Morpheme-Enhanced Spectral Word Embedding

Jiawei Liu
University of Science and Technology of China, China
ustcljw@mail.ustc.edu.cn

***Abstract*—Traditional word embedding models only learn word-level semantic information from corpus while neglect the valuable semantic information of words' internal structures such as morphemes. To address this problem, the goal of this paper is to exploit the morphological information to enhance the quality of word embeddings. Based on spectral method, we propose two word embedding models: Morpheme on Original view and Morpheme on Context view (MOMC) and Morpheme on Context view (MC). In vector space of MOMC and MC, both semantic-similar words and morphological-similar words locate near with each other. In experiments, MOMC, MC and the baselines are tested on word similarity and sentiment classification. The results show that our models outperform all comparative baselines on six datasets of word similarity and win the first on sentiment classification as well. Based on a large German corpus, we also inspect the ability of word embeddings to process morpheme-rich languages by using German word similarity task. The result shows that MOMC and MC significantly outperform the baselines more than 5 percentage on one dataset and nearly 4 percentage on the other. These impressive improvements demonstrate the effectiveness of our models in dealing with morpheme-rich languages like German.**

***Index Terms*—Spectral Method, Morphological Information, CCA, Random SVD, Word Embedding**

I. Introduction

Nowadays, Natural Language Processing (NLP) has been an important part of Artificial Intelligence (AI) for its effectiveness on many NLP tasks such as information retrieval [1] and text classification [2]. As we all know, computer can not directly deal with natural language due to the abstract semantic structure contained in corpus. To solve this problem, researchers developed a lot of models to represent words into vector space, which is also called word embedding.

Traditional word embedding models are divided into two main branches. One is based on neural network like Continuous Bag Of Words (CBOW) and Skip-gram [3]. The other is based on the matrix factorization such as the models in [4], which is also called spectral method. In general, the neural network-based models are famous for their robust performance. Nevertheless, these models with a large amount of parameters are time-consuming and tuning parameters requires enough experiences. Besides, CBOW and Skip-gram are also meaning-ambiguous methods, which lack theoretical explanation. On the contrary, the spectral models have their own advantages such as being on a theoretically provable basis and able to accelerate the calculation. Nevertheless, the models mentioned above are still word-level models and ignore many useful internal information of a word such as the morphemes. In English, a morpheme is the smallest unit and has some linguistic meanings. Mostly, morphemes can be divided into three categories. The prefix is one of the morphemes which has an affix placed before the stem of a word. Usually, adding a prefix before a word changes the meaning of the word but assigns similar semanteme. For example, putting *"uni-"* before the word *"form"* changes the sense of *"style"*, but all the words beginning with *"uni-"* will be endowed with the meaning of "single". The suffix is another morpheme which is an affix placed after the stem of a word. For instance, *"-ed"* is always added to the tail of the word to denote the past tense. Additionally, the root is the primary lexical unit of a word without prefix or suffix before or after it. In traditional models, the meaningful morphological information is abandoned.

Motivated by recent work, which exploits the internal structures to improve Chinese word embeddings [5], [6], in this paper, we utilize the spectral technique to incorporate the morphological information into word embedding training process. In our models, the word information, morphological information and context information are transformed into three matrices, respectively. Based on these matrices, we propose two models including Morpheme on Original view and Morpheme on Context view (MOMC) and Morpheme on Context view (MC) by using the Canonical Correlation Analysis (CCA). In MOMC, we combine the morphological information matrix with not only the word information matrix but the context information matrix. Then, CCA is conducted on these two new matrices. In MC, the morphological information matrix is only combined with the context information matrix. Then, we conduct CCA on the new combination matrix and the initial word information matrix.

