Incorporating Prior Knowledge into Word Embedding for Chinese Word Similarity Measurement

DEGEN HUANG, JIAHUAN PEI, CONG ZHANG, KAIYU HUANG, and JIANJUN MA,
Dalian University of Technology, China

Word embedding-based methods have received increasing attention for their flexibility and effectiveness in many natural language-processing (NLP) tasks, including Word Similarity (WS). However, these approaches rely on high-quality corpus and neglect prior knowledge. Lexicon-based methods concentrate on human's intelligence contained in semantic resources, e.g., Tongyici Cilin, HowNet, and Chinese WordNet, but they have the drawback of being unable to deal with unknown words. This article proposes a three-stage framework for measuring the Chinese word similarity by incorporating prior knowledge obtained from lexicons and statistics into word embedding: in the first stage, we utilize retrieval techniques to crawl the contexts of word pairs from web resources to extend context corpus. In the next stage, we investigate three types of single similarity measurements, including lexicon similarities, statistical similarities, and embedding-based similarities. Finally, we exploit simple combination strategies with math operations and the counter-fitting combination strategy using optimization method. To demonstrate our system's efficiency, comparable experiments are conducted on the PKU-500 dataset. Our final results are 0.561/0.516 of Spearman/Pearson rank correlation coefficient, which outperform the state-of-the-art performance to the best of our knowledge. Experiment results on Chinese MC-30 and SemEval-2012 datasets show that our system also performs well on other Chinese datasets, which proves its transferability. Besides, our system is not language-specific and can be applied to other languages, e.g., English.

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

Additional Key Words and Phrases: Chinese word similarity, word embedding, prior knowledge

ACM Reference format:
Degen Huang, Jiahuan Pei, Cong Zhang, Kaiyu Huang, and Jianjun Ma. 2018. Incorporating Prior Knowledge into Word Embedding for Chinese Word Similarity Measurement. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.* 17, 3, Article 23 (April 2018), 21 pages.
https://doi.org/10.1145/3182622

1 INTRODUCTION

Word similarity is a task of measuring the lexical similarity degree between word pairs, which has attracted much attention as a fundamental research in many NLP tasks. To date, numerous approaches have been proposed for computing lexical similarity, which can be briefly categorized into lexicon-based methods, statistic-based methods, and embedding-based methods.

This work is supported by National Natural Science Foundation of China (Nos. 61672127, 61173100) and National Social Science Foundation of China (No.15BYY175). We also wish to thank NVIDIA Corporation for their donation of Tesla K40c GPU device.

Authors' addresses: D. Huang, J. Pei, C. Zhang, K. Huang, and J. Ma, Innovation Park Building A0933, School of Computer Science and Technology, Dalian University of Technology, Dalian, Liaoning, China; emails: huangdg@dlut.edu.cn, {p_sunrise, cccaaag, loverainbow}@mail.dlut.edu.cn, majian@dlut.edu.cn.

https://doi.org/10.1145/3182622

Lexical resources are structured, manually knowledge bases that describe semantic relations between words, including Tongyici Cilin (Mei et al. 1983; Zhao et al. 2009; Tian and Zhao 2010), Hownet (Liu and Li 2002; Dong and Dong 2006; Fan et al. 2015), Wordnet (Wu and Wu 2010; Shan and Francis 2013; Ahsaee et al. 2014), and so on. Lexicon-based methods mainly rely on these lexical resources and calculate the degree of similarities based on the distance between the two items in the structure of lexicons. Additionally, there are also researchers focusing on integrating different lexicons (Liang et al. 2013; Zhang et al. 2013; Zhu et al. 2016). These semantic measures, while interpretable and effective, have the disadvantage that the computation method only works when both members are presented in the lexicons, which limits the application scope of these methods, especially in IR field. Therefore, statistic-based methods and embedding-based methods tend to be more attractive to predict unknown words.

Statistic-based methods rely on large-scale corpus, including on-line and off-line resources. Off-line corpus can hardly be live-updated, which means the problem of unknown words remains unsolved. On-line web corpus provides a new method for statistic-based word similarity computation. Many researchers obtain documents containing specific words from web search engines and calculate similarity according to semantic context or the numbers of retrieval documents. Representative work includes the PMI-IR algorithm (Turney 2001), the LC-IR algorithm (Higgins 2005), the Web-PMI algorithm (Bollegala et al. 2007), and the NGD algorithm (Rudi and Vitanyi 2007). Statistic-based methods using on-line corpus solve the problem of corpus dependency, but they still have the disadvantages of data sparsity, noise data, and calculation failure.

Word embedding, also known as word vector, is a distributional representation of word by modeling the relation between the target word and its context. Word embedding methods usually focus on the embedding model, which maps words into vectors in semantic space, and then calculate similarity using space distance formulas. Well-known word embedding models include skip-gram model (Mikolov et al. 2013a, 2013b) and GloVe model (Pennington et al. 2014). However, limited to the *distributional hypothesis* assuming that similar words occur in similar contexts (Turney and Pantel 2010), basic embedding methods generally have three drawbacks in nature. First, they can hardly distinguish *semantic similarity* from *conceptual association* (Hill et al. 2016). For instance, the words in the pair (残疾/disablity, 死亡/death, score=2.8) may be regarded as a similar pair by mistake, because they can both occur in the X position of contexts like "An illness resulted in his X." Second, the differences between synonyms and antonyms are difficult to be captured using only embedding techniques (Ono et al. 2015). For example, the words in (积极/positive, 消极/negative, score=4.1) have a cosine similarity of approximately 0.76 in the *pre-trained word2vec model* released by Mikolov et al. (2013a). At last, context-dependent embedding methods are unable to differentiate distinct senses of a word (Iacobacci et al. 2015). One example of polysemy phenomenon is the pair (包袱/baggage; jokes in the crosstalk, 段子/joke, score=2.6). To be specific, if the word "包袱" is assigned as the most direct meaning of "baggage," the two words are most probably dissimilar, but the other meaning "jokes in the crosstalk" makes a closer distance. In addition, for Chinese language, the challenge is also due to the lack of contexts for the single-character words in such pair as (面/face; noodles, 首/head, score=4.7), because it is a rare utterance in general texts. Therefore, the drawbacks arouse people's recent interest in integrating lexicons into word embedding to capture multiple semantics (Chen et al. 2015; Rothe and Schütze 2015).

In the previous work, we present a framework that combines word embedding and semantic lexicon for Chinese word similarity computation by simple but meaningful combinations. These combination strategies are intuitive and effective, but they are only combinations of the results of single methods without improving word vector space. Therefore, we exploit a more reasonable way to incorporate prior knowledge into pre-trained word embedding. Although some related

work has also designed constraints based on antonym and/or synonym pairs provided by linguistic resources, their performance is still limited to the quality and quantity of the semantic lexicons. As a solution, we expand lexicon constraints to lexicon-similarity constraints to find more pairs under the common circumstances and inject statistical similarity to consider information from search engines rather than the manual resources. To demonstrate the performance, we do comparable experiments on PKU-500 released by Chinese Word Similarity Measurement (CWSM) shared task (Wu and Li 2016) of NLPCC-ICCPOL 2016 conference. The dataset is proposed as a benchmark to evaluate and compare different semantic similarity methods.

2 RELATED WORK

Representative semantic resemblance measurement strategies have been leveraged to estimate the degree of similarity between words, which are basically classified into lexicon-based methods, statistic-based methods, and embedding-based methods as mentioned in the last section. Whereas, the recent study highly focuses on the combination methods of lexicon and word embedding, most of which attempt to improve the word representation via semantic constraints including compatible pairs (e.g., synonymy), contrastive pairs (e.g., antonyms), and lexical relationships in concept hierarchy (e.g., hyponyms-hypernyms, meronyms-holonyms & entailments), and so on.

Some researchers basically exploit the combination strategies by merging the results of lexicon-based similarity and word embedding-based similarity via fundamental math operations (Pei et al. 2016) or simple rules (Guo et al. 2016).

