Comparison Study on Critical Components in Composition Model for Phrase Representation

SHAONAN WANG, National Laboratory of Pattern Recognition, Institute of Automation, University of Chinese Academy of Sciences, Chinese Academy of Sciences
CHENGQING ZONG, National Laboratory of Pattern Recognition, Institute of Automation, CAS Center for Excellence in Brain Science and Intelligence Technology, University of Chinese Academy of Sciences, Chinese Academy of Sciences

Phrase representation, an important step in many NLP tasks, involves representing phrases as continuous-valued vectors. This article presents detailed comparisons concerning the effects of word vectors, training data, and the composition and objective function used in a composition model for phrase representation. Specifically, we first discuss how the augmented word representations affect the performance of the composition model. Then, we investigate whether different types of training data influence the performance of the composition model and, if so, how they influence it. Finally, we evaluate combinations of different composition and objective functions and discuss the factors related to composition model performance. All evaluations were conducted in both English and Chinese. Our main findings are as follows: (1) The Additive model with semantic enhanced word vectors performs comparably to the state-of-the-art model; (2) The Additive model which updates augmented word vectors and the Matrix model with semantic enhanced word vectors systematically outperforms the state-of-the-art model in bigram and multi-word phrase similarity task, respectively; (3) Representing the high frequency phrases by estimating their surrounding contexts is a good training objective for bigram phrase similarity tasks; and (4) The performance gain of composition model with semantic enhanced word vectors is due to the composition function and the greater weight attached to important words. Previous works focus on the composition function; however, our findings indicate that other components in the composition model (especially word representation) make a critical difference in phrase representation.

CCS Concepts: • **Computing methodologies → Lexical semantics**; **Language resources**

Additional Key Words and Phrases: Phrase representation, composition model, retrofitting, word paraphrasing, mean square error, max-margin

ACM Reference Format:
Shaonan Wang and Chengqing Zong. 2017. Comparison study on critical components in composition model for phrase representation. ACM Trans. Asian Low-Resour. Lang. Inf. Process. 16, 3, Article 16 (January 2017), 25 pages.
DOI: http://dx.doi.org/10.1145/3010088

1. INTRODUCTION

Learning phrase representation is the task of representing phrases as continuous-valued vectors to make similar phrases cluster more closely in the vector space. The

The research work is supported by the Natural Science Foundation of China under Grant No. 61333018 and the Strategic Priority Research Program of the CAS under Grant No.XDB02070007

Authors' addresses: S. Wang and C. Zong (corresponding author), Intelligence Building, No. 95, Zhongguancun East Road, Haidian District, Beijing, 100190, China; emails: {shaonan.wang, cqzong}@nlpr.ia.ac.cn.

DOI: http://dx.doi.org/10.1145/3010088

phrase representation is a fundamental and useful tool for a group of semantically related NLP tasks, such as paraphrase detection, textual entailment, question answering and machine comprehension, and the like. Moreover, phrases and sentences encode the general world knowledge more like that humans do [Norman 1972], a fact that is critically neglected in most of the current approaches to language understanding [Hill et al. 2016].

The problem of representing the meaning of phrases has traditionally been tackled by combining word vectors in conjunction with some functions to produce phrase vectors. Mitchell and Lapata [2008; 2010] constructed a similarity dataset for three types of bigram phrases: adjective-noun (AN), noun-noun (NN), and verb-object (VN) phrases, respectively. Then, they tested nine composition functions to assemble word meanings into the phrase meanings. They found that the simple Additive and multiplicative functions were as effective as more complex functions such as Dilation and Tensor. However, such simple composition functions ignore the word order and the interactions of words in phrases. This problem becomes more prevalent when dealing with sentences that have complex structures. To improve the performance of the composition model, researchers have proposed several more complicated composition functions including Matrix functions [Zanzotto et al. 2010; Socher et al. 2011, 2012], Tensor functions [Van de Cruys et al. 2013; Socher et al. 2013; Bride et al. 2015; Zhao et al. 2015] and linguistically motivated functions [Baroni and Zamparelli 2010; Guevara 2010; Grefenstette and Sadrzadeh 2011; Grefenstette et al. 2013].

More recent work has shown that word vectors trained with a neural network model are extremely efficient in composition models. A simple Additive composition function with word vectors produced by word2vec [Mikolov et al. 2013a] outperformed more complex composition functions with word vectors based on LSA or LDA models [Zhao et al. 2015; Iwai et al. 2015; Pham et al. 2015]. Hashimoto et al. [2014] proposed a compositional language model that works on Predicate Argument Structures (PASs) to learn the word representations. The PAS-based model can compose argument words into phrase representation, and it works well on simple composition tasks, especially SVO tasks. Socher et al. [2011] trained a recursive neural network (RecNN), whose structure is defined by a binarized parse tree, to learn representations for multi-word phrases. In particular, they used an unsupervised autoencoder to learn word vectors and compose words in the parse tree from bottom to top. Their model performed well on the sentiment classification task but poorly on phrase similarity related tasks because they used low-dimensional representations (25 or 50) to reduce the computational complexity. To avoid the high computational costs in autoencoder module, Wieting et al. [2015] used RecNN in a supervised setting to represent phrases. They leveraged PPDB paraphrase datasets to train their model and achieved state-of-the-art results on both the bigram phrase similarity task and the multi-word phrase similarity task. To reduce the computational complexity of the RecNN model and utilize the contextual features of phrases, Yu and Dredze [2015] proposed a method to learn phrase representations by composing representations of component words using weighted summation, in which the summation weights are defined by the features (parts-of-speech tags, word clusters, head words, and so on) of component words.

Another approach to represent the meanings of phrases is to directly estimate their surrounding contexts. Mikolov et al. [2013b] trained phrase representation using the same method used for word representation in word2vec. Specifically, they treated the high-frequency phrases[1] as pseudo-words and learned word representations and phrase representations together. However, this method cannot be generalized to longer

[1] In the work of Mikolov et al. [2013b], a phrase refers to n-grams that co-occur with high frequency, so most of them are not linguistically phrased.

phrases or sentences because the relatively rare occurrences of longer phrases in the corpus are quantitatively insufficient to learn good representations.[2] Still, this method is efficient for learning short phrase representations with good quality. Therefore, researchers have used these phrase representations as supervised training objectives to learn how to compose word vectors into phrase vectors [Zhao et al. 2015; Dima 2015].

In this article, we are concerned with the problem of how to represent phrases in a compositional manner, which is the most reasonable way to make the phrases consistent with human cognition [Fyshe 2015]. The construction and use of a composition model for phrase representation involves many design choices including the representation of the basic units, the type of training data, the composition function that assembles the basic unit representation, and the objective function to estimate the model parameters. The existing works mainly focus on one or two choices. Blacoe and Lapata [2012] compared combinations of three types of word representations with three composition functions to investigate their effects on the composition model. Milajevs et al. [2014] evaluated three word representations on a tensor-based composition model. Both the studies concluded that the different word representations and composition function combinations should be considered for different tasks. Dinu et al. [2013] and Dima [2015] provided an empirical performance evaluation of different composition models in English and German, respectively. The limitation of existing work is that while they compare different word representations and composition functions, they ignore the other components of the composition model. Therefore, the governing effects of different components in the composition model and their influences on model performance have not yet been fully investigated. We believe an extensive evaluation is essential to fully understand the effects of different composition models on phrase representation. This article makes contributions toward this goal: it systematically compares the four critical components in composition models both in English and Chinese. After performing the detailed investigations, we have the following main findings:

(1) The Additive model with semantic enhanced word vectors performs comparably to the state-of-the-art model;
(2) The Additive model which updates augmented word vectors and the Matrix model with semantic enhanced word vectors systematically outperforms the state-of-the-art model in bigram and multi-word phrase similarity task respectively;
(3) In the absence of high-quality training paraphrases, phrase vectors are a good training objective in bigram phrase similarity tasks;
(4) The high performance of the composition model with semantic enhanced word vectors is due to the composition function and the greater weight attached to important words.

The contributions of our work can be summarized as follows:

—Both qualitative and quantitative analyses of the four critical composition model components and their combinations on phrase similarity tasks have been made. We believe the findings listed above will be very helpful in finding appropriate and correct composition models for different tasks.
—Two new datasets for evaluating Chinese phrase similarities, bigram phrase similarity dataset and multi-word phrase similarity dataset, have been developed. Both datasets are carefully annotated with similarity scores assigned by hands. We

[2] Tian et al. [2015] made a quantitative statistics in British National Corpus (BNC, http://www.natcorp.ox.ac.uk/). The conclusion is that there are too few training samples for even two-word phrases.

believe these datasets are very helpful in evaluating how well the models measure the similarity of short phrases.[3]

2. COMPONENTS IN COMPOSITION MODEL

Various composition models for phrase representation have been proposed in recent years. Empirical evaluations of the composition models are complicated because constructing the models involves many choices. We will discuss the main choices separately in the following subsections.

