Empirical Exploring Word-Character Relationship for Chinese Sentence Representation

SHAONAN WANG and JIAJUN ZHANG, 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

This article addresses the problem of learning compositional Chinese sentence representations, which represent the meaning of a sentence by composing the meanings of its constituent words. In contrast to English, a Chinese word is composed of characters, which contain rich semantic information. However, this information has not been fully exploited by existing methods. In this work, we introduce a novel, mixed character-word architecture to improve the Chinese sentence representations by utilizing rich semantic information of inner-word characters. We propose two novel strategies to reach this purpose. The first one is to use a mask gate on characters, learning the relation among characters in a word. The second one is to use a max-pooling operation on words to adaptively find the optimal mixture of the atomic and compositional word representations. Finally, the proposed architecture is applied to various sentence composition models, which achieves substantial performance gains over baseline models on sentence similarity task. To further verify the generalization ability of our model, we employ the learned sentence representations as features in sentence classification task, question classification task, and sentence entailment task. Results have shown that the proposed mixed character-word sentence representation models outperform both the character-based and word-based models.

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

Additional Key Words and Phrases: Sentence representation, composition model, inner-word character, mixed character-word representation, mask gate, max pooling

ACM Reference format:
Shaonan Wang, Jiajun Zhang, and Chengqing Zong. 2018. Empirical Exploring Word-Character Relationship for Chinese Sentence Representation. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.* 17, 3, Article 14 (January 2018), 18 pages.
https://doi.org/10.1145/3156778

1 INTRODUCTION

To understand the meaning of a sentence is a prerequisite to solve many problems of natural language processing, such as question answering, machine translation, and human-computer interaction. Obviously, this requires a good representation of the meaning of a sentence. Recently,

Authors' addresses: S. Wang, J. Zhang, and C. Zong (corresponding author), Intelligence Building, 498 No. 95, Zhongguancun East Road, Haidian District, Beijing, 100190, China; emails: {shaonan.wang, jjzhang, cqzong}@nlpr.ia.ac.cn.

https://doi.org/10.1145/3156778

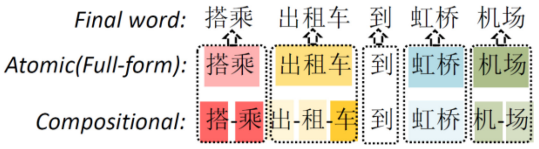

Fig. 1. An example sentence that consists of five words as "搭乘(take) 出租车(taxi) 到(to) 虹桥(Hongqiao) 机场(airport)." Most of these words are transparent, namely, the word "搭乘(take)" consists of the characters "搭(take)" and "乘(ride)," the word "出租车(taxi)" constitutes the characters "出(out)," "租(rent)," and "车(car)," and the word "机场(airport)" is composed of the characters "机(machine)" and "场(field)." The word "虹桥(Hongqiao)" is a place name, which is non-transparent. The color depth represents (1) contributions of each character to the compositional word meaning and (2) contributions of the atomic and compositional word to the final word meaning. The deeper color means more contributions.

neural network–based sentence representation models have gained a significant amount of research attention and show advantages in representing sentence meaning [2, 11, 12, 16, 17, 19, 28, 29, 32, 33, 34]. However, despite the fact that inner-word characters are important for representing word meaning, the most existing distributed sentence representations are usually built from representations of its constituent word sequences, ignoring rich semantic information in characters.

In this article, we situate our investigation in the context of Chinese, a language in which the majority of words are transparent, i.e., we can understand the meaning of the word if we know the meaning of its constituent characters. Li [18] analyzed the semantic transparency of 33,000 Chinese double and triple-syllable words in the Contemporary Chinese Dictionary. She discovered that 93% of these words are comparably (70%) or completely (30%) transparent, which means that the component characters play a vital role in understanding the word meaning. The high proportion of transparent words makes it necessary to explore how to take full advantage of information on a character level.

As illustrated in Figure 1, characters in Chinese words express two characteristics: (1) Each character in a word contributes differently to the compositional word meaning [35] such as the word "出租车(taxi)." The first two characters "出租(rent)" are descriptive modifiers of the last character "车(car)," and make the last character play the most important role in expressing word meaning. (2) The atomic (which ignore inner characters) and compositional representations contribute differently to different types of words [20]. For instance, the meaning of "机场(airport)," a low-frequency word, can be better expressed by the compositional word representation, while the non-transparent word "虹桥(Hongqiao)" is better expressed by the atomic word representation.

In this article, we explore inner-word characters to learn generic sentence representations and propose a mixed character-word architecture, which can be integrated into various sentence composition models from simple word averaging to a Long Short-Term Memory (LSTM) network model. In the proposed architecture, a mask gate is employed to model the relation among characters in a word, and a pooling mechanism is leveraged to model the contributions of the atomic and compositional word embeddings to the final word representations. As Chinese characters are much more ambiguous compared with words, we further explore two character disambiguation methods to verify if multi-prototype character embeddings have a positive effect on learning sentence representations. Furthermore, to evaluate effectiveness, we compare our models with both character-based and word-based models and conduct experiments on applying the learned sentence representations to the tasks of sentence similarity, sentence classification, question classification, and sentence entailment. Experimental results have demonstrated the superiority of our method. In addition, as there are no publicly available Chinese sentence similarity datasets, we build a dataset to directly test the quality of sentence representations. *The data and code are*

released on Github: https://github.com/wangshaonan/Chinese-sentence-representation with the hope that they can serve as a baseline and promote research on Chinese sentence representation.

2 RELATED WORK

Learning meaningful sentence representations, as the first step toward the goal of language understanding, has received wide research attention. Recently, neural network–based methods have shown an advantage in learning task-specific sentence representations [2, 4, 24, 28, 37] and generic sentence representations [8, 13, 16, 17, 27, 32, 33, 34]. Our work falls into the second category of models that capture sentence semantics and perform robustly across tasks. While most of existing work focuses on English, this article concentrates on learning Chinese generic sentence representations.