Also, some of them treat the improvement process as a constrained optimization problem and retrain a new embedding model. Yu and Dredze (2014) incorporate knowledge about synonyms from Paraphrase Database and Wordnet, and Ono et al. (2015) also focus on capturing antonyms through thesauri and distributional information. Liu et al. (2015) propose a framework to tackle three types of semantic ordinal ranking inequalities as a set of constrains to optimize their improved model based on skip-gram model using stochastic gradient descent. Nguyen et al. (2016) integrate lexical contrast into embedding by weighting features with a function representing the overlap difference between synonyms and antonyms and then inject the contrast into skip-gram model.

Others focus on refitting the embedding model by the similarity constrains to crawl the similar words closer in their semantic vector space. Rothe and Schütze (2015) propose a system, *AutoExtend*, to learn embedding for synsets and lexemes with only word embedding and lexical resource. Faruqui et al. (2015) present a *retrofitting* model to refine any input vectors by post-processing learning based on relation knowledge and their work is also followed by lightweight *counter-fitting* model that reported by Mrkšić et al. (2016).

More and more research tends to highlight the combination of lexicons and word embedding for word similarity measurement task. Compared with simple combination strategies, our system improves word vector space instead of simply combining the results of single methods. Unlike retraining methods, our system takes much less time and resource to get a better embedding model with high performance. Refitting methods usually only consider existing lexicons, while our system investigates more general prior knowledge such as lexicon-similarity constraints and statistical similarity constraints.

3 APPROACH

3.1 Architecture

Figure 1 briefly illustrates the general architecture of our Chinese Word Similarity Measurement system that incorporates prior knowledge into word embedding using counter-fitting method.

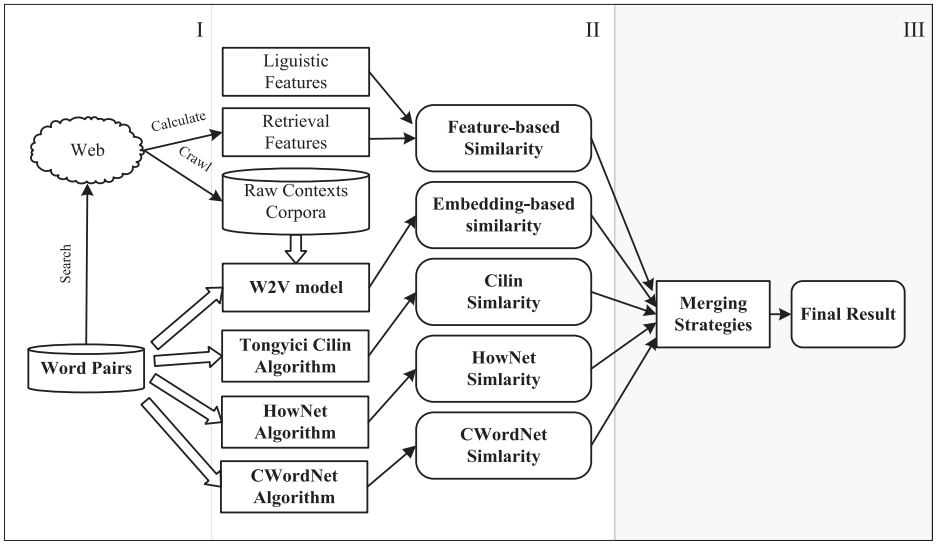

Fig. 1. The General Architecture of Chinese Word Similarity Computation.

It mainly consists of three components: component I is a preprocessing stage, which harvests the context corpus of the word pairs from the websites. Component II computes the word similarity based on single methods, including lexicon similarity based on three semantic resources (i.e., Cilin, HowNet, and WordNet), statistical similarity based on large quantities of corpus and embedding-based similarity. Component III adopts *counter-fitting* combination strategy to incorporate prior knowledge into an existing word embedding model. To prove the necessity of *counter-fitting* method, we also do experiments on fundamental *math operations* combination strategies.

3.2 Similarity Measurement based on Single Methods

In this section, we present different single methods for Chinese word similarity computation, which are ultilized in the component II.

3.2.1 Cilin Similarity. The *Cilin* model utilized in this article is developed from the algorithm proposed in reference (Zhu et al. 2016), which improves the basic algorithm from reference (Tian and Zhao 2010) to estimate both similarity and relevance between the words by fully exploiting the coding and structure information in the dictionary of *Tongyici Cilin (Extended)*.

The cilin dictionary is organized by a five-layer hierarchial structure, which is composed of large categories, medium categories, small categories, word group, and atomic word group layer from top to bottom. Correspondingly, it supplies five-layer patterns to generate coding for a group of words, which are joined by relationships like “synonym” or “relevance.” For instance, the words in the pair (紫禁城/the Forbidden City, 故宫/the Imperial Palace, score=10) are coded in the items “Bn23A03# 正殿...金銮殿 紫禁城” and “Bn23A02# 行宫 东宫...故宫” respectively. Figure 2 shows the two example words represented in cilin’s hierarchical structure.

Given two words w_a and w_b, let set $C_a = \{c_a^1, c_a^2, \ldots, c_a^A\}$ and $C_b = \{c_b^1, c_b^2, \ldots, c_b^B\}$ be concepts of w_a and w_b respectively. Set $W = \{W_1, W_2, W_3, W_4\}$ denotes the weights between every two adjacent layers, which subject to $0.5 \leqslant W_1 \leqslant W_2 \leqslant W_3 \leqslant W_4 \leqslant 5$ and $W_1 + W_2 + W_3 + W_4 \leqslant 10$. In our experiment, we set $W = \{0.5, 1.0, 2.5, 2.5\}$. The similarity between w_a and w_b can be computed as

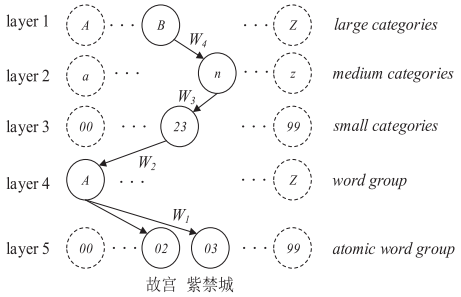

Fig. 2. The words "紫禁城" and "故宫" in the five-layer Hierarchical Structure.

maximum value of every "sense" in set C_a and C_b respectively, which is denoted as

$$SIM_{cilin}(w_a, w_b) = \max_{1 \leqslant i \leqslant A; 1 \leqslant j \leqslant B} \left\{ sim\left(c_a^i, c_b^j\right) \right\}, \tag{1}$$

where function $sim(c_a^i, c_b^j)$ defines the concept similarity between sense c_a^i and c_b^j as

$$sim\left(c_a^i, c_b^j\right) = \begin{cases} 1.0, \ if \ code(c_a^i) = code(c_b^j) \ \& \ end \ with \ " = " \\ 0.5, \ if \ code(c_a^i) = code(c_b^j) \ \& \ end \ with \ "\#" \\ (1.05 - 0.05 * dis(c_a^i, c_b^j))^{\sqrt{exp(-k/2n)}}, \ otherwise, \end{cases} \tag{2}$$

where n denotes the density of the nearest common parent node, k is the distance between two branches, $dis(c_a^i, c_b^j)$ can be defined as

$$dis\left(c_a^i, c_b^j\right) = \begin{cases} 18, \ if \ c_a^i, c_b^j \ not \ in \ the \ same \ tree \\ 2 * \sum_l W_{i-1}, \ otherwise, \end{cases} \tag{3}$$

where $l = 2, 3, 4, 5$ and it represents the number of layer in a hierarchial tree.

SIM_{cilin} is represented as a real number in domain **[0,1]**. Therefore, we transform the original domain **[0,1]** into **[1,10]** by a linear transformation.

3.2.2 HowNet Similarity. The HowNet model utilized in this article is developed from the algorithm proposed in reference (Liu and Li 2002), which computes word similarity depending on the elements, including *sememe*, *set* and *feature structure*, in the Chinese word dictionary proposed by Dong and Dong (2006).