2.1. Word Representations

A word representation associates a word with a high-dimensional vector. There are several ways to learn word representations. Baroni et al. [2014] summarized the existing models into two categories: the count model and the predict model. The count model learns word vectors by calculating the co-occurrence frequency of each word with features [Deerwester et al. 1990; Blei et al. 2003]. The predict model learns word vectors by maximizing the probability of the contexts in which the word is observed in the corpus [Bengio et al. 2003; Collobert and Weston 2008; Collobert et al. 2011; Huang et al. 2012; Mikolov et al. 2013a; Turian et al. 2010]. Among existing predict models, the Skip-Gram model and CBOW model included in word2vec tool [Mikolov et al. 2013a] are most widely used for generating word vectors [Fu et al. 2014; Taghipour and Ng 2015; Roth and Woodsend 2014]. Some work has attempted to explain the underlying principle of the Skip-Gram model and CBOW model both mathematically [Levy and Goldberg 2014] and empirically [Köhn 2015; Gupta et al. 2015]. Other works have attempted to improve the performance of word vectors both in general [Faruqui et al. 2015; Yu and Dredze 2014] and when using specific criteria [Kiela et al. 2015; Bollegala et al. 2016].

Taking performance and efficiency into account, we focus on three types of word representations in this article. One is the standard word representation learned directly by using of word2vec, a method described by many works. The other two are word vectors augmented beyond standard word vectors using two different methods: retrofitting and word paraphrasing. These two augmentation methods are introduced as follows:

- **Retrofitting Method**

The retrofitting method employs a graph-based learning technique, using word relations in semantic lexicon resources as restrictions to update word vectors. Experimental results on various word similarity datasets have proved its effectiveness [Faruqui et al. 2015]. Formally, the distance between a pair of vectors is defined as Euclidean distance. The objective of retrofitting method is to make the object word vector qi to be close to the observed value $\hat{q}_{\mathrm{i}}$ (pre-trained word vectors using standard data-driven method) and close to its neighbors q_i (adjacent vertices in semantic lexicon). The objective function is described as follows:

$$\psi(\mathrm{Q}) = \sum_{i=1}^{n} \left[\alpha_i \|q_i - \hat{q}_i\|^2 + \sum_{(i,j)\in E} \beta_{ij} \|q_i - q_j\|^2 \right], \tag{1}$$

where α and β values control the relative strengths of the associations,[4] E denotes the semantic relationship of interest, and n is the total number of training word pairs.

[3] The implementation of our model and the datasets is available for general use: http://www.nlpr.ia.ac.cn/cip/cqzong.htm or https://github.com/wangshaonan.

[4] When using PPDB data as semantic lexicons, α_i is set to 1 and β_{ij} is degree(i)$^{-1}$ (degree refers to paraphrase similarity provided in PPDB data).

• Word Paraphrasing

We use the word paraphrasing method described in Wieting et al. [2015]. This method trains word vectors with a contrastive max-margin objective function. Specifically, our training data consisting of a set X of word paraphrase pairs (x_1, x_2), while (t_1, t_2) are negative examples that are the most similar word pairs to (x_1, x_2) in a mini-batch during optimization. The objective function is given as follows:

$$\overset{\min}{W_w} \frac{1}{|X|}\left(\sum_{(x_1,x_2)\in X} \max\left(0, 1 - W_w^{x_1} \cdot W_w^{x_2} + W_w^{x_1} \cdot W_w^{t_1}\right)\right.$$
$$\left. + \max\left(0, 1 - W_w^{x_1} \cdot W_w^{x_2} + W_w^{x_2} \cdot W_w^{t_2}\right)\right) + \lambda \left\| W_{w_{initial}} - W_w \right\|^2, \qquad (2)$$

where λ is the regularization parameter, $|X|$ is the length of training paraphrase pairs, W_w is the target word vector matrix, and $W_{w_{initial}}$ is the initial word vector matrix.

2.2. Training Data

A composition model controls how word vectors are composed into phrase vectors. Simple composition models, such as Additive and Multiplicative models, do not have parameters, while others such as Matrix and Tensor models need training data to estimate their parameters. For bigram phrase similarity task, there are two general types of training data in existing work. One type consists of tuples in the form {adjective1 noun1, adjective1-noun1},[5] for instance, {older man, older-man}. The other type consists of {adjective1 noun1, adjective2 noun2} tuples, for example, {older man, elderly woman}. We call these two types of training data as pseudo-word training data and pair training data, respectively.

Pseudo-word training data have been widely used in composition models. Guevara [2010], Baroni and Zamparelli [2010], and Dinu et al. [2013] extracted example pairs {adjective1 noun1, adjective1-noun1} from corpora. For example, {county council, county-council}, {send message, send-message} and so on. They learn parameters that optimize the mapping from the composed adjective1 noun1 vectors to the "adjective1-noun1" vectors in corpus-extracted vector pairs. Using word2vec, Zhao et al. [2015] learn vectors of individual words and a collection of high-frequency phrases together. The learned phrase vectors are considered as gold training output, and the output of an ideal composition model should be approximate to the gold output.

Training data in the form of {adjective1 noun1, adjective2 noun2} can be considered to be a paraphrase training set. For example, {county council, town hall}, {send message, hear word} and so on. Wieting et al. [2015] extracted training paraphrases at a large scale from PPDB corpus. They trained the composition models with the objective of making phrase representation in training phrase pairs similar to each other.

For multi-word phrase, pair training data is the first choice because it is hard to learn the representation of pseudo-words with multiple words. Therefore, we only use pair training data for multi-word phrase experiments.

2.3. Composition Function

In general, the goal of a composition model is to transform the representations of component words into a representation that is close to the representation of phrase.

[5] "adjective1-noun1" means a pseudo-word with the component words adjective1 and noun1 linked by a hyphen ("-"). In the remainder of this paper, in general, we will use "adjective1 noun1" to refer to a bigram phrase where the first word is adjective1 and the second word is noun1.

Choosing the correct and good composition function is the key step in representing the meaning of phrase. Specifically, with a phrase p consisting of two words $w^{(1)}$ and $w^{(2)}$, we can establish Equation (3) as follows:

$$f\left(\vec{w}^{(1)}, \vec{w}^{(2)}\right) = \vec{p}, \tag{3}$$

where $\vec{w}^{(1)}$, $\vec{w}^{(2)}$ are the word representations, $\vec{p}$ is the phrase representation, and f is the composition function.

In recent years, several composition functions have been proposed, including Additive, Multiplicative, Tensor, and so on (see the summary in Dima [2015]). The Additive model assumes that the meaning of the composition is a linear combination of the constituent words. The Multiplicative model assumes that the meaning of composition is the element-wise product of the two vectors. Compared to the Additive and Multiplicative models, the Matrix and Tensor models transform word vectors to another space and express phrase meaning in a more flexible way using additional parameters.

More critically, the different composition models compose word vectors in specific ways that probably correspond to different cognitive processes [Chang et al. 2009; Chang 2011]. For example, they think that the Additive model described the phenomenon: people concatenate the meanings of two words when understanding phrases. Chang et al. [2009] showed that the multiplicative model has the highest correlation with the neural activity observed in human brain when reading adjective-noun phrase data. In Table I, we take bigram phrase as examples to show composition function and calculation. Given a phrase p which consists of two words u and v ($\hat{u}$ refers to the transformation of first word u which is the output of the first RNN step composing the initial hidden state with u), we have word vectors u, v and we want to calculate the phrase vector p. The subscript t means time point. $W \in R^{n\times 2n}$, $W_x \in R^{n\times n}$,$U \in R^{n\times n}$, $W_{p_{t-1}} \in R^{n\times n}$, $\mathrm{U}_i \in R^{n\times n}$, $\mathrm{U}_f \in R^{n\times n}$, $\mathrm{U}_o \in R^{n\times n}$, $\mathrm{U}_g \in R^{n\times n}$, $W_i \in R^{n\times n}$, $W_f \in R^{n\times n}$, $W_o \in R^{n\times n}$, $W_g \in R^{n\times n}$ are the composition functions which multiply with the corresponding vector. The b terms denote bias vectors, σ is the logistic sigmoid function, and *i, f, o* and *g* are, respectively, the input gate, forget gate, output gate, and cell activation vectors. The symbols $\oplus$ and $\odot$ mean element-wise additive and element-wise multiplicative, respectively.

Composition functions such as Matrix, RecNN and RNN (Recurrent neural network) transform word vectors into another vector space through matrix transformations and nonlinear functions. These three functions differ primarily in the order of transformation. The Matrix model first transforms component words into another vector space and then composes them using addition. The RecNN model takes word order into consideration, concatenates vector component words and then transforms the vector using a matrix and nonlinearity. The RNN model composes words in a sentence from left to right by forming new representations from previous representations ($\mathrm{p}_{\mathrm{t}-1}$ in Table I) and the representation of the current word.

2.4. Objective Function

Assume our training data consists of a set X that includes word paraphrase pairs[6] (x_1, x_2). Then, we calculate phrase representation $g(x_1)$ using the following equation:

$$g(x_1) = f(x_1; W, b), \tag{4}$$

where W is the composition function (e.g., W is a number in Additive function, W is a matrix in Matrix function), b is the offset value).

Traditional composition models consider the phrase similarity task as a regression problem and use a mean square error (MSE) objective function to estimate the model

[6] In pseudo-word training data, x_2 represents a pseudo-word.