Different from task-specific sentence representations, in which training and testing datasets are drawn from the same distributions, learning generic sentence representations requires raw or out-of-domain annotated text corpora. There have been some research efforts targeting this goal. One approach is to train recursive or recurrent sentence encoders with a reconstructive objective function to embed sentences into low-dimensional vectors [8, 27]. Another approach is the ParaphraseVec (PV) model [17], which represents sentences as fixed length vectors in a non-compositional way and trains them together with word vectors. Other methods utilize the encoder-decoder architecture to learn sentence representations by predicting the previous and next sentences, which are based on an extension of the distributional hypothesis (similar sentences occur in similar contexts) [13, 16]. However, the above methods either rely on complex model structures or huge training sets, leading to low training efficiency. To make the learning process of sentence representations as effective as word representations, Wieting et al. [32, 33, 34] proposed to learn generic sentence representations based on supervision from the Paraphrase Database [6]. Following their approach, we train our models based on supervision from Chinese paraphrases. The difference is that we extracted paraphrases from a machine translation evaluation corpus, which are high quality sentence pairs instead of noisy phrase pairs. As for textual representation learning in Chinese, Wang and Zong [31] conduct a comparison work on learning Chinese phrase representations. They find that the quality of word representations (enhance the word representations with semantic lexicon or not) plays a critical role in representing phrase meaning. Inspired by that, in this article, we explore the idea of using semantic enhanced word embeddings to learn sentence representations. Different from their method, instead of using semantic lexicon, we utilize inner-word characters to learn augmented word representations. Following Wang et al. [30], we utilize both of the characteristics of Chinese word-character relationships to learn generic Chinese sentence representations. Previously, Wang et al. [30] focused on designing model architectures to make full use of characters' semantic information. This article conducts a more comprehensive study of learning Chinese sentence representations. Specifically, besides character-word relationships, we also explore the effects of character disambiguation. Moreover, we also compare with more baseline models including both character and word level models, we test the learned sentence representations in downstream tasks of textual classification and entailment, and we perform a more detailed analysis of the learned sentence representations.

Another branch of related work is learning Chinese word representations with character level information. Chen et al. [3] propose a character-enhanced word embedding model, based on the framework of the Continuous Bag-of-Words (CBOW) model [21], by adding the averaged character embeddings to the word embeddings. Xu et al. [36] extend this work by using weighted character embeddings. The weights are cosine similarity between embeddings of a word's English translation and its constituent characters' English translations. However, their work calculates weights based on a bilingual dictionary, which brings lots of mistakes because words in two languages do not

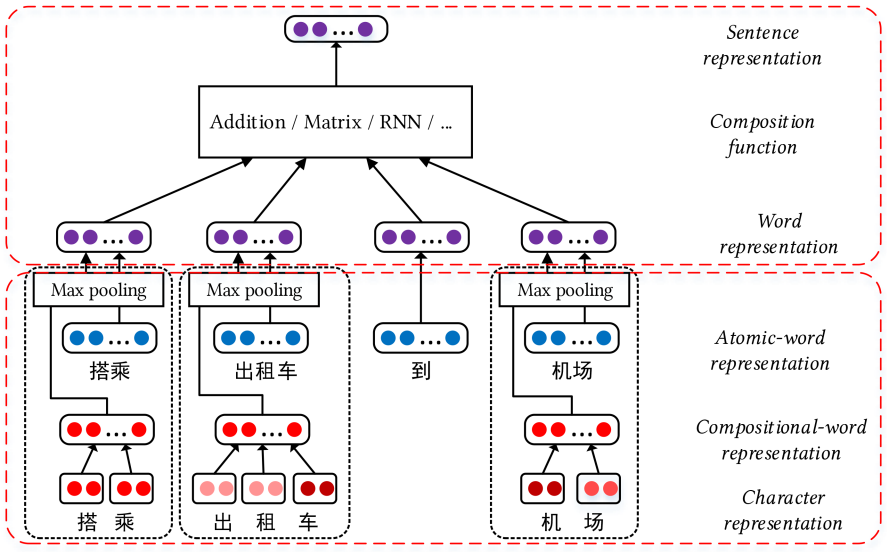

Fig. 2. The model architecture of our mixed character-word sentence representation model.[1] The top half denotes the architecture of the word-based sentence representation model. The bottom half illustrates our mixed character-word architecture, which can be incorporated into various sentence composition models to improve sentence representations.

have a one-to-one relationship. Furthermore, they only consider the first characteristic of inner-word characters, but ignore the contributions of the atomic and compositional word to the final word meaning. Similar ideas of adaptively utilizing character level information have also been investigated in English recently [7, 23, 25]. It should be noted that these studies do not focus on learning generic sentence embeddings.

3 MODEL DESCRIPTION

The problem of learning compositional sentence representations can be formulated as $R_{\text{sentence}} = f(\mathrm{x})$, where f is the **composition function** that combines the **word representations** $\mathrm{x} = \langle x_1, x_2, \ldots, x_n \rangle$ into the **sentence representation** R_{sentence}. To make full use of inner-word characters, we extend the word-based sentence representation models to include character level information. The model architecture is shown in Figure 2.

Next, we first describe how we utilize character level information to build mixed character-word representations, followed by the composition functions, which combine the generated word representations into sentence representations, and then introduce the training objective of our models.

3.1 Mixed Character-Word Representation

The final word representation is a fusion of the atomic and compositional word representations. The atomic word representation is calculated by regarding the words as inseparable units and projecting each word into a high-dimensional space by a look-up table, while the compositional

[1]In this article, the mixed character-word (sentence representation) model denotes the sentence representation model that utilizes the mixed character-word architecture.

word representation is computed as a gated composition of character representations:

$$x_i^{comp} = \sum_{j=1}^{m} v_{ij} \odot c_{ij}, \tag{1}$$

where $c_{ij} \in \mathrm{R}^{\mathrm{d}\times 1}$ is the j-th character representation in the i-th word and $\odot$ denotes inner product. The mask gate $v_{ij} \in \mathrm{R}^{\mathrm{d}\times 1}$ controls the contribution of the j-th character in the i-th word. The subscript d denotes dimension of the vector and m denotes the number of characters in a word. The mask gate is performed by using a feed-forward neural network operated on the concatenation of a character and a word, under the assumption that the contribution of a character is correlated with both character itself and its relation with the corresponding word:

$$v_{ij} = tanh(\mathrm{W}\,[c_{ij}; x_i]), \tag{2}$$

where $W \in \mathrm{R}^{\mathrm{d}\times 2\mathrm{d}}$ is a trainable parameter and $[c_{ij}; x]$ denotes the concatenation of vector c_{ij} and vector x. The proposed mask gate is a vector instead of a single value, which introduces more variations to character's meaning in the composition process.

Then, the atomic and compositional word representations are mixed with a max-pooling approach:

$$x_i^{final} = max_{k=1}^{d}\left(x_{ik}^{atomic}, x_{ik}^{comp}\right), \tag{3}$$

where the *max* is an element-wise function to capture the most important features (i.e., the highest value in each dimension) in the two generated word representations.