In experiments, MOMC and MC together with the baselines are tested on word similarity and sentiment classification. The results demonstrate the advantages of our models, which outperform the baselines on six datasets of word similarity. Besides, MOMC and MC also achieve the best performance on sentiment classification. In order to evaluate the ability of our methods to deal with the morpheme-rich languages, we train all models on a German corpus and also test them on word similarity task. The result indicates that MC and MOMC significantly outperform the baselines on all the datasets and even get more than 5 percentage advantage on RG-65-German dataset. The performance of our models convinces us that incorporating morphological information into word embedding can generate a good structure in vector space and enhance the quality of word embeddings.

II. Related Work and background

As we mentioned above, there are a lot of word embedding models. In this section, we are going to give a brief review

DOI reference number: 10.18293/SEKE2017-105

of CBOW [3] and OSCCA (One Step CCA) [4], which are chosen as the baselines in experiments.

CBOW With a slide window, CBOW [3] utilizes the context words in the window to predict the target word. Given a sequence of tokens $T = \{t_1, t_2, \cdots, t_n\}$, the goal of CBOW is to maximize the following average log probability equation:

$$L = \frac{1}{n}\sum_{i=1}^{n} \log\ p(t_i|context(t_i)), \tag{1}$$

where $context(t_i)$ means the context information of t_i in the slide window. Based on *Hierarchical Softmax* and *Negative Sampling* [7], Equation (1) can be solved efficiently.

OSCCA Based on CCA, Dhillon et al. proposed a spectral word embedding model named OSCCA, which is proven to be effective in [4]. For a sequence of tokens $T = \{t_1, t_2, \cdots, t_n\}$, in OSCCA, we firstly need to construct two matrices including word information matrix W and context information matrix C. Building of these matrices will be introduced in the following section. Then, we directly do CCA on these matrices. As we all know, the goal of CCA is to find a pair of projection matrices Φ_w and Φ_c such that the correlation between the projection of W onto Φ_w and C onto Φ_c is maximized. Based on *Eigendecomposition*, the solution of Φ_w and Φ_c is as follows.

$$\begin{aligned} \Sigma_{11}^{-1}\Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}\Phi_w &= \lambda\Phi_w \\ \Sigma_{22}^{-1}\Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12}\Phi_c &= \lambda\Phi_c \end{aligned} \tag{2}$$

where Σ_{11} is the covariance matrix of W, Σ_{12} is the covariance matrix between W and C, Σ_{22} is the covariance matrix of C, Σ_{21} if the covariance matrix between C and W. Nevertheless, this solution is time-consuming and memory-consuming due to many large sparse matrix multiplications, which origins from the huge vocabulary size. To solve this problem, in [4], they demonstrated that the solution of OSCCA can be transformed into an equivalent equation.

$$\Sigma_{11}^{-1/2}\Sigma_{12}\Sigma_{22}^{-1/2} = \Phi_w\Lambda\Phi_c^T \tag{3}$$

where (Φ_w, Φ_c) are the left and right singular vectors and Λ is the diagonal matrix of singular values. Hence, $\Phi_w^{proj} = \Sigma_{11}^{-1/2}\Phi_w$ and $\Phi_c^{proj} = \Sigma_{22}^{-1/2}\Phi_c$. In OSCCA [4], the projection Φ_w^{proj} is viewed as the word embedding. For clarity, we define the anterior CCA progress as $(\Phi_w^{proj}, \Phi_c^{proj}) \equiv CCA(matrix_1, matrix_2)$ and use it if we want to do CCA on a pair of matrices. Due to the effectiveness of the second solution, all spectral word embeddings are trained based on Equation (3) in this paper.

III. Incorporating Morphological Information into Word Embedding

Based on spectral technique, we propose two word embedding models: Morpheme on Original view and Morpheme on Context view (MOMC) and Morpheme on Context view (MC). The objective of these models is to incorporate morphological information to improve the quality of word embeddings so that both semantic-similar words and morphological-similar words can group together in vector space. Both MOMC and MC are built based on three information matrices including word information matrix, context information matrix and morphological information matrix. For clarity, we introduce how to build these matrices firstly.