Table I. Composition Functions and Calculation

Function	Formulation of composition function	Example of Composition Function Calculation
Additive	$p = u + v$	$p = [0\,6\,2\,10\,4] \oplus [1\,8\,4\,4\,0]$
Multi	$p = u\,.\,v$	$p = [0\,6\,2\,10\,4] \odot [1\,8\,4\,4\,0]$
Matrix	$p = \tanh(W_x u + W_x v)$	$p = \tanh\left(\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 2 & 3 \end{bmatrix} [0\,6\,2\,10\,4].T + \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 2 & 3 \end{bmatrix} [1\,8\,4\,4\,0].T\right)$
RecNN	$p = \tanh(W\,[u; v])$	$p = \tanh\left(\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 0 & 0 & 2 & 1 & 3 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 2 & 3 & 0 & 0 & 0 & 2 & 1 \end{bmatrix} [0\,6\,2\,10\,4\,1\,8\,4\,4\,0].T\right)$
RNN	$p_t = \tanh(U v + W_{pt-1}\hat{u})$	$p_t = \tanh\left(\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 2 & 3 \end{bmatrix} [18440],\,T + \begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ 3 & 1 & 3 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 2 & 1 \end{bmatrix} [1\,2\,0\,6\,2].T\right)$
LSTM	$i = \sigma(U_i v + W_i \cap u + b_i)$ $f = \sigma\left(U_f v + W_f \cap u + b_f\right)$ $o = \sigma(U_o v + W_o \cap u + b_o)$ $g = \tanh\left(U_g v + W_g \cap u + b_g\right)$ $c_t = c_{t-1} \odot f + g \odot i$ $p_t = \tanh(c_t) \odot o$	$p_t = \tanh(c_t) \odot \sigma\left(\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 3 & 2 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 2 & 3 \end{bmatrix}\right) [1\,8\,4\,4\,0].T + \begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ 3 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 2 & 1 \end{bmatrix} [1\,2\,0\,6\,2].T + [1\,2\,3\,4\,5].T$

parameters. MSE has been widely used for various text similarity tasks [Dinu et al. 2013; Grefenstette et al. 2013; Dima 2015; Mueller and Thyagarajan 2015]. The MSE objective function is as follows:

$$\mathrm{J} = \min_{W,\,b,\,W_w} \|\mathrm{g}(x_1) - \mathrm{g}(x_2)\|^2 + \lambda_{W_w} \|\mathrm{W}_{\mathrm{W}_{\mathrm{initial}}} - W_w\|^2 + \lambda_W(\|W\|^2 + \|b\|^2), \qquad (5)$$

where $W_w \in R^{v \times d}$ is the word representation matrix, v is the vocabulary size, d is the size of word dimension and λ_{W_w} and λ_W are regularization parameters.

Because the Additive model is a strong baseline in composition models, we initialize the composition matrix W to an identity matrix. We do not want it to be far away from the initial value. Thus, we add a regularization constraint item $\lambda_{W_w} \|\mathrm{W}_{\mathrm{W}_{\mathrm{initial}}} - W_w\|^2$.

Another objective function is max-margin. The max-margin methods such as support vector machines (SVMs) are usually applied to various discriminative problems. The purpose of this objective is to ensure that the score computed for the training example is higher than the score computed for the negative example [Socher et al. 2014].

Assume that (t_1, t_2) is a negative example sampled by the same method as word paraphrasing method in Equation (2). The max-margin objective function is computed as

$$\begin{aligned}\mathrm{J} = W, \overset{\min}{b}, W_w \, \frac{1}{|X|} \quad & \left(\sum_{(x_1, x_2) \in X} \max(0, 1 - \cos(g(x_1), g(x_2)) + \cos(g(x_1), g(t_1))) \right. \\ & \left. + \max(0, 1 - \cos(g(x_1), g(x_2)) + \cos(g(x_1), g(t_2))) \right) \\ & + \lambda_{W_w} \left\| W_{\mathrm{w_{initial}}} - W_w \right\|^2 \\ & + \lambda_W (\|W\|^2 + \|b\|^2), \end{aligned} \tag{6}$$

where W is the composition matrix, b is the offset, W_w is the word representation matrix, λ_{W_w} and λ_W are regularization parameters, $|X|$ is the length of the training paraphrase pairs, and $\|\cdot\|$ denotes L2 norm.

3. EXPERIMENTAL SETTINGS

3.1. Word Vector

- **Standard Word Vectors**

We use standard word vectors trained with Skip-gram model [Mikolov et al. 2013a]. The English word vectors are trained on collections of public corpora and preprocessed using script provided by word2vec toolkit (a total of 8B words last)[7] with the default parameter setting. The Chinese word vectors are trained on Xinhua News and Baidu encyclopedia corpora[8] (a total of 2.18B words) crawled from the Internet and segmented with Urheen[9]. We use the window size of 5 and the minimum-count cutoff of 50 with a negative sampling number of 5, producing 503K English word vectors and 318K Chinese word vectors. We kept only the 100K most frequent words, and averaged the rest to obtain a single vector for unknown words.

- **Augmented Word Vectors**

We use two methods to augment the standard word vectors. The first is retrofitting method, which updates word vectors by running belief propagation on a graph constructed from semantic lexicons. We set iteration number (the only hyper-parameter in retrofitting method) to 10, as same as the default setting. The second augmentation method is word paraphrasing, which updates word vectors with the constraint that word pairs in training data should have the similar vectors. Same as in Wieting et al. [2015], we set batch size to 50, iteration number to 10 and the regularization parameter to 10^{-9}. Neither augmentation method changes the size or the dimension of the vectors. For both methods, we use the word pairs provided by Wieting et al. [2015] which contains 186,733 word pairs in English. For Chinese, we use 100,000 word pairs extracted from English-Chinese parallel data in LDC corpora (Please see the details about the dataset in Section 3.2).

[7] Find the specific datasets and training parameters in the scripts provide in word2vec tool.

[8] http://baike.baidu.com/.

[9] http://www.openpr.org.cn/index.php/zh/NLP-Toolkit-For-Natural-Language-Processing/68-Urheen-A-Chinese/English-Lexical-Analysis-Toolkit/View-details.html.

3.2. Datasets

- **English Datasets for Multi-Word Phrases**

Pair-training data for multi-word phrase similarity tasks is extracted from the XL portion of PPDB-2.0. For instance, "*shall be adopted by*" and "*will now proceed to take*". PPDB-2.0 is a newly published dataset that includes a discriminative re-ranked set of paraphrases. Ranking scores in PPDB-2.0 achieve a higher correlation with human judgement than the heuristic ranking of PPDB-1.0. The extraction method of multi-word phrase training data refers to the method given in Wieting et al. [2015]. Specifically, we filter phrases that contain non-alphanumeric symbols and use edit distance to avoid the high degree of word overlap. To balance the length of the training data, we chose 20,000 phrase pairs in phrase lengths of 3, 4, and more than 5, respectively. Finally, we obtain 60,000 phrase pairs.

For the test and development data for the multi-word phrase similarity task, we use 1,000 phrase pairs for testing and 260 phrase pairs from Wieting et al. [2015] for development.

- **English Datasets for Bigram Phrase**

Pair-training data for bigram phrase similarity tasks has been extracted from the XL portion of PPDB-2.0. For example, "*crucial element*" and "*essential element*". Reference to Wieting et al. [2015], to extract specific types of bigram phrases, we use POS tagger [Manning et al. 2014] to tag the tokens in each phrase. Then, we extract the pairs containing aligned, adjacent tokens in the two phrases with appropriate part-of-speech tags. Finally, we obtained 66,427 AN pairs, 22,275 NN pairs, and 333,578 VN pairs.

Pseudo-word training data are used only in the bigram similarity task because there is limited data from which to obtain high quality multi-word phrase vectors. For instance, "*crucial element*" and "*crucial_element*". We chose those phrases with frequencies greater than 50 from bigram training phrases. Finally, we obtained 39,669 pseudo-words for training.

For the development and test data, we use the bigram similarity dataset from Mitchell and Lapata. [2010]. With reference to this work, we divided the dataset into a development set and a test set. Specifically, participants were randomly allocated to either the development set or the test set. For each experiment the test set contained 44 participants, and the development set contained 10.

- **Chinese Datasets**

There are no public Chinese datasets for testing phrase similarity. The multilingual PPDB-1.0 provides a Chinese paraphrase dataset (45,977,217 pairs), but most of the phrases in that dataset do not met the requirements for our linguistic analysis. For example, phrases like "同时, 正如(at the same time, as)" and "同时, 随着(at the same time, along with)", which is not linguistically valid, appear frequently. After we filtered out phrases containing non-alphanumeric symbols, we got 853,983 bigram phrase pairs and 167,130 multi-word phrase pairs. Moreover, most of them share a high word overlap ratio like "作为 主权 国家(as a sovereign state)" and "作为 一个 主权 国家(as one sovereign state)", which are less interesting in phrase similarity task. Therefore, we constructed Chinese multi-word phrase data and bigram phrase data as follows:

- **Construction of Chinese Datasets for Multi-word Phrase**

We extracted multi-word phrase data from the Chinese-English parallel dataset (0.6 M sentences) in the LDC corpora. First, we used Giza++ to learn word alignments. We wanted the phrase pairs to be linguistically valid, so we trained a CRF Chunker using the Penn Treebank corpora to tag the Chinese corpus. After chunking, we obtained lists of phrases (chunks) in Chinese and the corresponding phrase in English. Then, we filtered the phrase pairs by words not in the 100K most frequent words, phrases

Table II. Multi-Word Phrase Similarity Datasets in English and Chinese

English multi-word phrase			Chinese multi-word phrase		
Train	Dev	Test	Train	Dev	Test
60,000	260	1,000	40,000	264	1,000

with 2 words or less, high-overlap phrase pairs, and phrase pairs with large sentence-length gaps. Next, we extracted the Chinese phrase pairs using the principle that "phrases aligned to the same target should have the same meaning" using paraphrase probabilities introduced by Chris et al. [2006]. Thus, we get 112988 raw paraphrases.