3.2 Sentence Composition Model

Given word embeddings, we make a systematic comparison of five different composition models for sentence representations as follows:

1. $R_{\text{sentence}} = \text{Average(x)} = \frac{1}{n}\sum_{i=1}^{n} x_i$
2. $R_{\text{sentence}} = \text{Matrix(x)} = \frac{1}{n}\sum_{i=1}^{n} f(W_m x_i)$
3. $R_{\text{sentence}} = \text{Dan(x)} = f(W_d(\frac{1}{\mathrm{n}}\sum_{i=1}^{n} x_i) + b)$
4. $R_{\text{sentence}} = \text{RNN(x)} = f(W_x x_i + W_h h_{i-1} + b)$
5. $R_{\text{sentence}} = \text{LSTM(x)} = o_t \odot f(c_i)$, where $c_i = f_i \cdot c_{i-1} + i_i \cdot \widetilde{c_i}$ and $\widetilde{c_i} = \sigma(W_{xc} x_i + W_{hc} h_{i-1})$

Average model, as the simplest composition model, represents sentences with averaged word vectors that are updated during training. The Matrix and Dan models are proposed in Zanzotto et al. [38] and Iyyer et al. [10], respectively. By using matrix transformations and nonlinear functions, the two models represent sentence meaning in a more flexible way. We also include RNN and LSTM models, which are widely used in recent years. The parameters $\{i_t, f_t, o_t\} \in \mathrm{R}^d$ denote the input gate, the forget gate, and the output gate, respectively. $\mathrm{c}_t \in \mathrm{R}^d$ is the short-term memory state to store the history information. $\{W_m,\ W_d,\ W_x, W_h,\ \ W_{xc}, W_{hc}\} \in \mathrm{R}^{d\times\mathrm{d}}$ are trainable parameters. h_{i-1} denotes representations in the hidden layer. Sentence representations in RNN and LSTM models are hidden vectors of the last token.[2]

3.3 Objective Function

Following Wieting et al. [32], we employ the max-margin objective function to train generic sentence representations by maximizing the distance between positive examples and negative examples. Specifically, our training data consists of a set X of sentence paraphrase pairs (x_1, x_2), while

[2]We have also tried to use the averaged hidden vectors or the max-pooling results of hidden vectors, and we find the hidden vectors of the last token perform the best in the experiments.

t_1 and t_2 are negative examples that are the most similar sentences to x_1 and x_2, respectively in a mini-batch during optimization. The objective function is given as follows:

$$\min_{W_w} \frac{1}{|X|}\left(\sum_{(x_1,x_2)\in X} \max\left(0, 1 - W_w^{x_1}\cdot W_w^{x_2} + W_w^{x_1}\cdot W_w^{t_1}\right) + \max\left(0, 1 - W_w^{x_1}\cdot W_w^{x_2} + W_w^{x_2}\cdot W_w^{t_2}\right)\right) + \lambda\left\|W_{w_{initial}} - W_w\right\|^2, \quad (4)$$

where λ is the regularization parameter, $|X|$ is the number of training paraphrase pairs, W_w is the target word vector matrix, $W_{w_{initial}}$ is the initial word vector matrix, and W_w denotes the representation of a specific sentence.

4 EXPERIMENTS

4.1 Datasets

To build the **training dataset**, we extract Chinese paraphrases in machine translation evaluation corpora by combining every two sentences in the four equivalent Chinese translations. Specifically, we extract Chinese paraphrases in NIST2003,[3] which contains 1100 English sentences with four Chinese translations, and CWMT2015,[4] which contains 1859 English sentences with four Chinese translations. Moreover, we select aligned sub-sentence pairs between paraphrases to enlarge the training corpus. Specifically, we first segment the sentences into sub-sentences according to punctuations of comma, semicolon, colon, question mark, ellipses, and periods (, ; : ? … 。). Then, we pair all sub-sentences between a paraphrase and select sub-sentence pairs (s_1, s_2), which satisfy the following two constraints: (1) the number of overlapping words of sub-sentence s_1 and s_2 should meet the condition: $0.9 > len(\text{overlap}(s_1, s_2))/\min(\text{len}(s_1), \text{len}(s_2)) >= 0.2$, where *len(s)* denotes the number of words in sentence *s*; (2) the relative length of a sub-sentence should meet the condition: $\max(\text{len}(s_1), \text{len}(s_2))/\min(\text{len}(s_1), \text{len}(s_2)) <= 2$. Finally, we get 30,846 paraphrases (18,187 paraphrases from NIST including 11,413 sub-sentence pairs, and 12,659 paraphrases from CWMT which include 7912 sub-sentence pairs). An example of the training dataset (one paraphrase and its sub-sentence pairs) is as follows:[5]

— 日股(day shares) 价格(price) 周二(Tuesday) 收市(close) 平平(mediocre), 因为(because) 缺乏(lack) 任何(any) 国内(domestic) 正面(positive) 消息(news), 政治(political) 担忧(worry) 使(make) 人(people) 心绪(feel) 沉重(heavy) 。 ||| 日本(Japan) 股价(stock price) 在(on) 周二(Tuesday) 收盘(close) 持平(flat), 是(-) 因为(because) 对(-) 政局(political situation) 的 (-) 忧虑(worry) 压抑(suppressed) 了(-) 情绪(mood), 又(and) 没有(no) 任何(any) 积极(positive) 的(-) 国内(domestic) 消息(news) 。

— 日股(day shares) 价格(price) 周二(Tuesday) 收市(close) 平平(mediocre) ||| 日本(Japan) 股价(stock price) 在(on) 周二(Tuesday) 收盘(close) 持平(flat)

— 因为(because) 缺乏(lack) 任何(any) 国内(domestic) 正面(positive) 消息(news) ||| 又(and) 没有(no) 任何(any) 积极(positive) 的(-) 国内(domestic) 消息(news)

The **testing and development datasets** are sentence pairs collocated with human similarity ratings. We choose candidate sentences from the People's Daily and Baidu encyclopedia corpora.

[3] https://catalog.ldc.upenn.edu/LDC2006T04.

[4] http://www.ai-ia.ac.cn/cwmt2015/index.html.

[5] In this article, we segment all Chinese sentences into words with Urheen: 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.

Table 1. Inter-Annotator Agreement on Different Datasets Using Spearman Correlations

	Mean	Max	Min	SD
Renmin	0.8844	0.9271	0.8044	0.0395
Baidu	0.8783	0.9184	0.8113	0.0300
Total	0.8786	0.9160	0.8088	0.0318

The mean, max, min and SD represents average value, maximum value, minimum value and standard deviation, respectively. The higher values mean more consistency among the participants.