A. Methodology of Building Information Matrices

1) Notation: We define the word information matrix as $W \in \mathbb{R}^{n\times p}$, morphological information matrix as $M \in \mathbb{R}^{n\times(np+ns+nr)}$, left context information matrix as $L \in \mathbb{R}^{n\times cp}$, right context information matrix as $R \in \mathbb{R}^{n\times cp}$ and whole context information matrix as $C \in \mathbb{R}^{n\times 2cp}$ where n is the length of the word sequence, p is the number of the vocabulary, np is the number of the prefix, nr is the number of the root, ns is the number of the suffix and c is context window size.

2) Word Information Matrix: The word information is mapped into a sparse matrix W. Given a corpus $\{t_1, t_2, t_3, ..., t_n\}$, the vocabulary is $\{null, v_1, v_2, ..., v_p\}$ where $n \geq p$. Every word is represented as a one-hot vector in which $w_{ij} = 1$ means word t_i is in the j_{th} position of the vocabulary. In practice, the vocabulary is so huge that the computational cost is rather high. Since quite a number of words only turn up at a low frequency due to the *zipfian* distribution, we set a threshold of the frequency of the words. If the word's frequency is under that threshold, the word will be assigned to the first column *null*. More detail is give in Equation (4).

$$w_{ij} = \begin{cases} 1, \ when\ t_i = v_j\ or\ t_i = null \\ 0, \ otherwise \end{cases} \tag{4}$$

3) Morphological Information Matrix: In this paper, morphemes are divided into prefix, suffix and root. Therefore, the whole morphological information matrix M can be represented as the concatenation of three separate matrices. We assume that $root = \{null, r_1, r_2, \cdots, r_{nr}\}$, $prefix = \{null, p_1, p_2, \cdots, p_{np}\}$, and $suffix = \{null, s_1, s_2, \cdots, s_{ns}\}$. If a word has a certain morpheme, the corresponding column position should be set as 1. A constraint is that in the whole matrix, every morpheme should be active at least once. If there is no feature in the morpheme list, the first column *null* will be set as 1. We define function $match(t_i, a_j) \in \{1, 0\}, a_j \in \{\ prefix \cup root \cup suffix\}$ to simplify our description. The function means that if word t_i contains morpheme a_j, it will be set as 1 or 0 otherwise. Hence, we can utilize Equation (5) to summarize the construction of morphological information matrix.

$$m_{ij} = \begin{cases} 1, \ when\ match(t_i, a_j) = 1 \\ 0, \ otherwise \end{cases} \tag{5}$$

4) Context Information Matrix: Context information matrix C is composed of the left context matrix L and the right context matrix R. For a target word, the construction of the left context matrix is built based on the contextual words, which are located in the left-side of the target word. Building of the right context matrix is associated with the right-side contextual words of the target word. For a single token in corpus, the corresponding positions of its context words in the left context matrix and right context matrix will be set as 1. If the context word is not in the vocabulary, then the corresponding position of $null$ will be set as 1. We define a function $context(t_i, v_j, k),\ k \in [1, c],\ j \in [1, c \times p]$. When the k_{th} context word of t_i is equal to v_j, this function is equal to 1 or 0 otherwise. The mathematical description of

Algorithm 1 MOMC model

function MOMC (D, M, h, k)
Input: huge size corpus D
morpheme set M
context window size h
dimension of word embedding k
Output: word vector ϕ_w^{proj}
Initialization: Initialize expanded word information matrix W_e and context information matrix C_e according to the input parameters.
Calculate the approximate correlated matrix
$C_{ww} = W_e^T W_e$
$C_{wc} = W_e^T C_e$
$C_{cc} = C_e^T C_e$
Do SVD on
$C_{ww}^{-1/2} C_{wc} C_{cc}^{-1/2}$
Get *right* singular matrix ϕ_{w_e}
Get the eigenword $\phi_{w_e}^{proj} = C_{ww}^{-1/2} \phi_{w_e}$
Extract the vocabulary part skipping morpheme part ϕ_w^{proj}
Return ϕ_w^{proj}

left context matrix L is in Equation (6). Right context matrix can be established in the same way.