In order to get high-quality paraphrases as training data, we use a simple Cilin similarity method which is defined as the distance in Tongyi Cilin[10] dictionary to calculate the similarity between phrase pairs[11] in paraphrases. Then, we extracted 40,000 pairs of most high similarity scores with lengths of 3, 4, and more than 5, consisting of 20,000, 14,348, and 5,652 pairs, respectively.

For the testing and development data, we selected 1,500 phrase pairs randomly from the raw paraphrases mentioned above. Three volunteers evaluated these phrase pairs, giving each phrase pair a similarity score. Finally, we obtained 1,000 test pairs and 264 development pairs. Table II lists the size of multi-word phrase dataset in English and Chinese.

- **Construction of Chinese Datasets for Bigram Phrase**

The training data for the bigram phrase similarity tasks was extracted from the Chinese-English parallel data as above by the same methods used for the English data. In the end, we obtained 2,577 AN pairs, 3,376 NN pairs, and 744 VN pairs.

For pseudo-word training data, we chose those phrases with frequencies greater than 50 from the Chinese bigram training phrases, leaving 5,106 bigram phrases.

To build the testing and development set in the Chinese bigram phrase similarity task, we chose candidate phrases from the Chinese Gigaword and Baidu encyclopedia corpora (3B words in total). To select phrase pairs with similarities ranging from high to low, we used a Chinese semantic thesaurus, Tongyi Cilin, to evaluate phrase similarities as described above. Finally, we obtained 120 pairs of phrases for each phrase type (AN, NN, VN).

This article aims to collect human ratings of phrase similarity other than phrase relatedness. Therefore, we explain these two concepts and give examples in the experimental instruction to guide the participants to assess similarity of the phrase pairs. For example, "著名歌星(famous singer)" and "流行歌曲(pop songs)". These two phrases are related because of the fact that singers often sing songs, but the former is a person and the latter is a song, which are pretty different concepts. So we should give low score to these phrase pairs. We collected human ratings of phrase similarity via an online questionnaire; participants were paid 2 cents for rating each phrase pair. In total, we obtained 178 valid questionnaires; every phrase pair was evaluated by 45 persons on average.

[10] http://ir.hit.edu.cn/demo/ltp/Sharing_Plan.html.

[11] Specifically, each word in Cilin (Che et al. 2010) has one or more sense code which indicates its position in the hierarchy of word senses. We compute the similarity of phrase pairs via computing the overlap ratio of the corresponding position of words in phrase. Take the phrase pair of "野生动物(wild animal)" and "水生植物(aquatic plant)" for instance, "野生(wild)" and "水生(aquatic)" have the same code of "Ib01C07" in Cilin, and the code for "动物(animal)" and "植物(plant)" is "Ba02A02" and "Ba02A07". The overlap ratio of this phrase pair is overlap of code of "野生(Ib01C07)" and "水生(Ib01C07)" plus "动 物(Ba02A02)" and "植物(Ba02A07)", which is 13 (=7 + 6).

Table III. Inter-Annotator Agreement on Different Types of Phrases Using Spearman Correlations

	AN	NN	VN
Mean	0.836	0.784	0.818
Max	0.916	0.905	0.916
Min	0.562	0.590	0.609
SD	0.065	0.074	0.067

Table IV. Bigram Phrase Similarity Datasets in English and Chinese

		English bigram			Chinese bigram		
Data type		Train	Dev	Test	Train	Dev	Test
Pair	AN	66,427	540	1,404	2,577	1,200	3,600
	NN	22,275	540	1,404	3,376	1,200	3,480
	VN	33,578	540	1,404	744	1,200	4,320
Pseudo-word	AN	24,289	540	1,404	1,471	1,200	3,600
	NN	9,733	540	1,404	2,503	1,200	3,480
	VN	6,049	540	1,404	451	1,200	4,320

In the following, we take a look at the quality of the data we collected. Especially, reference to Mitchell and Lapata [2010], we examine how well participants agreed in their similarity judgments for each phrase type, which is called *intersubject agreement*. The indicator of intersubject agreement is an upper bound for the task and allows us to evaluate how well our model performs comparing with humans. To calculate intersubject agreement, we use leave-one-out cross-validation method. For each subject group we divided the set of the subjects' responses with size m into a set of size m-1 (we average the m-1 human data) and a set of size one. We then correlated the ratings of the former set with the ratings of the latter using Spearman's correlation coefficient ρ. This was repeated m times and we get the results in Table III.

In Table III, higher values mean more consistency among the participants.[12] A value of 1 would means that the ratings of all participants were the same. As shown by the Mean (row 2) and SD (row 5) results, there was high consistency among the questionnaire participants' responses—even though the participants thought that the phrase similarity evaluation task was difficult.

Reference to Mitchell and Lapata [2010], participants were randomly allocated to a development set used for optimizing model parameters and a test set used for the final evaluation of all models[13]. For each experiment, the test set contained approximately 30 items and the development set[14] contained 10. We list the size of bigram phrase dataset[15] in English and Chinese in Table IV.

For all the experiments, we deleted any training data that also appeared in the development set or the testing set.

[12] "Mean", "Max", "Min" and "SD" represents average value, maximum value, minimum value and standard deviation, respectively.

[13] One phrase pairs is annotated by around 50 persons which is independent. Thus, Mitchell and Lapata [2010] separate collected data by different persons as the development set.

[14] In fact, the Chinese development dataset is larger than the English development dataset. In English bigram similarity datasets, participants evaluated only a portion of the candidate data. In the Chinese data, every participant evaluates all the candidate data.

[15] The size of development set and test set in Table IV are all annotations of the corpus. There are 108 unique English phrase pairs for each phrase type and 120 unique Chinese phrase pairs for each phrase type.

Table V. Correlation Coefficients of Model Predictions with Subject Similarity Ratings on English Word Similarity Tasks

	50	100	200	Para50	Para100	Para200	Ret50	Ret100	Ret200
353	0.662	0.699	0.731	0.719	0.733	**0.755**	0.678	0.713	0.734
Sim	0.718	0.763	0.775	0.765	0.793	**0.795**	0.746	0.78	0.781
Rel	0.599	0.63	0.667	0.639	0.645	**0.674**	0.6	0.631	0.662
999	0.267	0.309	0.354	0.502	0.545	**0.561**	0.398	0.436	0.475

3.3. Training Procedure

Our model is trained using AdaGrad [Duchi et al. 2011]. We initialized the composition functions in Matrix model as identity matrix and we used orthogonal initialization in RNN and LSTM because this is the most suitable value in these tasks according to our experiments. We fixed the initial learning rate to 0.05. The parameters of the composition models are W_w, λ_W, λ_{W_w} and mini-batch size. We employed a coarse grid search over a parameter space for λ_W, λ_{W_w} and mini-batch size. We considered λ_W values in $\{10^3, 10^4, 0\}$, λ_{W_w} values in $\{10, 1, 10^1, 10^2, 10^3, 10^4, 0\}$ and mini-batch size in {50, 100, 200, 500, 1,000, 2,000}. The hyper-parameters were selected by testing different parameter values and evaluating their effects on the development set. We trained each set of parameters for five epochs and used early stopping to avoid overfitting. In this article, we run all experiments for three times and reported the mean values. We used Spearman correlation method to evaluate the models.

4. RESULTS

In this section, we first discuss how different word representations affect the performance of the composition model in phrase similarity tasks. We then verify whether pseudo-word training data degrade the performance of the composition model and, if so, discuss how that occurs. Finally, we compare different combinations of composition functions and objective functions and discuss the potential factors that are related to the models' performances.

4.1. Effect of Different Word Representations

In this subsection, we look into influence of different word representations in English and Chinese, respectively. Wieting et al. [2015] showed the amazing performance of augmented word representation using word paraphrasing method on phrase similarity task. In this article, we provide a more comprehensive study on effect of different word representations, which considers the standard word representations and two augmented word representations with different word dimensions on both word similarity task and phrase similarity task. In particular, we do experiments on both English and Chinese.

• **English**

Table V summarizes the results of word similarity tests (Wordsim353, Wordsim-sim, Wordsim-rel and Simlex-999). Number 50, 100, 200 in Table V means the dimension of word vectors. As Table V shows, the paraphrasing and retrofitting methods improved the word representation performance in modeling word similarity. We thus argue that these augmentation methods can help the word representation to capture more semantic information. What's more, the test performance is positively correlated to the dimensionality of word representations. This indicates that the higher dimensional word vectors are more powerful in capturing word semantics.

To demonstrate the intuitive impact of word vector augmentation, we present some examples in Table VI.