To assure sentence pairs to accommodate full variations in semantic similarity, we pair sentences in one paragraph, select high similarity sentence pairs by calculating cosine similarity,[6] and delete unrelated sentences manually. Then we randomly pair the left sentences to construct sentence pairs with low similarity. Finally, we get 1360 sentence pairs (1025 sentence pairs from the Baidu encyclopedia corpora and 335 sentence pairs from the People's Daily). To collect human similarity ratings for sentence pairs, we use an online questionnaire[7] and follow the gold standard[8] to guide the rating process of participants. The subjects are paid 7 cents for rating each sentence pair within a range of 0~5 score. In total, we obtain 104 valid questionnaires and every sentence pair is evaluated by an average of eight persons. We use the average subjects' ratings for one paraphrase as its final similarity score, and the higher score means that the two sentences have more similar meaning. We then randomly partition the datasets into test and development splits in 9:1. An example of the testing and development datasets is as follows:

—外汇(foreign exchange) 管理局(administration) 批复(reply) 中信(CITIC) 银行(bank) 经营(operating) 外汇(foreign exchange) 业务(business) 范围(range) ||| 在(in) 同一(same) 顺序(order) 法定(legal) 继承人(heir) 中(-), 不(do not) 得(-) 歧视(discriminate) 妇女(women) ||| 0.125
—紫薯(purple potato) 洗净(wash) 去皮(peeled) 切成(cut into) 片(pieces) ||| 紫薯(purple potato) 摊凉(spread out), 切成(cut into) 正方形(square) 。 ||| 2.75
—供应商(vendors) 开始(begin) 对(-) 采购商(buyers) 提供(offer) 小(small) 批量(quantities) 的(of) 产品(products) ||| 供应商(vendors) 对(-) 采购商(buyers) 提供(offer) 小(small) 批量(quantities) 的(of) 产品(products) ||| 4.125

To check the quality of the data we have collected, we follow Mitchell and Lapata [22] and examine how well participants agreed in their similarity judgments, 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 compared with humans. To calculate intersubject agreement, we use the 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 (and average them) 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. This was repeated m times and we get the results in Table 1.

In Table 1, "Mean," "Max," "Min," and "SD" represent average value, maximum value, minimum value, and standard deviation, respectively. The higher value of "Mean" and lower value of "SD" denote more consistency among the participants. As shown by the "Mean" (column 2) and "SD"

[6]Here, we use averaged word embeddings as the sentence representation.
[7]https://wj.qq.com/.
[8]http://alt.qcri.org/semeval2015/task2/index.php?id=semantictextual-similarity-for-english.

(column 5) results, the consistency is high among the questionnaire participants' responses—even though the participants thought that the sentence similarity evaluation task was difficult.

4.2 Baselines

We construct four groups of models (G1~G4), which serve as baselines to test the proposed mixed character-word models (G5). Group G1 includes six baseline models, which have shown impressive performance in English. The first two are averaged word vectors (Average-word) and averaged character vectors (Average-character) without training. Followed by two character-based models (Char-CNN and Charagram), and two word-based models (PV-DM and FastSent). A short description of these models is as follows:

- —The Character convolutional neural network model (Char-CNN) [14] is one kind of CNN model with character n-gram filters, which has been widely used to embed sentences or documents for several NLP tasks.
- —The Charagram model [33] represents words or sentences using character n-gram count vectors, which are embeded into a low-dimensional space by nonlinear transformation.
- —The distributed memory model of paragraph vectors (PV-DM) [17] is an extension of the Skip-gram word representation model [21]. The sentences in the PV-DM model are mapped to a unique vector and averaged (or concatenated) to the word vectors in the sentence to predict the next word in a context.
- —As a simplification of the SkipThought model [16], the FastSent model [8] replaces the LSTM sentence representation module with the simple additive model and exploits the same objective function as SkipThought, which is to predict the previous and the next sentences.

In group G2, G3, and G4, each group of models applies a different word embedding method with five sentence composition models in Section 3.2. The character-based model (G2) and word-based model (G3) utilize basic units of characters and words, respectively, in which the word-based models are explored in Wieting et al. [32] for English. The averaged character-word models (G4) use the summation of a word vector and an averaged character vector as the final word vector, which is the method used in Chen et al. [3].

4.3 Experimental Settings

In all models, the word and character embeddings are initialized with 300-dimension vectors trained by the Skip-gram model [21]. Specifically, we use a corpus[9] which contains 3 billion Chinese words, and we use this corpus to construct another corpus by segmenting it at the character level. Finally, we use a concatenation of the two corpora to train word and character embeddings. This simple method enables us to easily train the distributed representations of words and characters simultaneously. All models are implemented with the work of Theano [1] and Lasagne [5], and optimized using the work of Adam [15]. In this article, we run all experiments five times and report the mean values. We use a mini-batch of 25 and tune the initial learning rate over {0.001, 0.005, 0.0001, 0.0005}. For the Dan and the Matrix models, we tune the over-activation function (tanh or linear or rectified linear unit) and the number of layers (1 or 2). The hyper-parameters are selected by testing different parameter values and evaluating their effects on the development set.

[9]The corpus is datasets crawled from Xinhua News: http://www.xinhuanet.com/ and Baidu encyclopedia: https://baike.baidu.com/.

For the other baseline models, we tuned the parameters as follows:

—Char-CNN: We use the set of filters from Kim et al. [14], which are filters of width [1; 2; 3; 4; 5; 6] of size [25; 50; 75; 100; 125; 150] for a total of 525 filters. We tune over the activation functions in the convolution (tanh or linear or rectified linear unit) and the output layer (tanh or linear or rectified linear unit).
—Charagram model: We tune over the number of n-grams (3,4,5) and the activation function in the output layer (tanh or linear or rectified linear unit).
—PV-DM: We use the implementation in gensim tool[10] with the same parameters used in the Skip-gram model. The training data is one-tenth of the corpora as in the Skip-gram model to prevent memory overflow.
—Fastsent: We use the training data from the People's Daily corpus[11] (22M words), which are contextually coherent paragraphs, and train the model with the default parameters.

4.4 Results on Sentence Similarity

We use the Pearson's correlation coefficient to examine relationships between the averaged human ratings and the predicted cosine similarity scores of all models. Moreover, the Wilcoxon's test shows that significant difference ($p < 0.01$) exists between our models (G5) with baseline models (G1, G2, G3, G4).