Given two words w_a and w_b, let set $C_a = \{c_a^1, c_a^2, \ldots, c_a^A\}$ and $C_b = \{c_b^1, c_b^2, \ldots, c_b^B\}$ be sememes of w_a and w_b. Similar to Equation (1), $SIM_{hownet}(w_a, w_b)$ can be regraded as the similarity between w_a and w_b, which can be computed as the maximum similarity of every concepts in set C_a and C_b, respectively. And the concept similarity can be denoted as

$$sim\left(c_a^i, c_b^j\right) = \sum_{i=1}^{4} \beta_i \prod_{j=1}^{i} sim_j\left(c_a^i, c_b^j\right), \tag{4}$$

where $sim_1(c_a^i, c_b^j), \ldots, sim_4(c_a^i, c_b^j)$ refer to basic sememe similarity, set similarity, feature structures similarity, and relationship symbol similarity, correspondingly. Let parameter β_i represent the importance of the ith part, which should be subject to the constraints $\beta_1 + \beta_2 + \beta_3 + \beta_4 = 1$ and $\beta_1 \geqslant \beta_2 \geqslant \beta_3 \geqslant \beta_4$. In this experiment, we set $\beta_1 = 0.5, \beta_2 = 0.2, \beta_3 = 0.17, \beta_4 = 0.13$.

SIM_{hownet} is represented as a real number in domain [0,1]. Hence, we transform original the domain [0,1] into [1,10] by a linear transformation.

3.2.3 Chinese WordNet Similarity. The Chinese WordNet algorithm used in this article is developed based on the implementation of the reference (Wu and Hsieh 2010), which exploits the structural semantic resource, the Chinese Open Wordnet (COW), constructed by Shan and Francis (2013).

Given two words w_a and w_b, first, we can traverse the COW network to find out the synsets $synsets(w_a)$ and $synsets(w_b)$ that the two words belong to, so that semantically related words can be known and a search for a general term w_a or w_b will match documents containing specific terms. Then, we traverse each element pair in the two synsets and compute their similarity by the similarity measurement algorithms (e.g., *path_similarity*, *wup_similarity* and *lch_similarity*) from NLTK module. In our experiment, we choose the *path_similarity* because of its best performance of the three. It calculates the similarity relying on the shortest path between the two concepts in the hypernym hierarchy and ranges the score from 0 to 1. At last, we assign the maximum value of candidate similarities as the final result of similarity measurement.

As $SIM_{cwordnet}$ is represented as a real number in domain [0,1], we transform the original domain [0,1] into [1,10] by a linear transformation.

3.2.4 Statistical Similarities. Not only the distance in the lexicons, but also similarity-related retrieval statistics can provide clues on the degree of the word similarity. We use 4 types of web features (Bollegala et al. 2007), including web-jaccard, web-overlap, web-dice, and web-pmi to measure the similarity of word pairs.

We use the notation P and Q to denote the two words in a word pair, $H(P)$ to denote the result counts for the query P in a search engine, $P \cap Q$ to denote the conjunction query P and Q. The retrieval statistics web-jaccard, web-overlap, web-dice, and web-pmi can be defined as Equations (5)–(8):

$$web - jaccard(P \cap Q) = \begin{cases} 0, & H(P \cap Q) < c \\ \frac{H(P \cap Q)}{H(P)+H(Q)-H(P \cap Q)}, & H(P \cap Q) \geq c \end{cases}, \tag{5}$$

$$web - overlap(P \cap Q) = \begin{cases} 0, & H(P \cap Q) < c \\ \frac{H(P \cap Q)}{min\{H(P),H(Q)\}}, & H(P \cap Q) \geq c \end{cases}, \tag{6}$$

$$web - dice(P \cap Q) = \begin{cases} 0, & H(P \cap Q) < c \\ \frac{H(P \cap Q)}{H(P)+H(Q)}, & H(P \cap Q) \geq c \end{cases}, \tag{7}$$

$$web - pmi(P \cap Q) = \begin{cases} 0, & H(P \cap Q) < c \\ \log_2 \frac{H(P \cap Q)/N}{H(P)/N * H(Q)/N}, & H(P \cap Q) \geq c \end{cases}, \tag{8}$$

where N stands for the estimated number of documents indexed by the search engine. In the present work, we set $N = 10^{16}$, c = 10.

As all the four features range in domain **[0,1]**, we transform each of them into **[1,10]** by a linear transformation.

3.2.5 Embedding-based Similarity. We generate a pseudo-similarity score, namely Weak Similarity Score (WSS), using a linear average weighted combination of the results of single methods (mentioned in Section 3.2.1 to Section 3.2.4). We introduce a general approach for improving word embedding by weakly supervising the learning process with WSS. We begin with reviewing the basic skip-gram model and then present our improved method.

The skip-gram model is a learning framework to learn continuous word vectors from text corpus (Mikolov et al. 2013a, 2013b). It maps each word in the vocabulary into a continuous vector space by the method of looking up the embedding matrix $W^{(1)}$. $W^{(1)}$ is learned through maximizing the prediction probability of its neighbouring words within a context window, and the prediction

probability is calculated using another embedding matrix $W^{(2)}$. For a sequence of training data: $w_1, w_2, \ldots, w_N$, this model aims at maximizing the following objective function:

$$Q = \frac{1}{N} \sum_{n=1}^{N} \sum_{-c \leq j \leq c, j \neq 0} \log p(w_{n+j}|w_n), \tag{9}$$

where N represents the number of words, c is the size of context windows, w_n denotes the input central word, and w_{n+j} stands for its neighbouring word and the conditional probability $p(w_{n+j}|w_n)$ is defined as

$$p(w_{n+j}|w_n) = \frac{\exp(\mathbf{w}_{n+j}^{(2)} \cdot \mathbf{w}_t^{(1)})}{\sum_{k=1}^{V} \exp(\mathbf{w}_k^{(2)} \cdot \mathbf{w}_t^{(1)})}, \tag{10}$$

where $\mathbf{w}_n^{(1)}$ and $\mathbf{w}_k^{(2)}$ denote row vectors in matrices $W^{(1)}$ and $W^{(2)}$, corresponding to word w_n and w_k, respectively.

The basic unsupervised skip-gram model can produce impressive results depending on large context corpus. However, unsupervised learning may not be suitable for the task of interest as Yu and Dredze (2014) stated. Therefore, they incorporate semantic knowledge by supervised learning. But supervised learning highly relies on tagged resources. To avoid this limitation, we use WSS to select a better skip-gram model. Thus, we can use a weakly supervised method in this article, by which the automatically computed WSS can reflect the similarity between the word pair to some degree with the hypothesis—not only the contexts of the words, but also other types of similarity measures can reflect the degree of the word similarity. So far, our objective function can be denoted as

$$J = \max_{1 \leqslant iter \leqslant I} \{\Phi_{iter}(\overrightarrow{S_{pred}}, \overrightarrow{S_{wss}})\}, \tag{11}$$

where $\overrightarrow{S_{pred}} = \{s_1, s_2, \ldots, s_n\}$ is the sequence of prediction similarity scores, s_i denotes the similarity of the ith word pair, $\overrightarrow{S_{wss}} = \{s_1^*, s_2^*, \ldots, s_n^*\}$ is the sequence of WSS scores, s_i^* denotes the WSS score of the ith word pair, Φ denotes the spearman equation that detailed in Section 4.1, since it is the primary index for our evaluation, and I is the maximum number of iterations. In each iteration, a new W2V model is trained and evaluated by the function J. And the model with highest performance is selected to be used in the merging stage.

As cosine similarity between 2 vectors ranges from −1 to 1, we transform the original domain [−1,1] into [1,10] by a linear transformation.

3.3 Combination Strategies

3.3.1 Math Operations. We utilize five strategies based on fundamental math operations to merge the results of single methods stated in Section 3.2. For each word pair, we calculate a set of similarity scores through single methods and get a merged similarity score according to the merging strategies. Let $S_1, S_2 \ldots, S_k$ denote the scores in the k results respectively, and S_m to denote the merged score. The five merging strategies are defined as

—Max:

$$S_m = max\{S_1, S_2 \ldots, S_k\} \tag{12}$$

—Min:

$$S_m = min\{S_1, S_2 \ldots, S_k\} \tag{13}$$

—Arithmetic Mean:

$$S_m = \frac{S_1 + S_2 + \cdots + S_k}{k} \tag{14}$$

—Geometric Mean:

$$S_m = \sqrt[k]{S_1 * S_2 * \cdots * S_k} \tag{15}$$

—Linear Interpolation:

$$S_m = S^- + (k_m - k_m^-)\frac{S^+ - S^-}{k_m^+ - k_m} \tag{16}$$

where S_m is regarded as a missing value between S^- and S^+, and k_m, k_m^-, k_m^+ are the indexes of S_m, S^-, S^+ in the sorted list SORTED($\{S_1, S_2 \ldots, S_k\}$). When the number of results k is an odd number, S^+ and S^- are both the median number of the sorted list, and when k is an even number, S^+ and S^- are the two most median numbers of the sorted list, and $S^+ > S^-$.