Table VI. Five Most Similar Words of Different Word Representations Calculate with Cosine Similarity

	200	Para200	Ret200
Buy	sell poster/ resell psychedelics, artwork/image	buys purchase purchased sell buying	purchase buying buys resell sell
Scholarship	fulbright scholarshipes professorship fellowships	scholarships bursary scholarships bursaries	fulbright scholarship bursary fellowships
	doctorate	fellowship	bursaries
Cute	latm eteeq autostale longew lat_ns	adorable autostale lovable girly charming	adorable nerdy geeky goofy lovable

Table VII. Correlation Coefficients of Additive Model Predictions with Subject Similarity Ratings on Phrase Similarity Tasks

	50	100	200	Para50	Para100	Para200	Ret50	Ret100	Ret200
AN	0.436	0.471	0.464	**0.528**	0.518	0.523	0.499	0.52	0.511
NN	0.481	0.489	0.497	0.478	0.463	0.479	0.506	0.511	**0.521**
VN	0.357	0.377	0.403	0.456	0.464	**0.483**	0.432	0.436	0.45
Multi	0.306	0.316	0.318	0.388	0.395	**0.401**	0.368	0.38	0.388
AN2	0.389	0.417	0.426	0.532	**0.545**	0.544	0.474	0.490	0.490
NN2	0.374	0.369	0.397	0.436	0.397	0.413	0.417	0.42	**0.445**
VN2	0.388	0.391	0.421	0.525	0.530	**0.552**	0.494	0.486	0.502

As shown in Tables V and VI, the word paraphrasing method is better than the retrofitting method both in the word similarity task and in the most similar words result. The both methods exceed the standard word representations by a large margin. Augmentation has amazing impact on word similarity tasks. Still, the question remains: Can augmented word representation improve the performance of the composition model?

Table VII shows the performance of Additive model with standard word representation and augmented word representation. The augmented word vectors perform better than the standard word vectors on most of the phrase similarity datasets, except that augmented vectors with paraphrasing method perform worse on the NN similarity dataset. This goes against the common sense: word representations with richer semantics should be better at representing phrase meaning by simply adding component word vectors. When checking the testing dataset for NN, we have found that the annotation by Mitchell and Lapata [2010] for bigram pairs is one that leans more toward capturing topical similarity. For example, *television set* and *television program* had the highest score in the NN set (based on the average annotation score). However, these two phrases are more similar in topic than in meaning and should not get the highest score. This is consistent with the result of the experiments by Wieting et al. [2015], who re-annotated the bigram phrase similarity data. Using the re-annotated phrase similarity (AN2, NN2, VN2) from Wieting et al. [2015], we

Table VIII. Correlation Coefficients of Baseline Model Predictions with Subject Similarity Ratings on Phrase Similarity Tasks

Model	AN	NN	VN	Multi
Mitchell and Lapata [2010]	0.46	**0.49**	0.38	–
Hashimoto et al. [2014]	0.49	0.45	0.46	–
Word overlap	–	–	–	0.26
Wieting et al. [2015]	**0.51**	0.40	**0.50**	**0.40**

Table IX. Correlation Coefficients of Model Predictions with Subject Similarity Ratings on Chinese Word Similarity Tasks

	50	100	200	Para50	Para100	Para200	Ret50	Ret100	Ret200
240	0.534	0.569	0.558	0.58	**0.599**	0.586	0.569	0.597	0.586
297	0.583	0.581	0.588	0.61	0.615	0.622	0.625	0.628	**0.637**

Table X. Correlation Coefficients of Additive Model Predictions with Subject Similarity Ratings on Chinese Phrase Similarity Tasks

	50	100	200	Para50	Para100	Para200	Ret50	Ret100	Ret200
AN	0.655	0.681	0.672	0.669	0.687	0.69	0.677	**0.696**	0.691
NN	0.611	0.614	0.608	0.618	0.621	0.628	0.63	**0.635**	0.632
VN	0.577	0.594	0.597	0.598	0.614	**0.624**	0.605	0.613	0.616
Multi	0.548	0.572	0.594	0.58	0.605	**0.639**	0.573	0.595	0.614

report the results in the last row of Table VII. We can see that, now, both augmented vectors perform better than the standard vectors on NN2.[16]

To better express the effectiveness of augmented word representations, we list the baseline results and state-of-the-art results on phrase similarity tasks in Table VIII.

From Table VIII, we can find that most of the results with augmented word vectors shown in Table VII are comparable to or outperform the state-of-the-art results.

- **Chinese**

We used the Chinese word similarity datasets (word similarity dataset 297 and word relatedness datasets 240) to test the quality of Chinese word vectors. From Table IX, we can see similar results as those obtained for English. Specifically, the augmented word vectors perform better on word similarity tasks and the word vectors with higher dimensions are better at capturing the semantic meanings of words.

To better reflect the effect of augmented word representations, we listed the results of different word vectors with the Additive model on phrase similarity tasks in Table X.

In Table X, we can still see the improved results with two augmented word vectors, but the advantage is not as obvious as that found in English. One possible reason is that our training data for the augmentation methods in Chinese is smaller than the English training data. The results in Table IX indirectly prove this hypothesis because the effect of augmentation on the word similarity task is limited when the training data is smaller.

To sum up, the augmented word representations can help the word representation to capture more semantic information. Moreover, the augmented word representations improve the performance of composition model on bigram and multi-word phrase similarity task in both English and Chinese.

[16] In the following experiments, we show results on both annotation datasets because of their advantages. Datasets of Mitchell and Lapata [2010] is more widely used and provided more testing data with more human annotations.

Table XI. Correlation Coefficients of Model Predictions by Pseudo-Word Training with Subject Similarity Ratings on Word Similarity Tasks

	English				Chinese	
Dim	353	Sim	Rel	Sim999	240	297
50	0.657	0.720	0.585	0.259	0.534	0.599
100	0.694	0.757	0.615	0.298	0.564	0.606
200	0.715	0.767	0.642	0.348	0.554	0.601

4.2. Pseudo-Word Training Data vs. Word Paraphrasing Training Data

To compare the performance in two types of training data, first, we explore the effect of pseudo-word training data on word similarity task. Then, we discuss whether pseudo-word training data degrades the performance of the composition model.

The pseudo-word training data take the form {adjective1 noun1, adjective1-noun1}, where “adjective1-noun1” is adopted as a pseudo-word when preprocessing the corpus. We call this corpus the pseudo-word corpus. The vector of “adjective1-noun1” is calculated by estimating the surrounding contexts. This method ignores the constituent words (adjective1 and noun1) when learning representations of “adjective1-noun1.” For example, take an extreme case where the words *neural* and *network* appear only in the phrase *neural network*. First, we rewrite the words *neural network* in the corpus as a unity: *neural-network*. Then, we learn the word representation using the pseudo-word corpus. Obviously, this method cannot learn word representations of *neural* and *network*.

This is a special case that almost never happens in large corpora. We care more about two questions: first, “How much does pseudo-word training affect the word representation performance?” and second, “Will pseudo-word training affect the performance of composition model?” The following experiments will try to answer these questions.

- **Effect of Training with Pseudo-Word Corpus on Word Similarity Task**

Table XI shows the performance of word representations trained with the pseudo-word corpus on word similarity tasks. Compared to Table V, in English, the pseudo-word training degrades the word representation’s performance on word similarity tasks. However, on the Chinese word similarity, 297 dataset, the performance improves with word representations learned by pseudo-word training data. One possible reason is that the words in 297 rarely appear in the pseudo-word training data. To check this hypothesis, we divided the number of words in the word test data by the number of words in the pseudo-word training data. The result on datasets 297, 240 and the phrase testing data are 0.46, 0.55, and 0.88, respectively. These results verify our hypothesis and indicate that pseudo-word training phrase should have more effect on the phrase composition model. Another reason is that the pseudo-word training set in Chinese is relatively small; consequently, the effects of pseudo-words can be ignored in a large training corpus.[17].

- **Effect of Training with Pseudo-Word Corpus on Phrase Similarity Task**

Table XII compares the results of word representations using the ordinary corpus (Add) and the pseudo-word training corpus (PAdd) with the Additive model on phrase similarity tasks.

As shown in Table XII, the word representations learned from the pseudo-word training corpus degrades the performance of phrase similarity tasks. Even in Chinese, the gap is obvious. The reason is that pseudo-word training degrades the word representation’s performance and further influence the performance of Additive model.

[17] We verified this hypothesis by decreasing number of pseudo-word training set in English.

Table XII. Correlation Coefficients of Additive Model Predictions (Based on Word Representation Learned from Ordinary Corpus and Pseudo-Word Training Corpus) with Subject Similarity Ratings on Bigram Phrase Similarity Tasks

		English			Chinese		
	Dim	AN	NN	VN	AN	NN	VN
Add	50	0.436	0.481	0.357	0.655	0.611	0.577
	100	0.471	0.489	0.377	**0.681**	**0.614**	0.594
	200	**0.464**	**0.497**	**0.403**	0.672	0.608	**0.597**
PAdd	50	0.402	0.451	0.357	0.637	0.607	0.568
	100	0.446	0.464	0.376	0.647	0.608	0.581
	200	0.447	0.475	**0.403**	0.65	0.593	0.585

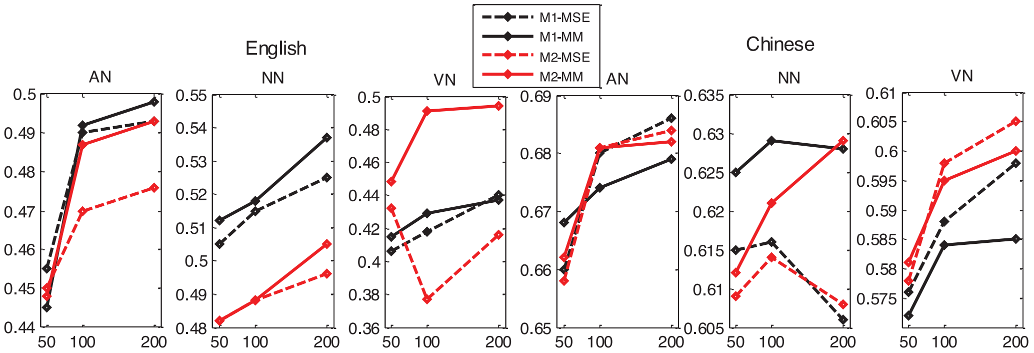

Fig. 1. The effect of training word vectors and phrase vectors in separate trials. M1 is the baseline Matrix composition model which trains two types of vectors in one model while the M2 model trains them separately. We also show the combination of the Matrix model with two objective functions: mean square error and max-margin. The horizontal coordinate represents the dimension of word vectors. The vertical coordinate represents the correlation coefficients of model predictions with subject similarity ratings.