Comparing different sentence composition functions in Table 2, we find that the two simple composition functions, i.e., Matrix and Dan, achieve the best performance in all groups of models. The two recurrent neural network models (i.e., RNN and LSTM), which have more complex sentence structures, perform even worse than the simplest average model. This indicates that simpler composition functions are more suitable for learning generic sentence representations in the condition of the small amount of paraphrase training data used in this article. We leave this to the future work to explore other kinds of training data to effectively train complex composition functions like RNN and LSTM.

Comparing the best performing sentence composition function (which is Dan) in different groups in Table 2, we can see that the proposed mixed character-word models (G5), which utilize gate and pooling methods, have significantly improved the performance over baseline models in group G2, G3, and G4. Specifically, using mask gate alone and max pooling alone yield an improvement of 1.05 points and 0.83 points, respectively, and using both strategies improves the averaged character-word models by 1.52 points. This result indicates that it is important to find the appropriate way to fuse character and word-level information. Moreover, as shown in Table 2, models exploiting both word and character level information (G4, G5) perform better than the pure word-based models (G3), which proves the usefulness of characters in representing sentence meaning.

Another observation is that the majority of the character-based models (G2) perform better than the word-based models (G3). This is surprising because the word is the basic semantic unit of Chinese language. There are two possible explanations for this phenomenon. One is word segmentation error, which is still an unsolved problem. The other is insufficient training of rare words. To test this, we increase the training dataset with large short paraphrase pairs constructed in Wang and Zong [31] and retrain the word-based models. In the extended corpus, words that appear less than five times drop from 13.5% to 10%. However, we only find a 1% improvement in the average

[10] https://radimrehurek.com/gensim/models/doc2vec.html.
[11] http://www.lancaster.ac.uk/fass/projects/corpus/pdcorpus/pdcorpus.htm.

Table 2. Correlation Coefficients of Model Predictions with Subject Similarity Ratings on Chinese Sentence Similarity Task

Group	Model	Renmin	Baidu	Total
G1: Baselines	Average-character	0.6737	0.6957	0.6976
	Average-word	0.7745	0.7657	0.7518
	Char-CNN [14]	0.8086	0.8077	0.8095
	Charagram [33]	0.8359	0.8393	**0.8382**
	PV-DM [17]	0.7541	0.7552	0.7561
	FastSent [8]	0.7423	0.7359	0.7369
G2: Character-based	Average	0.8631	0.8451	0.8484
	Matrix	0.8612	0.8464	0.8496
	Dan	0.8638	0.8483	**0.8507**
	RNN	0.8167	0.8338	0.8286
	LSTM	0.7762	0.7714	0.7726
G3: Word-based	Average	0.8271	0.8192	0.8199
	Matrix	0.8419	0.8365	0.8382
	Dan	0.8419	0.8378	**0.8385**
	RNN	0.7995	0.8176	0.8121
	LSTM	0.7922	0.7801	0.7834
G4: Averaged Character-Word	Average	0.8320	0.8238	0.8245
	Matrix	0.8498	0.8410	**0.8427**
	Dan	0.8502	0.8379	0.8407
	RNN	0.8105	0.8233	0.8185
	LSTM	0.7949	0.7875	0.7895
G5: Mixed Character-Word	Average	0.8467	0.8477	0.8471
	Matrix	0.8462	0.8486	0.8517
	Dan	0.8639	0.8491	**0.8521**
	RNN	0.8416	0.8425	0.8408
	LSTM	0.7829	0.8062	0.8000

The bold data refers to the best result in each group of models.

model and a slight performance drop on the others. The above results indicate that Chinese characters have great potential in learning generic sentence representations.

4.5 Effects of Character Disambiguation

Compared with words, Chinese characters are much more ambiguous. Previous work has proven that multi-prototype character embeddings are useful for learning better word representations. However, it is still unknown whether multi-prototype character strategy is effective in learning Chinese sentence representations. In this article, we propose two character disambiguation methods to address this issue:

1. Cluster-based method
   Cluster-based word disambiguation methods cluster the textual context of a word to distinguish different word senses [9, 26]. Similarly, to distinguish different senses of a character, we cluster the embeddings of all words in which it appears. For instance, we have a character "音(sound, tone,…)" and it appears in words "音乐(music)," "知音(bosom friend)," "司音(siyin)," and so on. By utilizing the k-means method, we can summarize these

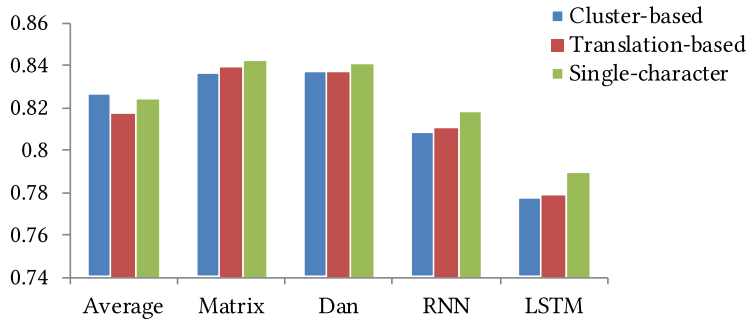

Fig. 3. Performance of the averaged character-word (G3) models on sentence similarity tasks (with total datasets) with single characters, cluster-based, and translation-based multi-prototype characters.

words into k clusters,[12] resulting in different character senses as {"音-1": "音乐(music)," "音调(tone)", ...}, {"音-2": "知音(bosom friend)", ...}, and {"音-3": "司音(siyin)", ...}.

2. Translation-based method
Following the character disambiguation method of Xu et al. [36], we first obtain translations of Chinese words and characters with a bilingual dictionary.[13] For example, this Chinese word "音乐" has two characters, which are "音" and "乐". Their English translations are "music", "sound; news; tidings tone; a surname; ...," and "music; a surname; pleasure; cheerful; laugh; ...," respectively. Then, we merge similar meanings of these English translations for Chinese characters to limit the number of word senses. As in the above character "乐," its translation words "pleasure" and "cheerful" are merged because their word embeddings have high cosine similarity. Finally, we disambiguate the sense of a character by computing cosine similarity between embeddings of their translation words. For instance, character "音" has multiple senses, and we give it the sense "sound" when used in the word "音乐(music)" because the embeddings of "sound" and "music" have high cosine similarity.

To inspect if multi-prototype character embeddings improve the quality of sentence representations, we employ the above cluster-based and translation-based methods to disambiguate characters in the training and testing datasets. Here, we take the simple averaged character-word models as an example for illustration. Specifically, we replace all characters in the training and testing datasets with the disambiguated characters and train the model with the same approach described in Section 4.4.