3.3.2 Counter-Fitting. The math operation combination is intuitive and effective but does not improve the word embedding model. Therefore, we exploit a more reasonable way to incorporate prior knowledge into pre-trained word embedding. Our idea of counter-fitting is a new approach to incorporate semantic resources into word vectors in the spirit of the basic one (Mrkšić et al. 2016), which takes advantage of synonymy and antonyms relationships to fine-tune a pre-trained word embedding model.

Let A and S denote a set of linguistic constrains of antonyms and synonymy respectively, which consist of word pairs, e.g., pair (i, j), extracted from lexicons. Given a pre-trained word vector space $V = \{v_1, v_2, \ldots, v_N\}$ and its new word vector space after counter-fitting $V' = \{v_1', v_2', \ldots, v_N'\}$, this model aims at minimising the following objective function:

$$C(V, V') = k_1 VSP(V, V') + k_2 AR(V') + k_3 SA(V') + k_4 RSI(V'), \tag{17}$$

where hyper-parameters $k_1, k_2, k_3, k_4 \geqslant 0$ are the importance of four items. Let $d(v_i, v_j) = 1 - cos(v_i, v_j)$ be a distance between vector v_i and v_j, the margin of cost is computed as function $\tau(x) \triangleq max(0, x)$, then, the four item can be defined as Equation (18) to Equation (21), respectively:

—**Vector Space Preservation (VSP):** The original pre-trained word vectors capture the distributional information from large-scale corpus, therefore it is significant to design an item for preserving as much textual information in the original vectors as possible. Moreover, to reduce computational cost, radius ρ is introduced to only compute the nearest neighbourhoods $N(i)$ of the current word vector. And the VSP item can be defined as

$$VSP(V, V') = \sum_{i=1}^{N} \sum_{j \in N(i)} \tau(d(v_u', v_w') - d(v_u, v_w)). \tag{18}$$

—**Antonym Repel (AR):** Basically, the original embedding that solely based on context can hardly learn the differences between antonyms and synonyms. Therefore, the AR item is introduced to make sure that the distance between the words, which can be found in the word pairs A from antonym dictionaries should be no less than a certain threshold δ, and it can be denoted as

$$AR(V') = \sum_{(u,w) \in A} \tau(\delta - d(v_u', v_w')). \tag{19}$$

—**Synonym Attract (SA):** Likewise, the distance between the words that can be found in the word pairs S should be assigned more closer and no larger than the ideal maximum distance threshold γ. The SA item can be denoted as

$$SA(V') = \sum_{(u,w) \in S} \tau(d(v_u', v_w') - \gamma). \tag{20}$$

— **Residual Similarity Index (RSI):** The basic counter-fitting method successfully injects the antonym and synonym pairs from dictionaries to refit the original vectors. However, the performance can be still limited to the linguistic resources. Hence, we add the RSI item to capture the residual index related to different similarity measurements, to be more specific, we use lexicon-similarity constraints to find more pairs under the common circumstances and inject statistical similarity to consider information from search engines in addition to the manual resources. Similar to SA item, there is a threshold θ representing the ideal maximum distance between words to measure the cost of injecting other automatic similarity measurements into the original word vectors,

$$RSI(V') = \sum_{(u,w)\in S'} \tau(d(v'_u, v'_w) - \theta), \tag{21}$$

where S' stands for the constrain pairs that are extracted by the function $max\{sim_1, sim_2, \ldots, sim_N\} \geqslant \sigma$. Within, sim_i denotes the ith similarity measurement and σ is the lower bound in the selection procedure.

In our experiments, we tune parameters by 5 cross validation and use spearman's ρ as the evaluation index. Each development set consists of 300 word pairs that are randomly selected from PKU-500 and 40 word pairs of sample data PKU-40. Correspondingly, the rest 200 words pairs of PKU-500 is used as test set for each experimental group. Thus, five groups of experimental corpus are extracted for parameter tuning and performance evaluation. Finally, we get the importance weights as $k_1, k_2, k_3 = 0.02, 0.08, 0.14$, $k_4 = 1 - k_1 - k_2 - k_3$, the "ideal distance" parameters δ, γ, θ as $\delta = 0.5, \gamma = 0, \theta = 0.25$, the radius $\rho = 0.3$. We select word pairs for RSI with the minimum score $\sigma = (1 - \theta) * 10 = 7.5$ (the similarity score ranges from 1 to 10). We minimise the object function by Stochastic Gradient Descent (SGD) in the refitting process and iterate for 20 epochs.

To sum up, our model aims at minimising the objective function, which consists of four items, i.e., VSP, AR, SA, and RSI, corresponding to four sets of knowledge constrains items. VSP is designed for preserving as much textual information in the original vectors as possible. AR item serves to push antonymous words' vectors away from each other in the new transformed vector space V'. SA item brings the word vectors of known synonymous word pairs closer together in the new transformed vector space V'. RSI item is devised to capture the residual index related to different similarity measurements, in which lexicon-similarity constraints can be used to find more pairs under the common circumstances and statistical similarity can be injected to consider information from search engines in addition to the manual resources. Different from SA item, for each word pair, there is a variable threshold from a set of thresholds rather than a constant one representing the ideal maximum distance between words. So, it can alleviate overfitting and increase the robustness of our system.

4 EXPERIMENT SETTINGS

4.1 Data Set

The proposed approach is evaluated on the PKU-500[1] released by NLPCC-ICCPOL 2016 CWSM shared task (Wu and Li 2016). The dataset contains 40 sample data and 500 test word pairs with their similarity scores, which are properly balanced in terms of the different factors including *Domain*, *Frequency*, *POS Tags*, *Word length* and *Senses*. The similarity score between the two words ranges from 1 to 10, and the higher the score is, the more similar the two words are.

[1] http://tcci.ccf.org.cn/conference/2016/.

Table 1. Comparison between Different Lexicon-based Methods

No.	Methods	Overall ρ/r	Found ρ/r	Coverage %	Size
1	Cilin	**0.421/0.472**	0.520/0.550	**90.8**	**77,343**
2	HowNet	0.258/0.336	0.483/0.490	76.4	11,000
3	CWordNet	−0.053/0.204	**0.520/0.606**	31.2	61,533

4.2 Evaluation

The performance in our experiments is evaluated by the Spearman (ρ) and Pearson (r) rank correlation coefficient, which are widely used to test the consistency between automatic predicting results and the gold-standard human labelled data. The Spearman correlation coefficient (ρ) is defined as

$$\rho = 1 - \frac{6\sum_{i=1}^{n}(R_{Xi} - R_{Yi})^2}{n(n^2-1)}, \tag{22}$$

where R_{X_i} and R_{Y_i} are the standard deviations of the rank variables, which are converted from the raw scores X_i and Y_i, and n is the size of observations.

The Pearson correlation coefficient (r) is shown as

$$r = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \bar{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \bar{Y})^2}}, \tag{23}$$

where X_i and Y_i are the raw score, $\bar{X}$ and $\bar{Y}$ are the mean value, respectively, and n is the number of sample data.

For example, given $Data = \{(w_1^1, w_1^2, s_1), (w_2^1, w_2^2, s_2), \ldots, (w_n^1, w_n^2, s_n)\}$, where w_i^j denotes the ith word in the ith pair, and s_i is the ith gold-standard score, we can get $s_i^{'} = sim(w_i^1, w_i^2), i \in [1, n]$. Then we can get two sequences $X = \{s_1, s_2, \ldots, s_n\}$ and $Y = \{s_1^{'}, s_2^{'}, \ldots, s_n^{'}\}$, and use Equations (22) and (23) to calculate Spearman and Pearson index.

In our experiment, we regard the Spearman ρ as the main index and the Pearson r as the second important index for evaluation. Limited of space, all the resource utilized in this article are listed at https://github.com/JiahuanPei/NLPCC-2016-CLSC.