Naturally, a question occurs: would the performance be better if we use word representations learned from ordinary corpus and phrase representations learned from the pseudo-word corpus? The result is unpredictable for the following reason.

If we use word vectors and phrase vectors in different training trials, we would obtain the vectors in different vector spaces and, consequently, the learned composition function would have to include transformation and composition. Thus the effect of better word representation learned from the ordinary corpus might be canceled out. We tested this hypothesis with a simple matrix composition function on phrase similarity datasets and plot the results in Figure 1.

As shown in Figure 1, using phrase vectors and word vectors from two trials does not improve the results. Moreover, almost all performances are worse except for verb noun phrases. A possible reason for this exception is that better word representation is more important than space transformation with smaller available training data (see Table IV for detailed data size). Another reason is VN phrase structure is closer to sentence than AN phrase and NN phrase; therefore VN phrase show different properties. We leave a more detailed investigation of VN phrase as future work.

To sum up, in the absence of high-quality training paraphrases, learning the phrase vectors directly from a large corpus is a good choice as a training objective for the bigram phrase task. Moreover, pseudo-word training is particularly helpful to languages without paraphrase resources.

Table XIII. Correlation Coefficients of Model (Add, RecNN, and Matrix) Predictions (Based on Pair Training Data) with Subject Similarity Ratings on Bigram Phrase Similarity Tasks

	English						Chinese					
	AN		NN		VN		AN		NN		VN	
	MSE	MM	MSE	MM	MSE	MM	MSE	MM	MSE	MM	MSE	MM
Add	0.464	0.464	0.497	**0.497**	0.403	0.403	0.672	0.672	0.608	0.608	0.597	0.597
A-Add	0.523	0.523	0.479	0.479	0.483	0.483	0.69	0.69	0.628	0.628	0.624	0.624
Add2	0.538	0.474	0.515	**0.497**	0.471	0.460	0.699	0.693	0.632	0.623	0.610	0.607
A-Add2	**0.557**	0.523	0.495	0.479	**0.502**	**0.491**	**0.706**	**0.700**	**0.644**	**0.641**	**0.634**	0.628
Rec	0.463	0.460	0.463	0.487	0.436	0.402	0.687	0.691	0.617	0.619	0.6	0.620
A-Rec	0.524	**0.524**	**0.524**	0.479	0.484	0.484	0.696	0.694	0.636	0.637	0.624	0.629
M	0.464	0.456	0.488	0.488	0.454	0.40	0.688	0.686	0.616	0.621	0.605	0.620
A-M	0.523	0.523	0.478	0.478	0.484	0.484	0.699	0.690	0.635	0.638	0.624	**0.631**

We only present word dimension of 200 for clarity.

4.3. Effects of Different Combination of Composition and Objective Functions

Even though Additive model represents phrase meaning simply by averaging representations of constituent words, it is a strong baseline in various textual similarity tasks [Pham et al. 2015; Wieting et al. 2016a, 2016b]. However, one possible reason is that the existing test datasets cannot reflect the function of syntax structure and word order in phrase and sentences.

In contrast to Matrix model, RecNN and RNN model take word order into consideration. But it is unclear whether word order is essential in understanding phrase meanings. Iyyer et al. [2015] argued that the deep unordered composition model makes similar errors with syntactically aware models. They argued that transforming the input is more important than tailoring a network to incorporate word order and syntax. We found similar results when comparing the RecNN model and the Matrix model. Specifically, we compared the performance of RecNN in Wieting et al. [2015] to the Matrix model using the same word vectors. The Matrix model achieved a correlation score of 0.46, better than the 0.40 achieved by RecNN in the multi-word phrase similarity task. We think the reason is the tree-building errors because RecNN model needs to compose words in a binary parse tree. LSTM[18] is a powerful model and achieves the state-of-the-art results on a number of textual similarity tasks. Unfortunately, in the multi-word phrase experiment, we found it performs poorly on English task and only slightly better than the Additive model in Chinese.

The following sections discuss the quantitative comparison made among different compositions and objective functions and their effects on phrase similarity tasks.

- **Effects of Composition Function**

For the bigram similarity task, we used Additive (update word vectors or not), RecNN and Matrix composition functions (In this article, we represent these models as Add, Add2, Rec and M. A- means models with augmented word vectors). Their performances are shown in Table XIII (Table VIII shows the results of baseline models). We found that all composition models with augmented word vectors outperform the state-of-the-art results. An exception is NN phrase because of the annotation dataset (see detailed reasons in Subsection 4.1). To capture the effects of composition function clearly, we show results on annotation dataset from Wieting et al. [2015] in Table XIV. Moreover, the Additive model which updates augmented word vectors achieves best results in most datasets, outperforming the state-of-the-art by a large margin. This is consistent with the finding of Wieting et al. [2015] which shows the powerfulness of updating augmented word vectors in bigram phrase similarity tasks.

[18] We use LSTM (which is one special type of RNN) in the multi-word task, for we find LSTM outperform RNN steadily in this task.

Table XIV. Correlation Coefficients of Model (Add, RecNN, and Matrix) Predictions with Subject Similarity Ratings on English Bigram Phrase Similarity Tasks (with Datasets from Wieting et al. [2015])

	AN		NN		VN	
	MSE	MM	MSE	MM	MSE	MM
Add	0.426	0.426	0.397	0.397	0.421	0.421
A-Add	0.544	0.544	0.413	0.413	0.552	0.552
Add2	0.568	0.467	0.432	0.397	0.575	0.458
A-Add2	**0.615**	**0.546**	**0.450**	0.414	0.60	**0.556**
Rec	0.425	0.543	0.395	0.395	0.552	0.420
A-Rec	0.544	0.544	0.415	**0.415**	0.572	0.552
M	0.425	**0.546**	0.396	0.396	0.556	0.421
A-M	0.544	0.544	0.414	0.414	**0.605**	0.551

Table XV. Correlation Coefficients (and Standard Deviation) of Model (Add, LSTM, Matrix) Predictions with Subject Similarity Ratings on Multi-Word Phrase Similarity Tasks

	English		Chinese	
	MSE	MM	MSE	MM
Add	0.318	0.318	0.594	0.594
A-Add	0.401	0.401	0.639	0.639
Add2	0.318(0.0)	0.481(0.003)	0.638(0.0)	0.622(0.003)
A-Add2	0.437(0.0)	0.506(0.003)	0.677(0.001)	0.653(0.001)
LSTM	0.276(0.001)	0.363(0.002)	0.584(0.0)	0.626(0.0)
A-LSTM	0.343(0.0)	0.410(0.001)	0.621(0.0)	0.652(0.0)
Matrix	0.460(0.002)	0.460(0.02)	0.69(0.02)	0.679(0.02)
A-Matrix	**0.522(0.001)**	**0.509(0.002)**	**0.709(0.0)**	**0.685(0.02)**

For the multi-word phrase similarity task, we used Additive (update word vectors or not), LSTM and Matrix composition functions. Table XV shows their performances with MSE and max-margin objective functions, respectively. As the table shows, the Matrix model based on standard and augmented word representations achieves best results in most datasets. LSTM model, which is reported to achieve the state-of-the-art results on a number of textual similarity tasks, do not show advantage in multi-word phrase similarity task in both languages. Even though achieving best results in bigram phrase similarity task, Additive model with updating augmented word vectors perform worse than the Matrix model in multi-word phrase similarity task. Moreover, the standard deviations in Table XV show the robustness of the models.

- **Effects of Objective Function**

Do different objective functions have an impact on the performance of composition model? To answer this question, we compared two objective functions (MSE and max-margin) on the phrase similarity tasks. The results are shown in Tables XIII, XIV and XV. We have the following observations.

There is no definitive answer for the question that which objective function is better. On the bigram phrase similarity task, the performance of max-margin and MSE are almost the same. An exception is Additive model with updating word representations which shows superiority of MSE over max-margin. On the multi-word phrase similarity task, LSTM model with max-margin function performs better than with MSE function in both languages. In Chinese, Additive and Matrix model with MSE function achieve better results. Results on English datasets are not as clear. Additive model prefer max-margin function and Matrix model prefer MSE objective function. In general, the max-margin objective requires more computation than MSE, and it is less stable

Table XVI. Correlation Coefficients of Additive Model (Weighted by L2 Norm or Not) Prediction with Subject Similarity Ratings on Multi-Word Phrase Similarity Tasks

	English		Chinese	
	MSE	MM	MSE	MM
Add+norm(Add)	0.318	0.273	0.634	0.607
A-Add+norm(Add)	**0.443**	0.420	0.674	**0.648**
Add+norm(LSTM)	0.318	0.318	0.594	0.594
A-Add+norm(LSTM)	0.401	0.401	0.639	0.639
Add+norm(Matrix)	0.310	0.307	0.637	0.617
A-Add+norm(Matrix)	0.383	**0.432**	**0.678**	0.642

because of randomly selecting negative samples. However, LSTM model with max-margin function is the first choice. In Chinese, MSE objective function is preferred.