$$l_{ij} = \begin{cases} 1, \ when\ context(t_i, v_j, k) = 1 \\ 0, \ otherwise \end{cases} \tag{6}$$

B. Morpheme on Original view and Morpheme on Context view (MOMC)

In MOMC, the morpheme information is viewed as a unit feature. It means that all words including the target words and context words need to be featured by morpheme information. MOMC uses the expanded word matrix $W_e = [W\ M]$ and expanded context matrix $C_e = [[L\ M]\ [R\ M]]$ as the input. [ ] means concatenating two matrices upon row direction. Like what we did above, we utilize the following equation to describe the MOMC in a mathematic way.

$$(\Phi_w^{proj}, \Phi_c^{proj}) = CCA(W_e, C_e) \tag{7}$$

The detailed algorithm is described as the pseudocode in Algorithm 1 shown below. This model will lead words with similar morphemes to locate closer than that in the original vector space. Due to the fact that morphemes have some semantic roles, the method encoding the morphological information into the word vectors will be useful to identify words' semantic role and morphological role. Hence, it will have a better performance than *OSCCA* ideally.

C. Morpheme on Context view (MC)

This method is similar to *MOMC* which also exploit the internal structures of words. Nevertheless, this method deals with the morphological information in an entirely different way. In MC, the morphological information is viewed as a global feature rather than unit feature in MOMC, which acts as a unique character about the corpus. We don't represent words into the other space but take their context words and morphemes as the context. The solution of this model is also based on *CCA* model. The pseudocode of MC is similar to Algorithm 1. The only difference is the input matrix. For brevity, we don't give the algorithm of MC. The equation is like the MOMC which is shown as follows:

$$(\Phi_w^{proj}, \Phi_c^{proj}) = CCA(W_e, C_e) \tag{8}$$

where $W_e = W$ and $C_e = [L\ R\ M]$.

D. Random SVD for CCA

In [4], the authors discussed how to calculate Σ_{11}, Σ_{12} and Σ_{22} in Equation (3). Based on *zipfian* distribution, in OSCCA, the empirical expectations of all words are set as zero so that matrices $W^T W$, $W^T C$ and $C^T C$ can be utilized to stand for Σ_{11}, Σ_{12} and Σ_{22}, respectively. As we all know, a word and its morpheme are occurred simultaneously, which means the distribution of morpheme in large corpus follows the *zipfian* distribution as well. Hence, the empirical expectations of all morphemes are also set as zero so that we can utilize the same way to calculate the Σ_{11}, Σ_{12} and Σ_{22} in MOMC and MC.

As we mentioned above, getting word embeddings needs to train a huge corpus. Doing decomposition on so vast matrix still needs a great deal of time and memory even though the computing power has become more powerful. Hence, recent advances in SVD algorithms are necessary to be taken to solve the huge matrix decomposition. In [8], Halko et al. gave a powerful tool which uses random projections to do SVD on large matrices. Basic idea of the method is to find a lower dimensional basis for large matrix and then to calculate the singular vectors in this lower dimensional basis. For a large input $m \times n$ matrix A, the first stage is to find a basis Q, based on which A can be represent as following equation.

$$A \approx QQ^T A \tag{9}$$

where Q has few columns. When calculate Q, we firstly generate a Gaussian test matrix Ω with dimensions of $n \times (k+l)$ where k is a target number of singular vectors and l is the extra basis vector ranging from 0 to k. By multiplying alternately with A and A^T, we secondly generate a matrix $Y = (AA^T)^q A\Omega$ where q is another parameter from input. Lastly, matrix Q whose columns form an orthonormal basis for the range of Y will be constructed. The second stage of this method is to do SVD algorithm on A with the help of Q. In this stage, a matrix $B = Q^T A$ needs to be calculated firstly. Then, SVD algorithm is conducted on the small matrix $B = U' \Sigma V^T$. Finally, the left singular matrix U of A can be approximately set as $U = QU'$. More theoretical demonstrations are illustrated in [8].