5 RESULTS AND ANALYSIS

To show the effectiveness of the proposed methods, in this section, we present the results of our similarity measurement experiments and analyze the key observations in terms of lexicon-based methods, statistical methods, embedding-based methods, and their combination methods.

5.1 Results of Lexicon Similarity Measurement

Table 1 shows the similarity measurement performance of three lexicon-based methods using Tongyici Cilin, Hownet, and Chinese Wordnet, respectively, where *Found* represents the local performance regardless of those *out of vocabulary* (OOV) words and *Overall* denotes the overall results that assigning similarity of the pairs containing OOV words with the minimum score of 1. *Coverage* denotes the proportion of the word pairs that can be looked up and *Size* lists the number of the words that contained in the dictionaries.

It can be clearly seen that the Cilin method reaches the ρ value of 0.421 and r value of 0.472 in the *Overall* case, which obviously outperforms the HowNet (0.258 ρ / 0.336 r) and Chinese WordNet (−0.053 ρ / 0.204 r) method in terms of both Spearman ρ and Pearson r. However, when it comes to the *Found* case, neglecting the effect of OOV, Chinese WordNet achieves a ρ value of

Table 2. Results of W2V Models Based on Different Corpus

No.	Corpus	ρ	r	ρ'	r'
1	Xieso (62M)	0.205	0.203	0.249	0.230
2	Wiki (1.1G)	0.211	0.213	0.324	0.343
3	Weibo (261M)	0.227	0.077	0.278	0.240
4	Datatang (199M)	0.267	0.272	0.337	0.343
5	News (381M)	0.311	0.305	0.317	0.277
6	News+Xieso	**0.311**	**0.311**	**0.359**	**0.310**
7	News+Xieso+Weibo	0.180	0.128	0.236	0.185
8	News+Xieso+Wiki	0.178	0.197	0.221	0.220
9	News+Xieso+Datatang	0.174	0.190	0.211	0.207
10	News+Xieso+DataTang+Wiki	0.214	0.239	0.314	0.308
11	News+Xieso+DataTang+Wiki+Weibo	0.214	0.134	0.247	0.203

Table 3. Comparison between the Original and Weakly Supervised W2V Models

No.	Strategy	ρ	r	ρ'	r'
1	Original W2V	0.296	0.241	0.330	0.262
2	Weakly Supervised W2V	**0.311**	**0.311**	**0.359**	**0.310**

0.520 and r value of 0.606, which performs better than those of Cilin (+10.18 % of r) and HowNet (+7.66% of ρ). The reason is that the low coverage of non-OOV, as well as the size of dictionaries, can be the main cause to the decrease of performance, meanwhile, more sematic information like hyponyms-hypernyms or meronyms-holonyms benefits to capture the similarity.

5.2 Results of Embedding Similarity Measurement

Table 2 shows the results of W2V models trained with 11 groups of different corpus, where ρ represents the Spearmans Rank Correlation between the result and the gold-standard score, and ρ' denotes that between the result and WSS, which is the main index used for weak supervision, r is the Pearsons Rank Correlation between the result and the gold-standard score, and r' stands for that between the result and WSS. For each W2V result, we train the W2V model under the weak supervision of WSS and choose the best one within 50 iterations.

As is shown in Table 2, both quantity and quality of corpus have a significant effect on the performance of W2V model. Comparing the results Nos. 1, 5, 6 of corpus with different scale, we can see that larger quantity of corpus may enrich more contexts to improve the performance. To be specific, No. 6 achieves a ρ value of 0.311 and r value of 0.311, which performs 0.106 and 0.108 higher than No. 1. However, the larger scale does not absolutely mean the higher performance (see Nos. 7–11), we infer that the quality of the corpus leads to these phenomena. Specifically, there are some paragraphs offending against the rules of grammar in the Datatang and Wiki corpus. Finally, the model of No. 6 is selected as the best W2V model in this step.

Table 3 shows the comparison between the original and weakly supervised W2V models. The original model could be any W2V model that is generated randomly at one iteration. The weakly supervised model is the best one in all 50 iterations, which is also illustrated in Table 2. This also indicates that the weak supervision method can distinguish the good models from the bad ones.

Table 4. Result of Statistical Similarities

No.	Strategy	ρ	r
1	web-jaccard	0.128	0.158
2	web-overlap	0.153	0.130
3	web-dice	0.128	0.156
4	web-pmi	0.262	0.286

Table 5. Merging Results Based on Five Math Operation Strategies

No.	Strategy	ρ	r
1	Max	**0.557**	**0.481**
2	Min	−0.078	0.040
3	Arithmetic Mean	0.486	0.467
4	Geometric Mean	0.360	0.346
5	Linear Interpolation	0.281	0.346

5.3 Results of Statistical Similarities

Table 4 is the results of statistical similarities based on statistics extracted from web corpus using information retrieval techniques. From the table, we can see that web-pmi's performance (0.262 ρ / 0.286 r) is much better than the other methods in this table, followed by web-overlap (0.153 ρ / 0.130 r), web-jaccard (0.128 ρ / 0.158 r) and web-dice (0.128 ρ / 0.156 r). In contrast, the web-pmi method based on the web search results is not as good as the Cilin method in Table 1, but it is better than HowNet and Chinese WordNet, which shows that the lexicon methods have more advantages when the coverage rate of known words are high, otherwise the methods based on web search results are more advantageous. Therefore, when there are a large number of unknown words, web-pmi is one of the important factors that can be considered.

5.4 Result of Combination Strategies

5.4.1 Result of Math Operations Combination. Table 5 shows the results of five merging strategies based on math operations, where the symbols ρ, r share the same meanings with those of Table 2. According to Table 5, different merging strategies can greatly affect the final result. The possible reason why *Max* performs best is that the single methods (e.g., Cilin, HowNet, WordNet) have a preference for Max strategy. In addition, it's much easier to understand two word objects that are similar than dissimilar for human. However, combination results based on math operations are very limited, therefore, we introduce counter-fitting to combine the single methods.

5.4.2 Result of Counterfitting Combination.

(1) Parameter Tuning.
In this article, the parameters tuning experiments are carried out on the basis of the experience value proposed by Mrkšić et al. (2016). For antonym pairs, their semantic meanings are opposed but they are highly related, thus we set the ideal distance between two antonym pairs as an intermediate value $\delta = 0.5$. For synonym pairs, their semantic similarity is very high and the ideal distance is set as $\gamma = 0$. Then, we tune parameter θ, k_1, k_2, k_3 and radius ρ in turn with initial values of $\theta = 0$, $k_1 = k_2 = k_3 = 0.1$, $\rho = 0.2$.

Figure 3 shows the Spearman's rank correlation coefficient with different θ values, assuming that other parameters are kept at their initial values. Each of these data points is

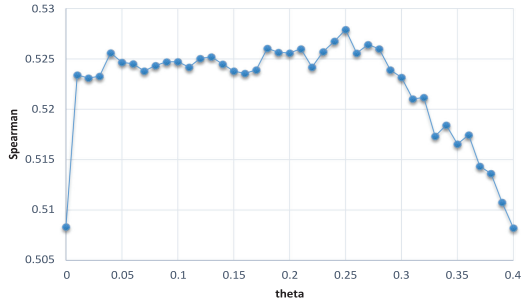

Fig. 3. Spearman's Rank Correlation Coefficient with Different θ Values.

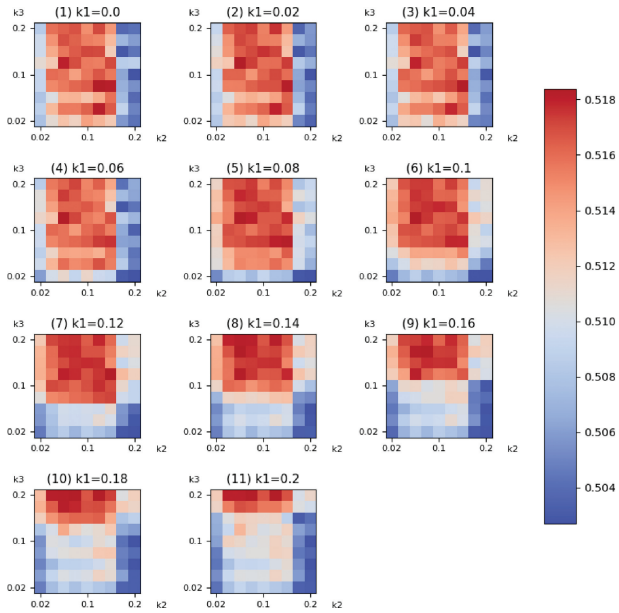

Fig. 4. Spearman's Rank Correlation Coefficient with Different k_1, k_2, k_3.

calculated from the average of the five groups of experiments. As can be seen from Figure 3, the Spearman index achieves the highest value when $\theta = 0.25$, so later θ will be set to 0.25.