- **How Do Composition Models Work?**

To delve deeper into how composition models work, we take multi-phrase similarity task as an example and compare the different composition models on two aspects. One aspect focuses on what composition models learned to make it better than the baseline Additive model. The other aspect is that what factors in the testing data differentiate the performance of these models.

For the first aspect, there are two possibilities that make the composition model more powerful. One is that the composition model can learn suitable word representations for phrase similarity tasks[19]. Another is that the composition function is more powerful at capturing complex relations. To test our first hypothesis, we calculate the L2 norm of word representations after training with Additive, Matrix and LSTM models with two objective functions. A higher L2 norm value means the more important the word is. After obtaining the L2 norm for each word, we test the multi-word phrase similarity task using the word vectors before training and weighted them by the L2 norm from Additive, Matrix and LSTM models, respectively. Table XVI shows the correlation coefficient results of Additive model weighted by L2 norm from different composition models with subject similarity ratings.

From Chinese results in Table XVI and baseline results in Table XV, we can see that Additive and Matrix model with enhanced word embedding or not benefit from the learned L2 norm weights. Both models attached higher L2 norms to content words, which indicate that these two models can learn more suitable word representations for the phrase similarity task. Take the Additive model with MSE function for example, the five highest L2 norms words are “世界杯赛(The world cup)”, “汉朝(The han dynasty)”, “溜冰(skating)”, “明代(The Ming dynasty)”, “居所(residence)” which are all content words, and the five lowest L2 norms words are “了(an auxiliary word in Chinese)”, “是(is)”, “和(and)”, “等(etc.)”, “的(of, the)” which are function words. LSTM model with max-margin function, which shows improvement in phrase similarity tasks in Table XV, do not attach higher L2 norms for content words in both Chinese and English as shown in Table XVI. Results in English are beyond our expectations. Improvements of the Additive and the Matrix model are not from L2 norms. Moreover, the L2 norm learned by augmented Matrix model with MSE function degraded the performance of Additive model, which indicate the importance of the Matrix composition function.

[19] Wieting et al. [2016a] and Pham et al. [2015] proved that composition function of Addition and updating word embeddings method actually learn large L2 norm for content words.

Table XVII. Examples of Testing Sentences and Corresponding Factors

Sent1	Sent2	Edit distance	Length ratio	Structure distance	Similarity score
a_DT solution_NN to_TO the_DT conflict_NN	the_DT resolution_NN of_IN the_DT conflict_NN	1.4	1.0	0.4	4.8
‘re_VB not_RB gon_VBG na_TO make_VB it_PRP	going_VBG down_RB	9	0.33	7	2.2
全国_JJ 导弹_NN 防御_NN 系统_NN	国家_NN 导弹_NN 防卫_NN 系统_NN	1.5	1	0.5	3.0
人工_AD 便宜_JJ	低廉_VA 的_DEC 劳动_NN 成本_NN	9.5	0.5	4.5	3

Wieting et al. [2016a] showed that about half and more improvement of their model over initial vectors is due to the weighting tokens by their importance in sentence similarity tasks. In this article, we explore whether the conclusion is valid for different composition models in the phrase similarity task. We compare additive model, Matrix model and LSTM model with MSE and max-margin objective function in both English and Chinese. We found that both composition function and learned L2 norm account for performance of composition models. In some cases, composition models do not learn higher L2 norm for content words but can still achieve better performance.

For the second aspect, we found that the phrase edit distance ratio,[20] the length ratio, and the structure distance ratio all have large impacts on the performances of these three models. The length ratio is the length of the smaller phrase divided by the length of longer phrase. We calculate the structure distance ratio by first replacing words with parts-of-speech tags in the phrase pairs and then use the same calculation as the word distance ratio to obtain the structure distances in the phrase pairs.

We show some examples of the testing data and their corresponding factor values in Table XVII.

We calculated the performances of Additive model, LSTM model, and Matrix model with two objective functions for these different factors. The results for English and Chinese tasks are shown in Figures 2 and 3, respectively. We divided the testing data into three parts equally according to every factor value. Then, we calculated the correlation coefficients of model predictions with subject similarity ratings on each part of the data. For example, the value of edit distance ranges from 1 to 19; therefore, the three parts are 1–7, 7–13, and 13–19. The sentences whose edit distance scores are within 1–7, 7–13, and 13–19 were placed into the first, second, and third parts, respectively. We calculated the correlation scores between the model's predictions on these parts with the subject similarity ratings. In this manner, we obtained three correlation coefficients for each model factor as plotted in Figures 2 and 3.

From Figure 2, we can make the following observations. For English datasets, all models' performance is highly correlated to the edit distance ratio, the length ratio, and the structure distance ratio. Specifically, all models perform better on phrases with similar lengths, higher word overlap and similar syntax structure. Moreover, LSTM model is more robust than Matrix model and Additive model as word overlap change.

For Chinese tasks shown in Figure 3, all models' performance are highly correlated to the edit distance ratio, the length ratio, and the structure distance ratio as was found in English, but the difference between models is not as obvious. The variation of

[20] The phrase edit distance ratio is calculated by first concatenating words in the phrase and calculating the edit distance, then. dividing the results by the minimum phrase length. The same procedure is performed for structure distance ratio.

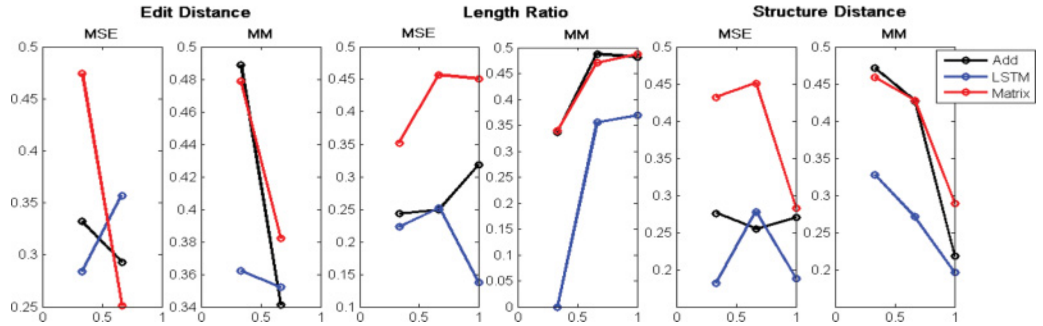

Fig. 2. The performance of different composition models with the different text factors in English. The black, red, and blue lines show the performances of the composition function for the Additive, LSTM and Matrix models, respectively. The horizontal coordinate represents the factor value ratio. We chose 1/3, 2/3 and 1 to plot the performance of the different composition models with each part of data (for edit distance, there are only nine examples whose edit distance is more than 2/3 of the overall, so we ignored those examples). The vertical coordinate represents the correlation coefficients of the model predictions with the subject similarity ratings.

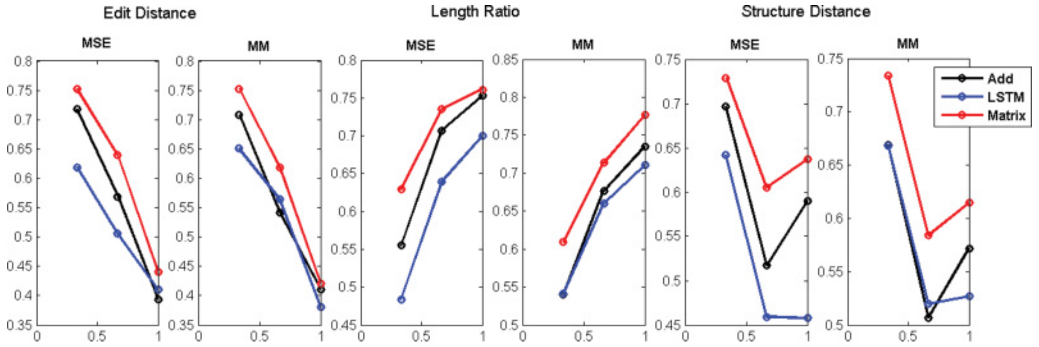

Fig. 3. The performance of different composition models with different text factors in Chinese. The black, red, and blue lines show the performances of the composition function for the Additive, LSTM and Matrix models, respectively. The horizontal coordinate represents the factor value ratio. We chose 1/3, 2/3 and 1 to plot the performances of different composition models with each part of data. The vertical coordinate represents the correlation coefficients of the model predictions with the subject similarity ratings.

different models with Max-margin and MSE objective functions in Chinese is smaller than in English. The following two reasons can account for the difference between results in these two languages. One is linguistic variations. Chinese is a language that attaches more importance to semantic factors than to syntax. The other is the different construction between the training datasets. We extract Chunks in parallel corpora to construct linguistically valid Chinese multi-phrase pairs. Without this constraint, data in multi-phrase similarity datasets in English is more a fragments of text than a phrase which contained function words in a large scale.