As shown in Figure 3, the two character disambiguation methods have mostly negative effects on sentence similarity task. The only exception is the cluster-based multi-prototype characters in the simple average composition model. In the experiment, we have tried various experimental settings, such as using various similarity calculation methods like edit distance and heuristic rules, restricting the number of words in which characters are disambiguated (specifically, we choose words which occur more than 10 times as candidates and only disambiguate characters in those words). However, no improvement over a single prototype character is found. Moreover, we have also tuned the number of character senses. Results show that a smaller number of senses for a character leads to better performance.

[12]Follow the suggestions in [3], we set k as 3 in the experiments.
[13]We use ICIBA as an English-Chinese translation tool for free. API: http://www.iciba.com/.

There are several reasons for this phenomenon. One is that the quality of disambiguated character representations is not good enough, which we test with character "道(road, Taoism, talk, etc.)" as an example. Specifically, for each sense of the character, we calculate the three most similar words with cosine similarity.[14] As for the cluster-based method, we get "树(tree), 花台(flower bed), 林荫(trees)" for "道-1;" "贫道(pindao), 仕(official), 奈(chennai)" for "道-2;" and "报道(reported), 邮报(post), 国家报(national newspaper)" for "道-3." As for the translation-based method, we get "网(network), 管(tube), 线(wire)" for "道-1;" "路(road), 村道(village road), 道路(road)" for "道-2;" and "说(say), 媒体(media), 报道(reported)}" for "道-3." Due to differences in methodology, the cluster-based method disambiguates characters according to word usage, while the translation-based method disambiguates characters according to word meaning. From the example, we can see that both methods can generate reasonable multi-prototype characters, but meanwhile, introduce some errors. The second reason is that although single character representations contain multiple character senses, they can be disambiguated when combined with specific word representations. This is reasonable because a character usually has one sense in a specific word. The last reason is that the training data of paraphrase pairs are not enough. We leave this to the future work to explore other kinds of training data, such as aligned articles for the same topic, which can be easily constructed in a large quantity.

4.6 Using Learned Representations as Features

To verify the generalization ability of the learned sentence representations, we perform sentence classification and question classification tasks, which classify sentences and questions, respectively, according to the predefined categories. In addition, we also test the learned sentence representations on the sentence entailment task that recognize the entailment relation (entailment, contradiction, and neutral) between two sentences.

For sentence classification, we build a dataset based on the Fudan document classification dataset.[15] The Fudan corpus contains 20 categories of documents, including art, literature, sports, and so on, in which we extract the title or the first sentence of the abstract.[16] Then, we delete the categories with less than 20 sentences and the sentences with a length of less than 5 words. Finally, we obtain 2142 sentences in nine categories. For question classification, we use the Fudan question classification dataset,[17] which contains 13 types of questions, including description, fact, evaluation, and so on. To avoid imbalance, we delete the categories with less than 30 questions, and finally, obtain a total of 17252 questions in 10 categories. For both datasets, we split training, development, and testing datasets as 7:1:2.

For both tasks, we use the multi-layer perceptron classifier with cross-entropy objective function. Let h_c be the learned sentence representations, y be the target distribution of sentence category, and $\hat{y}$ be the predicted distribution. The goal of training is to minimize the cross-entropy loss between y and $\hat{y}$ for all sentences:

$$h_s = \sigma(W_s h_c),$$

$$\hat{y} = \text{softmax}(W_p h_s),$$

$$\text{Loss} = -\sum_i \sum_j y_i^j log\hat{y}_i^j + \lambda \|\theta\|, \tag{5}$$

[14] The three senses of character "道" are denoted as "道-1", "道-2", and "道-3".
[15] http://www.datatang.com/data/44139.
[16] We segment sentences based on Chinese punctuations: full stop, question mark, exclamatory mark.
[17] http://www.nlpir.org/?action-viewnews-itemid-106.

Table 3. Evaluation Accuracies (%) of Models without or with Learned (G2, G3, G4, G5) Sentence Representations on Sentence Classification Dataset

	Average	Matrix	DAN	RNN	LSTM
Standard Settings	75.80	76.48	77.63	74.43	**77.40**
Character-based (G2)	77.17	75.57	75.80	76.71	74.55
Word-based (G3)	78.77	77.85	76.26	78.31	75.37
Averaged Character-Word (G4)	79.00	76.94	77.63	77.52	76.31
Mixed Character-Word (G5)	**79.22**	**78.77**	**78.31**	**79.00**	74.43

The bold data refers to best results within the same sentence composition model.

where i is the index of the sentence, j is the index of the sentence category, σ is the logistic sigmoid function, $W_s \in \mathrm{R}^{d\times \mathrm{d}}$ and $W_p \in \mathrm{R}^{d\times \mathrm{k}}$ are trainable parameters, d is dimension of the vector, k is the size of the sentence category, and λ is the L_2 regularization term operated on the parameter set θ.

As for the sentence entailment task, we build a Chinese sentence entailment dataset based on the English sentence entailment dataset (Sentence Involving Compositional Knowledge (SICK)[18] in SemEval 2014) since no public Chinese dataset is available. Specifically, we use a neural machine translation tool[19] to translate the English SICK dataset into Chinese and modify the obvious mistakes manually. In this way, we get a Chinese sentence entailment dataset that consists of 9,927 sentence pairs in a 4,500/500/4,927 train/dev/test split.

To classify the entailment relation between two sentences, we first produce sentence representations h_l and h_r for each sentence pair. Next, we use their element-wise multiplication and subtraction to capture the similarity and difference between their representations, which are then transformed by the multi-layer perceptron. Finally, we use a cross-entropy objective function to train the model:

$$
\begin{aligned}
W_\times &= h_l \odot h_r, \\
W_+ &= |h_l - h_r|, \\
h_s &= \sigma\left(W_\times h_\times + W_+ h_+\right), \\
\hat{y} &= \mathrm{softmax}(W_p h_s), \\
\mathrm{Loss} &= -\sum_i \sum_j y_i^j log\hat{y}_i^j + \lambda\|\theta\|,
\end{aligned}
$$

where $W_\times \in \mathrm{R}^{d\times \mathrm{d}}$, $W_+ \in \mathrm{R}^{d\times \mathrm{k}}$, and $W_p \in \mathrm{R}^{d\times \mathrm{k}}$, and are trainable parameters. The $\odot$ denotes element-wise multiplication.