Figure 4 shows the spearman's rank correlation coefficient with different k_1, k_2, k_3, where each data point is calculated from the average result of five groups of experiments. The warmer the color is, the better the performance is. From the figure, we can see that the Spearman index achieves the highest value when $k_1 = 0.02, k_2 = 0.08, k_3 = 0.14$.

Figure 5 shows the Spearman's rank correlation coefficient with different ρ values. It can be seen that $\rho = 0.3$ achieves the best performance and then the results remain almost unchanged. Since a larger ρ value can lead to a slower computation process, we set $\rho = 0.3$ in the latter experiments.

(2) Stability Analysis.

To prove the stability of the improved model proposed in this article, we set the parameters as the former part stated and evaluate the spearman index for each test set in five

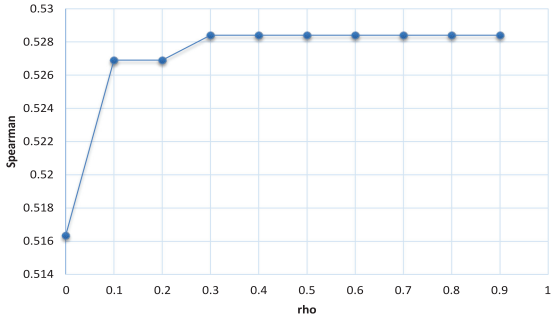

Fig. 5. Spearman's Rank Correlation Coefficient with Different ρ.

Table 6. Result of Five Groups of Five Random Sampling Experiments

Group	ρ	r
1	0.551	0.525
2	0.527	0.511
3	0.528	0.491
4	0.526	0.467
5	0.506	0.453
Average	0.528	0.489

Table 7. Result of Improvement by Basic Counter-Fitting and Improved Counter-Fitting

No.	Strategy	ρ	r
1	Baseline: W2V	0.311	0.311
2	Baseline+Basic Counter-fitting	0.508	0.491
3	Baseline+Improved Counter-fitting	**0.561**	**0.516**

experimental groups. The results are shown in Table 6. It can be seen that the results of the five random sampling experiments are better than those of the single models (see Tables 1, 2, 4). This shows that the method of integrating semantic constraints into word vector can effectively improve the performance of semantic similarity measurement, and its promotion effect is stable.

(3) Efficiency Analysis.
To demonstrate the performance of our improved counter-fitting method, we conduct three groups of experiments, the results of which are shown in Table 7. In this table, No. 1 stands for the basic W2V model before counter-fitting optimisation, No. 2 denotes the performance of the basic counter-fitting only relying on the synonym and antonym pairs of dictionaries, and No. 3 is the performance of our improved counter-fitting model, which incorporates prior knowledge including both lexicon-similarity and statistical similarity. The result No. 3 achieves 0.561/0.516 of ρ/r, which performs 0.25 (80.39%)/0.205 (65.92%) higher than No. 1, and 0.053 (10.43%)/0.025 (5.09%) higher than No. 2. It illustrates the effectiveness of the basic and improved counter-fitting methods.

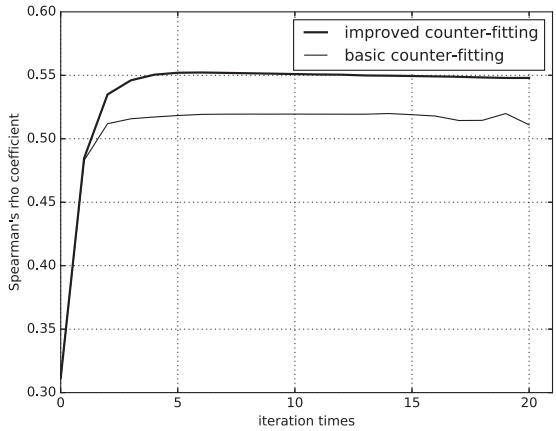

Fig. 6. The Learning Curves of the Basic Counter-fitting and Improved Counter-fitting Method.

Table 8. The Top 11 Word Pairs Change Better after Counter-Fitting

No.	Word Pair	Gold-standard Score	Before	After
1	睡/寐	8.4	5.375	10.00
2	赢/胜	9.8	6.616	9.888
3	行/可以	9.2	6.121	7.616
4	恶劣/坏	8.4	5.318	9.844
5	服气/心服口服	8.7	5.629	10.00
6	滚蛋/滚开	9.4	6.364	10.00
7	拖后腿/拉后腿	9.5	6.451	10.00
8	书籍/图书馆	3.4	7.589	6.582
9	利润/成本	3.2	7.694	6.923
10	互联网/因特网	9.5	6.895	9.999
11	计算机/电脑	9.9	7.397	10.00

Figure 6 shows the learning curves of the basic counter-fitting and improved counter-fitting methods within 20 epochs, from which we can see that both methods reach its own best performance after about 5 epochs and converges within 20 epochs. It can be clearly seen that our improved method gets higher performance than the basic one. The basic counter-fitting method shows a slight concussion in 15–20 epochs, while our improved method is more stable during 5–20 epochs.

For further analysis, we select 11 pairs as the samples that obviously improved by the counter-fitting method (See Table 8) and visualize their distribution after descending dimension of the vectors by Principal Components Analysis (PCA) algorithm (see Figure 7).

To analyse the reasons why counter-fitting is effective, we classify the disadvantages of the basic word embedding model into three categories:

(a) "Lack" of training data. There are lots of large-scaled corpus on the Internet, but they can't perfectly cover all kinds of language phenomena. For example, single-character words (Nos. 1–4), idioms (No. 5), and spoken language (Nos. 6 and 7).

(b) "Innate drawback" of word embedding. It is difficult for such a word embedding model to distinguish "similar" and "association." Taking the word pairs Nos. 8 and 9, for

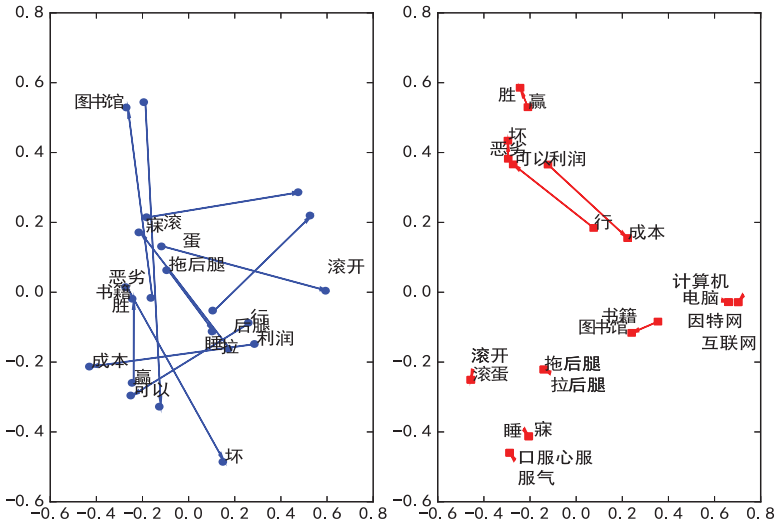

Fig. 7. Distribution of Word Pairs before and after Counter-fitting.

Table 9. Comparison between Single and Combination Models

No.	Strategy	ρ	r
1	Single: Cilin	0.421	0.472
2	Single: W2V	0.311	0.311
3	Single: web-pmi	0.262	0.286
4	Combination: Rule-based	0.518	0.519
5	Combination: Max	0.557	0.481
6	Combination: Counter-fitting	**0.561**	**0.516**

example, the two words of each of them are not quite conceptually similar, but have a strong association with each other due to high-frequency co-occurrence.

(c) "Cautious" about high score. According to the experience, the word embedding model hardly gives a very high similarity score between two words. A score of 0.6–0.7 is quite high relatively. For example, the cosine similarity of word pair (computer, laptop) is only approximate to 0.66, which is based on the English pre-trained word2vec model we mentioned above.