5. CONCLUSIONS AND FUTURE WORK

In this article, we reported the results of a large-scale comparison and evaluation of different composition models on phrase similarity tasks. Previous works focus on the composition function; however, our findings indicate that other components in the composition model (especially word representation) make a critical difference in phrase representation. We provide some setting suggestions in the discussions of the experiments. We believe applications that use phrase composition models will benefit from using the suggested settings. We have also introduced two datasets to evaluate composition models of Chinese short phrases, which could act as a reference for related fields. In addition, we hope that our results promote research into Chinese phrase representation. We summarize our main findings and suggestions as follows:

—Augmented word representations have important impacts on the performance of the composition models. To obtain a good representation for short phrases, more attention should be paid to achieving good word representations.
—In the absence of high-quality training paraphrases, the phrase vectors learned in pseudo-word training corpus are good enough to be the gold training output of a composition model in the bigram phrase similarity task.
—The importance attached to the key words and the composition function of the composition model both account for the performance gain. Moreover, the models that consider word order do not show an advantage in phrase similarity tasks.
—The performance of combinations of different composition and objective functions is sensitive to language and the type of training data. This finding indicates that it is necessary to take the specific task to be performed into consideration.

Composing words into phrases is complex—even with bigram phrases—because many ambiguities and multiple relations exist between the words. For example, the noun-noun phrase *war crime* can be interpreted as a crime during war, or consider *security guarantees* which means to provide a guarantee of security. The relationship between two nouns in bigram phrases is flexible and may take many different roles. Therefore, we should not assign the same composition function for all bigram phrases to represent their meanings. We have to develop new composition functions related to phrase context and word relations within the phrase. Additionally, the composition models proposed up the present lack prior world knowledge and may require assistance from other resources such as knowledge bases. Another direction to explore is how to use composition models for short phrases in sentence-level tasks. In contrast to short phrases, sentences contain more context and structural information, making them a better testing ground for developing semantic representation models.

ACKNOWLEDGMENTS

The authors would like to thank the three anonymous reviewers for their helpful comments and suggestions.

REFERENCES

Marco Baroni and Roberto Zamparelli. 2010. Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*. 1183–1193.

Marco Baroni, Georgiana Dinu, and German Kruszewski. 2014. Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*. 238–247.

Yoshua Bengio, Rejean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research* 3, 1137–1155.

William Blacoe and Mirella Lapata. 2012. A comparison of vector-based representations for semantic composition. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. 546–556.

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

Danushka Bollegala, Alsuhaibani Mohammed, Takanori Maehara, and Ken-ichi Kawarabayashi. 2016. Joint word representation learning using a corpus and a semantic lexicon. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*. 2690–2696.

Antoine Bride, Tim Van de Cruys, and Nicholas Asher. 2015. A generalisation of lexical functions for composition in distributional semantics. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*. 281–291.

Wanxiang Che, Zhenghua Li, and Ting Liu. 2010. Ltp: A chinese language technology platform. In *Coling2010: Demonstrations*. Beijing, China, 13–16.

Callison-Burch Chris, Philipp Koehn, and Miles Osborne. 2006. Improved statistical machine translation using paraphrases. In *Proceedings of the Main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*. 17–24.

Kai-min K. Chang, Vladimir L. Cherkassky, Tom M. Mitchell, and Marcel Adam Just. 2009. Quantitative modeling of the neural representation of adjective-noun phrases to account for fMRI activation. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing*. 638–646.

Kai-min K. Chang. 2011. *Quantitative Modeling of the Neural Representation of Nouns and Phrases.* Doctoral dissertation, University of Trento.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning.* 160–167.

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

Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science* 41, 6, 391.

Corina Dima. 2015. Reverse-engineering language: A study on the semantic compositionality of german compounds. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 1637–1642.

Georgiana Dinu, Nahia The Pham, and Marco Baroni. 2013. General estimation and evaluation of compositional distributional semantic models. In *Proceedings of the ACL Workshop on Continuous Vector Space Models and Their Compositionality*, 50–58.

J. Duchi, E. Hazan, and Y. Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* 12, 2121–2159.

Manaal Faruqui, Jesse Dodge, Sujay Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. 2015. Retrofitting word vectors to semantic lexicons. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.* 1606–1615

Ruiji Fu, Jiang Guo, Bing Qin, Wanxiang Che, Haifeng Wang, and Ting Liu. 2014. Learning semantic hierarchies via word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*. 1199–1209.

Alona Fyshe. 2015. *Corpora and Cognition: The Semantic Composition of Adjectives and Nouns in the Human Brain.* Doctoral dissertation, Air Force Research Laboratory.

Edward Grefenstette and Mehrnoosh Sadrzadeh. 2011. Experimental support for a categorical compositional distributional model of meaning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. 1394–1404.

Edward Grefenstette, Georgiana Dinu, Yao-Zhong Zhang, Mehrnoosh Sadrzadeh, and Marco Baroni. 2013. Multi-step regression learning for compositional distributional semantics. *arXiv preprint arXiv:1301.6939*.

Emiliano Guevara. 2010. A regression model of adjective-noun compositionality in distributional semantics. In *Proceedings of the 2010 Workshop on GEometrical Models of Natural Language Semantics*. 33–37.

Abhijeet Gupta, Gemma Boleda, Marco Baroni, and Sebastian Pado. 2015. Distributional vectors encode referential attributes. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing.* 12–21.

Kazuma Hashimoto, Pontus Stenetorp, Makoto Miwa, and Yoshimasa Tsuruoka. 2014. Jointly learning word representations and composition functions using predicate-argument structures. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, 1544–1555.

Felix Hill, Kyunghyun Cho, and Anna Korhonen. 2016. Learning distributed representations of sentences from unlabelled data. In *Proceedings of the 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.* 1367–1377.

Eric H. Huang, Richard Socher, Christopher D. Manning, and Andrew Y. Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*. 873–882.

Miki Iwai, Takashi Ninomiya, and Kyo Kageura. 2015. Acquiring distributed representations for verb-object pairs by using word2vec. In *Proceedings of the 29th Pacific Asia Conference on Language, Information and Computation*. 328–336.

Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daume III. 2015. Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*. 1681–1691.

Douwe Kiela, Felix Hill, and Stephen Clark. 2015. Specializing word embeddings for similarity or relatedness. In *Proceeding of the 2015 Conference on Empirical Methods in Natural Language Processing*. 2044–2048.

Arne Köhn. 2015. What's in an embedding? Analyzing word embeddings through multilingual evaluation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 2067–2073.

Omer Levy and Yoav Goldberg. 2014. Neural word embedding as implicit matrix factorization. In *Advances in Neural Information Processing Systems.* 2177–2185.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit. ACL 2014 Demo Session.

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

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

Dmitrijs Milajevs, Dimitri Kartsaklis, Mehrnoosh Sadrzadeh, and Matthew Purver. 2014. Evaluating neural word representations in tensor-based compositional settings. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*. 708–719.

Jeff Mitchell and Mirella Lapata. 2008. Vector-based models of semantic composition. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*. 236–244.

Jeff Mitchell and Mirella Lapata. 2010. Composition in distributional models of semantics. *Cognitive Science* 34, 8, 1388–1429.

Jonas Mueller and Aditya Thyagarajan. 2016. Siamese recurrent architectures for learning sentence similarity. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*.

Donald A. Norman. 1972. Memory, knowledge, and the answering of questions[J]. *Contemporary Issues in Cognitive Psychology the Loyola Symposium.*

Nghia The Pham, Germán Kruszewski, Angeliki Lazaridou, and Marco Baroni 2015. Jointly optimizing word representations for lexical and sentential tasks with the C-PHRASE model. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*. 971–981.

Michael Roth and Kristian Woodsend. 2014. Composition of word representations improves semantic role labelling. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*. 407–413.

Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. 2011. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing.* 151–161.

Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Ng. 2012. 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*. 1201–1211.

Richard Socher, Alex Perelygin, Jean Y. Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing.* 1631, 1642.

Richard Socher, Andrej Karpathy, Quoc V. Le, Christopher D. Manning, and Andrew Y. Ng. 2014. Grounded compositional semantics for finding and describing images with sentences. *Transactions of the Association for Computational Linguistics* 2, 207–218.

Kaveh Taghipour and Hwee Tou Ng. 2015. Semi-supervised word sense disambiguation using word embeddings in general and specific domains. In *Proceedings of the 2015 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. 314–323.

Ran Tian, Naoaki Okazaki, and Kentaro Inui. 2015. The mechanism of additive composition. *arXiv preprint arXiv:1511.08407*.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010, July. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. 384–394.

Tim Van de Cruys, Thierry Poibeau, and Anna Korhonen. 2013. A tensor-based factorization model of semantic compositionality. In *Proceedings of the Conference of the North American Chapter of the Association of Computational Linguistics*. 1142–1151.

John Wieting, Mohit Bansal, Kevin Gimpel, Karen Livescu, and Dan Roth. 2015. From paraphrase database to compositional paraphrase model and back. *arXiv preprint arXiv:1506.03487*.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2016a. Towards universal paraphrastic sentence embeddings. In *Proceedings of the 4th International Conference on Learning Representations*.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2016b. CHARAGRAM: Embedding words and sentences via character *n*-grams. *arXiv preprint arXiv:1607.02789*.

Mo Yu and Mark Dredze. 2014. Improving lexical embeddings with semantic knowledge. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*. 545–550.

Mo Yu and Mark Dredze. 2015. Learning composition models for phrase embeddings. *Transactions of the Association for Computational Linguistics* 3, 227–242.

Fabio M. Zanzotto, Ioannis Korkontzelos, Francesca Fallucchi, and Suresh Manandhar. 2010. Estimating linear models for compositional distributional semantics. In *Proceedings of the 23rd International Conference on Computational Linguistics*. 1263–1271.

Yu Zhao, Zhiyuan Liu, and Maosong Sun. 2015. Phrase type sensitive tensor indexing model for semantic composition. In *Proceedings of AAAI*. 2195–2202.

Received May 2016; revised August 2016; accepted October 2016