E. Complexity Analysis

The solutions of our models including MOMC and MC are based on the random SVD algorithm which is introduced in the previous subsection. Random SVD algorithm as the most important part of our models consumes most of the training time. For a large matrix A with the dimensions $m \times n$, the first stage needs to cost $O(mns)$ time to generate a low dimension basis Q with the dimensions $m \times s,\ s \ll n$. In [4], it is reported that the computational complexity of OSCCA is $O(p^2 cs)$. For MC, the cost of time will be $O(p(pc + np + ns + nr)s)$. Because np, ns and nr stand for the counts of morphemes

which are the constant numbers and much more smaller than pc, $O(p(pc+np+ns+nr)s)$ is equal to $O(p^2cs)$. Obviously, the complexity of MOMC is also $O(p^2cs)$. The notations p and c represent the vocabulary size and context window size which have been introduced. Compared with OSCCA, it is obvious that our models have same computational complexities.

IV. Experiments

In this section, we test MOMC, MC and the baselines on word similarity and sentiment classification. Moreover, parameter analysis is given in the end of this section. For clarity, some experimental settings are introduced firstly.

A. Experimental Settings

In this paper, all word embeddings are trained based on the news corpus of 2009, which is listed on the 2013 ACL Workshop on Machine Translation[1] and also used in [9]. We collect the morpheme from the website[2] and get 90 prefixes, 241 roots and 64 suffixes.

The existing word embedding model OSCCA [4] and CBOW [3] are chosen to compare with MOMC and MC. For a fair and unbiased comparison, all word embeddings are trained in the same condition. The size of the context window and the dimension of word embedding are set as 2 and 200, which are the same as the settings in [4]. CBOW is trained by using the source code[3]. For efficiency, the negative sampling algorithm is chosen to solve CBOW [7]. To generate the word embedding of OSCCA, we utilize the java toolkit, which is released in Dhillon's github[4]. Moreover, we modify the source code of this toolkit and generate our embeddings.

B. Word Similarity

This experiment is utilized to evaluate the ability of word embeddings to capture semantic information from large corpus. The dataset is composed of two parts: word pairs and human score. We need to calculate the similarities of word pairs firstly and then measure the correlation between the similarities and human score. The similarities of word pairs are measured using cosine distance. We use *Spearman's rank correlation coefficient (ρ)* to evaluate the correlation between word similarities and human score. Apparently, bigger ρ means better performance. In this task, we utilize 8 widely used benchmarks. RG-65 [10] has 65 noun pairs. MTurk-287 and MTurk-771 [11] contain 287 and 771 English word pairs, respectively. RW-STANFORD [12] has a large number of rare words with similarity score. WS-353-ALL [13] contains 353 pairs of English with human similarity ratings. WS-353-REL and WS-353-SIM are annotated based on WS-353-ALL in [14]. MEN-TR-3k [15] owns 3000 word pairs, which frequently turn up in a large web corpus.

The results are shown in Table I. It is obvious that our methods significantly outperform the comparative baselines. MOMC wins the first on four dataset and MC performs the best on other two datasets. The advantages of MOMC and MC are corresponding to our expectation. Obviously, more semantic information means better performance. The baselines are word-level models and neglect the internal semantic information. On the contrary, MOMC and MC exploit the morphological information and capture more semantic information, which interprets the best performance of our methods.

TABLE I
Results of Word Similarity and Sentiment Classification. "SA" stands for Sentiment Classification. The numbers in bold mean the best answers.