5.5 Comparison among Different Methods

To compare the performance of different methods on PKU-500 dataset, we list the representative results of 3 single methods and 3 combination methods in Table 9. Nos. 1–3, 5, 6 are the best results of different types of methods that we mentioned above and No. 4 is the the state-of-the-art result in the NLPCC-CWSM shared task participating systems using rule-based combination (Guo et al. 2016).

From test Nos. 1–3 we can see that the best lexicon-based method outperforms both the statistical method and the embedding method, which is most possibly due to the natural advantage of manual lexicons. Moreover, comparing Nos. 4–6 with Nos. 1–3, combination methods perform generally better than single models, because they can integrate more resources and have

Table 10. Spearman values of Different Methods on Chinese MC-30 and SemEval-2012

No.	Strategy	MC-30	SemEval-2012
1	Cilin	0.520	0.188
2	HowNet	0.682	0.224
3	CWordNet	0.044	0.138
4	W2V	0.697	0.375
5	web-pmi	0.468	0.359
6	Improved Counter-fitting	**0.731**	**0.411**

complementary advantages. Result No. 6 achieves 0.561/0.516 value of ρ/r, which performs 0.043 (8.3%) higher than No. 4 of ρ and 0.035 (7.28%) higher than No. 5 of r. It directly illustrates the effectiveness of our approach.

Furthermore, we compare our method with other word-embedding techniques that can also incorporate prior knowledge into consideration. Compared with Guo et al. (2016), which integrates different methods by handcrafted rules, our method can get the incorporated result by optimising word vectors rather than combining limited number of single results. Besides better performance, our model can generate a new set of word vectors that can be used into many NLP tasks. Compared with Mrkšić et al. (2016), our model uses lexicon-similarity constraints to find more pairs under the common circumstances and inject statistical similarity to consider information from search engines in addition to the manual resources. Besides, RSI item serves variable thresholds from a set of thresholds rather than a constant one, which can alleviate overfitting and increase the robustness of our system. Compared with retraining methods (Yu and Dredze 2014; Ono et al. 2015; Liu et al. 2015; Nguyen et al. 2016), counter-fitting methods fine-tune word vectors in the post-processing stage, which brings 2 advantages: (i) Time-saving, i.e., you can get the result in a very short time; (ii) Low coupling, i.e., you can easily choose whether incorporate knowledge or not. For example, it won't help a lot to distinguish synonym from antonym for noun phrase identification task. Among the post-processing methods, Mrkšić et al. (2016) has proved its advantages in performance and lightweight, which achieves the state-of-art result to the best of our knowledge. In this case, our experimental results are quite logical and convincing.

5.6 Experiments on Different DataSet

There are quite a few shared corpus that are commonly used as benchmark for English word similarity evaluation, wherein WordSim-353 and SimLex-999 are datasets of growing concern in recent research. However, there were no Chinese benchmark corpus until the newest PKU-500 dataset is released. To evaluate Chinese word similarity, researchers have translated English corpus into Chinese corpus for testing. For example, Zhu et al. (2016) have translated MC-30 into corresponding Chinese pairs according to part of speech and used the manual value of English data as the final score value. The score ranges from 0 to 1, and the higher the score is, the higher the similarity is. Another Chinese corpus is provided by SemEval-2012, where the Chinese word pairs are translated from English WordSim-353. Twenty human annotators are asked to give a similarity score between 0 and 5, and the final score is the average of different results.

To study on the performance for different corpus, we do experiments on Chinese MC-30 and SemEval-2012 dataset and compare the effectiveness of different methods. The results are listed in the Table 10.

Though we concentrate on Chinese word similarity measurement, the proposed method is not limited to Chinese. To demonstrate that the methods are not language specific, we add the

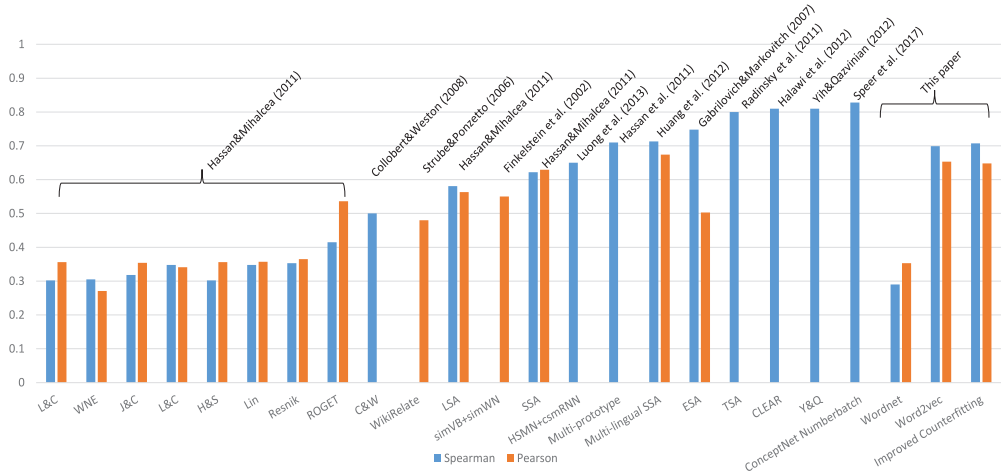

Fig. 8. Results of English Word Similarity Measurement based on WordSim-353.

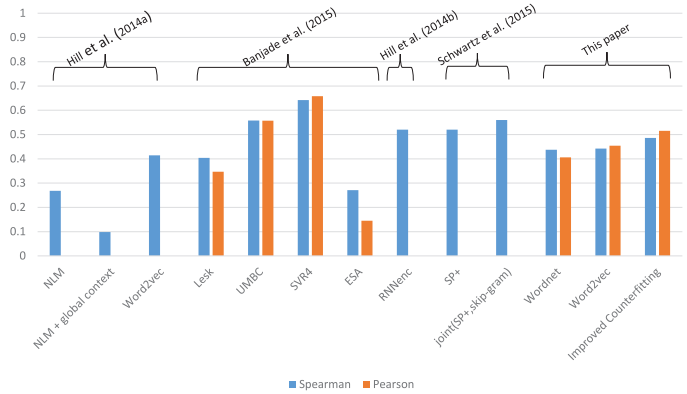

Fig. 9. Results of English Word Similarity Measurement based on SimLex-999.

experiments on English dataset, i.e., Word-353 and SimLex-999, to be more specific, we incorporate linguistic constraints from WordNet into Word2vec and compare with other research[2,3] on these two datasets. The results are shown in Figures 8 and 9, and we can see that counter-fitting method outperforms WordNet and Word2vec on both two datasets. However, its performance is still not the best in recent research.

Taking Speer et al's method for example, which reaches the highest performance on the WordSim-353 dataset. That method uses a multilingual graph of general knowledge called ConceptNet to calculate the similarity between words. The graph contains a lot of general knowledge and this may be the cause of the high performance of this method. Our method doesn't perform as well as that method and it is mainly because more semantic constrains need to be explored to enhance the word embedding.

[2]https://www.aclweb.org/aclwiki/index.php?title=WordSimilarity-353_Test_Collection_(State_of_the_art).
[3]https://aclweb.org/aclwiki/index.php?title=SimLex-999_(State_of_the_art).

Banjade et al's method performs best on the SimLex-999 dataset, and this method uses Support Vector Regression (SVR) to combine the results of different methods. High final performance depends on the high performances of combined single methods, so the single methods used by our method may need to be improved to reach a better performance.

6 CONCLUSION

This article proposes a three-stage framework for the Chinese Word Similarity Measurement task, which highly emphasises on incorporation of more prior knowledge and word embedding. In the first stage, we utilise retrieval techniques to crawl the contexts of word pairs from web resources. Next, we conduct comparable experiments using three kinds of single similarity measurements including lexicon similarities, statistical similarities and embedding-based similarities. Finally, we investigate simple combination strategies depending on math operations and the counter-fitting combination strategy based on a optimisation algorithm.

Instead of using only one corpus, we train different word embedding models on different corpus, define the WSS and use it to select the best model. On the basis of the basic counter-fitting method, this article proposes the improved counter-fitting method, which takes lexicon similarities and statistical similarities into consideration, and experiment results show that the improved method is more effective and more stable than the basic one.