For comparison, we also investigate how these models perform in the standard setting where sentence representations are the averaged word embeddings (without further training). In the experiments, we use a mini-batch of 25, tune the initial learning rate over {0.001, 0.005, 0.0001, 0.0005} with optimization method of Adam, and tune the L_2 regularization term over {1e-03, 1e-04, 1e-05, 1e-06}. The performance is evaluated by predicting accuracy on the testing set (shown in Table 3, Table 4, and Table 5).

The same as the results in sentence similarity experiments, Table 5 shows that in the sentence entailment task, the character-based models achieve better results than word-based models when incorporated with most sentence composition models. On the contrary, as shown in Table 3 and Table 4, character-based models achieve a much worse performance than word-based models in the sentence and question classification task. One possible reason is that the sentence similarity

[18] http://clic.cimec.unitn.it/composes/sick.html.
[19] https://www.microsoft.com/en-us/translator/translatorapi.aspx.

Table 4. Evaluation Accuracies (%) of Models without or with Learned (G2, G3, G4, G5) Sentence Representations on Question Classification Dataset

	Average	Matrix	DAN	RNN	LSTM
Standard Settings	0.6170	0.6246	0.6269	0.6196	**0.6741**
Character-based (G2)	0.6185	0.6153	0.6142	0.6261	0.5693
Word-based (G3)	0.6336	0.6231	0.6175	0.6397	0.6232
Averaged Character-Word (G4)	0.6449	0.6336	0.6332	**0.6408**	0.6443
Mixed Character-Word (G5)	**0.6453**	**0.6369**	**0.6409**	0.6281	0.6540

The bold data refers to best results within the same sentence composition model.

Table 5. Evaluation Accuracies (%) of Models without or with Learned (G2, G3, G4, G5) Sentence Representations on Sentence Entailment Dataset

	Average	Matrix	DAN	RNN	LSTM
Standard Settings	0.7061	0.7700	0.7651	0.7310	0.7635
Character-based (G2)	0.7454	0.7753	0.7810	0.7521	0.7691
Word-based (G3)	0.7266	0.7727	0.7735	0.7461	0.7767
Averaged Character-Word (G4)	0.7765	0.7757	0.7936	0.7603	**0.7899**
Mixed Character-Word (G5)	**0.7800**	**0.7814**	**0.8049**	**0.7733**	0.7875

The bold data refers to best results within the same sentence composition model.

Table 6. Correlation Coefficients of Model Predictions with Subject Similarity Ratings on Chinese Word Similarity Task, Where C, W, and G Represent Character, Word, and Mask Gate, Respectively

	Character	Word	C&W	C&W&G
Average	0.4903	0.4311	0.4584	**0.5245**
Dan	0.4672	0.4470	0.5410	**0.5716**
Matrix	0.4784	0.4496	0.5458	**0.5694**
RNN	0.4646	0.4562	0.5656	**0.5674**
LSTM	0.4669	0.4535	0.5674	**0.5734**

task depends more on overlap ratio of words or characters in two sentences, whereas the sentence classification task relies more on topical information where words take superiority. In general, the majority of models in G2, G3, G4, and G5 outperform models in standard settings, which indicate the effectiveness of the learned sentence representations used as features. Moreover, we can also see that models in G5 perform better than models in G4 and they both outperform other models, which indicate the importance of combining word level and character level information and the superiority of the proposed character-word mixing strategy.

5 ANALYSIS

In this section, we analyze the effects of the two key strategies (i.e., mask gate and max pooling) of the proposed mixed character-word model in learning sentence representations. The mask gate assigns different weights to characters in a word, hopefully leading to better word representations. To intuitively show effects of the mask gate, we check characters whose l2-norm increases after applying the mask gate approach. We find that characters like “罪(crime)” in “罪状(guilty),” “虎(tiger)” in “美洲虎(jaguar),” and “瓜(melon)” in “黄瓜(cucumber)” achieve more weights. The

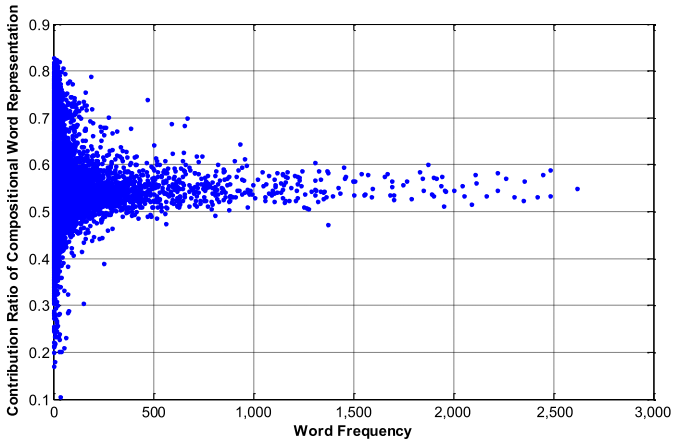

Fig. 4. Illustration of the relationship between word frequency and contribution ratio of the compositional word vectors to the final word vector.

above results illustrate that the mask gate approach successfully models the first characteristic of inner-word characters (i.e., assigning more weights to key characters). To quantitatively display the results, we extract the word representations calculated by the five sentence composition models in four different groups and evaluate their quality on WordSim-297 dataset[20] using the Pearson correlation method. As shown in Table 5, the mask gate approach significantly improves the quality of word representations.

The max-pooling approach is supposed to model contributions of the atomic and the compositional word vectors to the final word vector. To find out what the max-pooling method has learned, we use contribution weights by calculating cosine similarity between the final word representation with the atomic and compositional word representations. As shown in Figure 4, the results show interesting relationships with word frequency.[21] For high-frequency words, the contribution of compositional word representations is more dominant. While for low frequency words, both high and low contribution ratios of compositional word representations can be found. The high ratio of compositional word representations, which means that they are more important in representing meanings of the word, is more reasonable because generally poor atomic word representations are learned for low-frequency words. When looking into words with the lowest ratio, we find a large portion of English abbreviations like NBA, BBC, GDP, and so on, and a portion of metaphorical words, like "挂靴(retire, hanging boots)" and "扯皮(wrangle, pull skin)." Both kinds of these words are non-transparent, which illustrates that the max-pooling method can successfully model the second characteristic of inner-word characters and encode word transparency to some extent.

Another interesting observation is that name entities, which are non-transparent words, appear in both high- and low-contribution ratios of compositional word representations. This indicates that some characters in name entities are indicative, which is helpful to learn better word representations. For instance, the character "李(Li)" in the word "李建军(Li jianjun)" is a common Chinese family name which indicates that the word "李建军(Li jianjun)" is a name entity.

[20] https://github.com/Leonard-Xu/CWE/tree/master/data.

