated and task-specific, which make them non-trivial to be applied to other scenarios. CWE presents a simple and general way to integrate the internal knowledge (character) and external knowledge (context) to learn word embeddings, which are capable to be extended in various models and tasks.

Ambiguity is a common issue in natural languages. [Huang *et al.*, 2012] proposed a method of multiple embeddings per word to resolve this issue. To the best of our knowledge, little work has addressed the ambiguity issue of characters or morphemes, which is the crucial challenge when dealing with Chinese characters. CWE provides an effective and efficient solution to character ambiguity. Although this paper focuses on Chinese, our model deserves to be applied to other languages, such as English where affixes may have various meanings in different words.

5 Conclusion and Future Work

In this paper we introduce internal character information into word embedding methods to alleviate excessive reliance on external information. We present the framework of character-enhanced word embeddings (CWE), which can be easily integrated into existing word embedding models including CBOW, Skip-Gram and GloVe. In experiments of word relatedness computation and analogical reasoning, we have shown that the employing of character embeddings can consistently and significantly improve the quality of word embeddings. This indicates the necessity of considering internal information for word representations in languages such as Chinese.

There are several directions for our future work: (1) This paper presents an addition operation for semantic composition between word and character embeddings. Motivated by recent works on semantic composition models based on matrices or tensors, we may explore more sophisticated composition models to build word embeddings from character embeddings. This will endorse CWE with more powerful capacity of encoding internal character information. (2) CWE may learn to assign various weights for characters within a word. (3) In this paper we design a simple strategy to select non-compositional words. In future, we will explore rich information about words to build a word classifier for selection.

Acknowledgments

This work is supported by the 973 Program (No. 2014CB340501), the National Natural Science Foundation of China (NSFC No. 61133012, 61202140 and 61303075).

References

[Alexandrescu and Kirchhoff, 2006] Andrei Alexandrescu and Katrin Kirchhoff. Factored neural language models. In *Proceedings of the HLT-NAACL*, pages 1–4. Association for Computational Linguistics, 2006.

[Bengio *et al.*, 2003] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *JMLR*, 3:1137–1155, 2003.

[Botha and Blunsom, 2014] Jan A Botha and Phil Blunsom. Compositional morphology for word representations and language modelling. In *Proceedings of ICML*, pages 1899–1907, 2014.

[Chen *et al.*, 2014] Xinxiong Chen, Zhiyuan Liu, and Maosong Sun. A unified model for word sense representation and disambiguation. In *Proceedings of EMNLP*, pages 1025–1035, 2014.

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

[Huang *et al.*, 2012] Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. Improving word representations via global context and multiple word prototypes. In *Proceedings of ACL*, pages 873–882, 2012.

[Lazaridou *et al.*, 2013] Angeliki Lazaridou, Marco Marelli, Roberto Zamparelli, and Marco Baroni. Compositional-ly derived representations of morphologically complex words in distributional semantics. In *Proceedings of ACL*, pages 1517–1526, 2013.

[Lin *et al.*, 2015] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of AAAI*, 2015.

[Luong *et al.*, 2013] Thang Luong, Richard Socher, and Christopher Manning. Better word representations with recursive neural networks for morphology. In *Proceedings of CoNLL*, pages 104–113, 2013.

[Manning *et al.*, 2008] Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge, 2008.

[Mikolov *et al.*, 2013] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of NIPS*, pages 3111–3119, 2013.

[Mnih and Hinton, 2008] Andriy Mnih and Geoffrey E Hinton. A scalable hierarchical distributed language model. In *Proceedings of NIPS*, pages 1081–1088, 2008.

[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 EMNLP*, pages 1059–1069, 2014.

[Pennington *et al.*, 2014] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of EMNLP*, 2014.

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

[Socher *et al.*, 2011] Richard Socher, Cliff C Lin, Andrew Ng, and Chris Manning. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of ICML*, pages 129–136, 2011.

[Socher *et al.*, 2013] Richard Socher, John Bauer, Christopher D Manning, and Andrew Y Ng. Parsing with compositional vector grammars. In *Proceedings of ACL*, 2013.

[Turian *et al.*, 2010] Joseph Turian, Lev Ratinov, and Yoshua Bengio. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of ACL*, pages 384–394, 2010.

[Zhao *et al.*, 2015] Yu Zhao, Zhiyuan Liu, and Maosong Sun. Phrase type sensitive tensor indexing model for semantic composition. In *Proceedings of AAAI*, 2015.