To summarise our experiments, lexicon-based methods outperform both the statistical methods and the embedding-based method. It is natural that the similarity scores calculated through lexicon methods are close to manually labeled ones for scale limited dataset. Moreover, combination methods perform generally better than single methods, since they can integrate more resources and have complementary advantages. Furthermore, the counter-fitting method outperforms simple combination methods in terms of effectiveness and stability.

Our final results on the PKU-500 dataset are 0.561/0.516 of Spearman/Pearson rank correlation coefficient, which outperform the state-of-the-art performance to the best of our knowledge. Our system also performs well on Chinese MC-30 and SemEval-2012 datasets according to experiment results, which proves its transferability. Besides, our system is not language-specific and can be applied to other languages, e.g., English.

ACKNOWLEDGMENTS

The authors thank the anonymous reviewers for their constructive comments and NLPCC-ICCPOL 2016 conference for providing the benchmark dataset.

REFERENCES

Mostafa Ghazizadeh Ahsaee, Mahmoud Naghibzadeh, and S. Ehsan Yasrebi Naeini. 2014. Semantic similarity assessment of words using weighted wordnet. *Int. J. Mach. Learn. Cybernet.* 5, 3 (2014), 479–490.

Danushka Bollegala, Yutaka Matsuo, and Mitsuru Ishizuka. 2007. Measuring semantic similarity between words using web search engines. In *Proceedings of the International Conference on the World Wide Web (WWW'07)*. 757–766.

Zhigang Chen, Wei Lin, Qian Chen, Xiaoping Chen, Si Wei, Hui Jiang, and Xiaodan Zhu. 2015. Revisiting word embedding for contrasting meaning. In *Proceedings of the Association for Computational Linguistics Conference (ACL'15)*. 106–115.

Zhendong Dong and Qiang Dong. 2006. *HowNet and the Computation of Meaning*. World Scientific, Singapore.

Mengjia Fan, Yangsen Zhang, and Jiayuan Li. 2015. Word similarity computation based on HowNet. In *Proceedings of Fuzzy Systems and Knowledge Discovery*. 1487–1492.

Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard H. Hovy, and Noah A. Smith. 2015. Retrofitting word vectors to semantic lexicons. In *Proceedings of North American Chapter of the. Association for Computational Linguistics: Human Language Technologies (HLT-NAACL'15)*. 1606–1615.

Shaoru Guo, Yong Guan, Ru Li, and Qi Zhang. 2016. Chinese word similarity computing based on combination strategy. In *Proceedings of the Conference on Natural Language Processing and Chinese Computing (NLPCC'16)*. 744–752.

Derrick Higgins. 2005. Which statistics reflect semantics? Rethinking synonymy and word similarity. *Linguist. Evid. Empir. Theoret. Comput. Perspect. Studies Generat. Grammar* 85 (2005), 265–284.

Felix Hill, Roi Reichart, and Anna Korhonen. 2016. Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *Comput. Linguist.* (2016).

Ignacio Iacobacci, Mohammad Taher Pilehvar, and Roberto Navigli. 2015. SensEmbed: Learning sense embeddings for word and relational similarity. In *Proceedings of the Association for Computational Linguistics Conference (ACL'15)*. 95–105.

Chunxia Liang, Yanqiu Shao, and Jing Zhao. 2013. Construction of a Chinese semantic dictionary by integrating two heterogeneous dictionaries: TongYiCi cilin and HowNet. In *Proceedings of the Workshops on Web Intelligence and Intelligent Agent Technologies (IAT'13)*. 203–207.

Quan Liu, Hui Jiang, Si Wei, Zhenhua Ling, and Yu Hu. 2015. Learning semantic word embeddings based on ordinal knowledge constraints. In *Proceedings of the Association for Computational Linguistics Conference (ACL'15)*. 1501–1511.

Qun Liu and Sujian Li. 2002. Word similarity computing based on how-net. *Comput. Linguist. Chinese Lang. Process.* 7, 2 (2002).

Jiaju Mei, Yiming Zhu, Yunqi Gao, and Hongxiang Yin. 1983. *Tongyici Cilin*. Shanghai Lexicon Publishing Company, Shanghai.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. In *Proceedings of Workshop at the International Conference on Learning Representations (ICLR'13)*. 422–431.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS'13)*. 3111–3119.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gasic, Lina Maria Rojas-Barahona, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve J. Young. 2016. Counter-fitting word vectors to linguistic constraints. In *Proceedings of the North American Chapter of the. Association for Computational Linguistics: Human Language Technologies (HLT-NAACL'16)*. 142–148.

Kim Anh Nguyen, Sabine Schulte im Walde, and Ngoc Thang Vu. 2016. Integrating distributional lexical contrast into word embeddings for antonym-synonym distinction. In *Proceedings of the Association for Computational Linguistics Conference (ACL'16)*.

Masataka Ono, Makoto Miwa, and Yutaka Sasaki. 2015. Word embedding-based antonym detection using thesauri and distributional information. In *Proceedings of the North American Chapter of the. Association for Computational Linguistics: Human Language Technologies (HLT-NAACL'15)*. 984–989.

Jiahuan Pei, Cong Zhang, Degen Huang, and Jianjun Ma. 2016. Combining word embedding and semantic lexicon for Chinese Word Similarity Computation. In *Proceedings of the Conference on Natural Language Processing and Chinese Computing (NLPCC'16)*. 766–777.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the Conference on Empirical Methods for Natural Language Processing (EMNLP'14)*. 1532–1543.

Sascha Rothe and Hinrich Schütze. 2015. AutoExtend: Extending word embeddings to embeddings for synsets and lexemes. In *Proceedings of the Association for Computational Linguistics Conference (ACL'15)*. 1793–1803.

Cilibrasi Rudi and Paulf Vitanyi. 2007. The google similarity distance. *IEEE Trans. Knowl. Data Eng.* 19, 3 (2007), 370–383.

Wang Shan and Bond Francis. 2013. Building the Chinese open wordnet (COW): Starting from core synsets. In *Proceedings of the 6th International Joint Conference on Natural Language Processing*. 10–18.

Jiule Tian and Wei Zhao. 2010. Words similarity algorithm based on tongyici cilin in semantic web adaptive learning system. *J. Jilin Univ. (Info. Sci. Ed.)* 28, 6 (2010), 602–608.

Peter Turney. 2001. Mining the web for synonyms: PMI-IR versus LSA on TOEFL. In *Proceedings of the 12th European Conference on Machine Learning*. 491–502.

Peter D. Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *J. Artific. Intell. Res.* 37 (2010), 141–188.

Siying Wu and Yangyang Wu. 2010. Chinese and english word similarity measure based on Chinese wordnet. *J. Zhengzhou Univ. (Natural Sci. Ed.)* 2 (2010), 017.

Yunfang Wu and Wei Li. 2016. Overview of the NLPCC-ICCPOL 2016 shared task: Chinese word similarity measurement. In *Proceedings of the Conference on Natural Language Processing and Chinese Computing (NLPCC'16)*. 828–839.

Yueh-Cheng Wu and Shu-Kai Hsieh. 2010. PyCWN: A python module for Chinese wordnet. In *Proceedings of the 23rd International Conference on Computational Linguistics: Demonstrations*. Demonstrations Volume, 5–8.

Mo Yu and Mark Dredze. 2014. Improving lexical embeddings with semantic knowledge. In *Proceedings of the Association for Computational Linguistics Conference (ACL'14)*. 545–550.

Peiying Zhang, Zhanshan Zhang, and Weishan Zhang. 2013. An approach of semantic similarity by combining hownet and cilin. In *Proceedings of GreenCom/iThings/CPScom*. 1638–1643.

Jun Zhao, Shuanzhu Hu, and Xinghua Fan. 2009. Word similarity computation based on word link distribution. *J. Chongqing Univ. Posts Telecommun. (Natural Sci. Ed.)* 21, 4 (2009), 528–532.

Xinhua Zhu, Runcong Ma, Liu Sun, and Hongchao Chen. 2016. Word semantic similarity computation based on hownet and cilin. *J. Chinese Info. Process.* 30, 4 (2016), 29–36.

Received January 2017; revised November 2017; accepted January 2018