[21] We use total training and testing datasets to calculate word frequency.

6 CONCLUSIONS AND FUTURE WORK

In this article, we introduce a novel mixed character-word architecture to improve generic Chinese sentence representations by exploiting the internal character of words. Extensive experiments and analyses have indicated that our models can encode word transparency and learn different semantic contributions across characters. We have also created a dataset to evaluate composition models of Chinese sentences, which could advance the research for related fields. We summarize our main findings as follows:

—Modeling relations between characters and words with mask gate significantly improve the quality of word representations.
—Using max-pooling on compositional and atomic word representations can encode word transparency and help generate better sentence representations.
—In the condition of a relatively small amount of paraphrastic training data, using multi-prototype character embeddings has a negative effect on sentence similarity tasks.
—Using the generic sentence representations as features can improve performance of downstream tasks like sentence classification and question classification.

Building representations for sentences from their constituent words is complex, because different types of words have different effects in representing the meaning of a sentence and there exist multiple relations between words in a sentence. In the further work, we plan to conduct a finer-grained analysis on effects of different types of words (e.g., semantic words and syntactic words) and relations between words in a sentence. By exploiting methods like reinforcement learning, we hope to develop a sentence representation model that can assign different representations to different type of words and automatically learn the semantic relations between words in a sentence.

ACKNOWLEDGMENTS

The research work is supported by the Natural Science Foundation of China under Grant No. 61673380 and No. 61403379.

REFERENCES

[1] James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. 2010. Theano: A CPU and GPU math compiler in Python. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*.
[2] Samuel R. Bowman, Jon Gauthier, Abhinav Rastogi, Raghav Gupta, Christopher D. Manning, and Christopher Potts. A fast and unified model for parsing and sentence understanding. 2016. In *Proceedings of the 54th Annual Meetings of the Association for Computational Linguistics*. 1466–1477.
[3] Xinxiong Chen, Lei Xu, Zhiyuan Liu, Maosong Sun, and Huan-Bo Luan. 2015. Joint learning of character and word embeddings. In *Proceedings of the International Joint Conference on Artificial Intelligence*. 1236–1242.
[4] Jianpeng Cheng and Dimitri Kartsaklis. 2015. Syntax-aware multi-sense word embeddings for deep compositional models of meaning. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 1531–1542.
[5] Sander Dieleman, Jan Schlüter, Colin Raffel, Eben Olson, Søren Kaae Sønderby, Daniel Nouri, Daniel Maturana, Martin Thoma, Eric Battenberg, and Jack Kelly et al. 2015. *Lasagne: First release.* Zenodo: Geneva, Switzerland.
[6] Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The paraphrase database. In *Proceedings of the 12th Annual Conference of the North American Chapter of the Association for Computational Linguistics*. 758–764.
[7] Kazuma Hashimoto and Yoshimasa Tsuruoka. 2016. Adaptive joint learning of compositional and non-compositional phrase embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. 205–215.

[8] 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*. 1367–1377.

[9] 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.

[10] Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daum´e III. 2015. Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*. 1681–1691.

[11] Li. Jiwei, and Eduard H. Hovy. 2014. A model of coherence based on distributed sentence representation. In *Proceedings of the 2014 Conference On Empirical Methods in Natural Language Processing*. 2039–2048.

[12] Dimitri Kartsaklis. 2015. Compositional distributional semantics with compact closed categories and Frobenius algebras. *arXiv preprint arXiv:1505.00138*.

[13] Tom Kenter, Alexey Borisov, and Maarten de Rijke. 2016. Siamese CBOW: Optimizing word embeddings for sentence representations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. 941–951.

[14] Yoon Kim, Yacine Jernite, David Sontag, and Alexander M. Rush. 2016. Character-aware neural language models. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*. 2741–2749.

[15] Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

[16] Ryan Kiros, Yukun Zhu, Ruslan R. Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. *Advances in Neural Information Processing Systems*. 3294–3302.

[17] Quoc V. Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning*. 1188–1196.

[18] Jinxia Li. 2011. A quantitative analysis of the transparency of lexical meaning in modern Chinese dictionary. *Chinese Linguistics* 3, 54–62.

[19] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A self-attentive sentence embedding. arXiv preprint arXiv:1703.03130.

[20] Lucy J MacGregor and Yury Shtyrov. 2013. Multiple routes for compound word processing in the brain: Evidence from EEG. *Brain and Language* 126, 2, 217–229.

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

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

[23] Yasumasa Miyamoto and Kyunghyun Cho. 2016. Gated word-character recurrent language model. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. 1992–1997.

[24] Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song, and Rabab Ward. 2016. Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 24, 4, 694–707.

[25] Marek Rei, Gamal KO Crichton, and Sampo Pyysalo. 2016. Attending to characters in neural sequence labeling models. In *Proceedings of the 26th International Conference on Computational Linguistics*. 309–318.

[26] Joseph Reisinger and Raymond J. Mooney. 2010. Multi-prototype vector space models of word meaning. In *Proceedings of the 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. 109–117.

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

[28] Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*. 1556–1566.

[29] Shaonan Wang, Jiajun Zhang, and Chengqing Zong. 2017. Learning sentence representation with guidance of human attention. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. 4137–4143.

[30] Shaonan Wang, Jiajun Zhang, and Chengqing Zong. 2017. Exploiting word internal structures for generic Chinese sentence representation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 298–303.

[31] Shaonan Wang and Chengqing Zong. 2017. Comparison study on critical components in composition model for phrase representation. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)* 16, 3, 16.

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

[33] John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2016. Charagram: Embedding words and sentences via character n-grams. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. 1504–1515.

[34] John Wieting and Kevin Gimpel. 2017. Revisting recurrent networks for paraphrastic sentence embeddings. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*.

[35] Kam-Fai Wong, Wenjie Li, Ruifeng Xu, and Zhengsheng Zhang. 2009. Introduction to Chinese natural language processing. *Synthesis Lectures on Human Language Technologies* 2, 1, 1–148.

[36] Jian Xu, Jiawei Liu, Liangang Zhang, Zhengyu Li, and Huanhuan Chen. 2016. Improve Chinese word embeddings by exploiting internal structure. In *Proceedings of the 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics*. 1041–1050.

[37] Wenpeng Yin, Hinrich Schutze, Bing Xiang, and Bowen Zhou. 2016. ABCNN: Attention-based convolutional neural network for modeling sentence pairs. *Transactions of the Association for Computational Linguistics*. 4, 259–272.

[38] Fabio Massimo 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*. 1263–1271.18

Received May 2017; revised September 2017; accepted October 2017