	OSCCA	MOMC	MC	CBOW
MTurk-287	0.5664	**0.5847**	0.5657	0.5761
RG-65	0.5674	0.5734	0.5734	**0.6042**
RW-STANFORD	0.4966	0.4798	**0.5107**	0.4961
WS-353-ALL	0.5387	**0.5742**	0.5634	0.5675
WS-353-REL	0.4486	**0.4607**	0.4338	0.4420
WS-353-SIM	0.6752	0.6827	**0.7072**	0.6970
MEN-TR-3k	**0.6562**	0.6339	0.6562	0.6346
MTurk-771	0.5324	**0.5532**	0.5338	0.5478
SA	0.7086	0.7132	**0.7256**	0.7193

C. Sentiment Classification

This experiment is conducted in a similar way as we find in [16]. The average of the word embeddings of a given sentence is utilized as features in a logistic regression model for classification. In this task, we utilize the annotated sentences with sentiment labels by *treebank model* introduced in [17]. We report the accuracy in Table I.

The results show that MC and MOMC outperform the baselines as well. Actually, the sentiment of a word is related to the morphological information. For instance, prefix "dis", "un" and "in" have negative meanings. By incorporating morphological information, the morpheme-similar words will group together in vector space. Hence, our better performance may stems from this property.

TABLE II
Performance on Morpheme-rich languages. The numbers in bold mean best performances.

	OSCCA	**MC**	**MOMC**
RG-65-German	58.63	**63.98**	62.36
WS-353-German	59.94	**63.45**	63.21

D. Morpheme-Rich Language Test

In order to measure the ability of our models when applied to some morpheme-rich languages like German, we train OSCCA, MC and MOMC based on the 2009 news German corpus, which is also from the 2013 ACL Workshop on Machine Translation. Then, all word embeddings are tested on word similarity task as well. We utilize Google Translate

[1] http://www.statmt.org/wmt13/translation-task.html
[2] https://msu.edu/∼defores1/gre/roots/gre_rts_afx1.htm
[3] https://github.com/dav/word2vec
[4] https://github.com/paramveerdhillon/swell

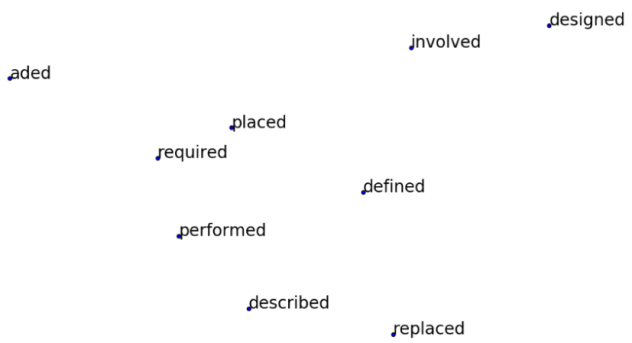

Fig. 1. Words ending with the suffix of "ed" are grouped together in vector space of MOMC.

to translate the golden standard WordSim-353 and RG-65 into German and named them WS-353-German and RG-65-German, respectively.

The results in Table II show that our methods beat OSCCA in a quite much sense. MC outperforms OSCCA more than 5 percentage on RG-65-German and nearly 4 percentage on WS-353-German. In addition, MOMC also performs well on both datasets. Hence, our methods incorporating morpheme information seem to have some positive effects on the quality of word embedding.

E. Word Structure

The word structure of MOMC is illustrated in Fig.1 by using t-SNE. From these figures, we find that the morphological-similar words locate near with each other, which is consistent with our expectation. Moreover, we upload a particular part of the embeddings to the website[5] providing word vector evaluation service by [18]. The result in Fig.2 shows that some male and female related words have exciting semantic structures. For example, in the top left of this picture, tokens of "he", "his", "she" and "her" sit nearby. Furthermore, it also indicates some analogy meaning that the vector from "her" pointing to "his" is parallel to that from "she" pointing to "he". Nevertheless, there are also some weaknesses in Fig.2. For instance, words "mother" and "mom" are semantic-related words but far from each other. As we all know, the quality of word embeddings have a strong connection with the corpus. Obviously, we can not capture all patterns because of some outliers in corpus, which can do harm to the quality of word embeddings.

F. Parameter analysis

In our models, there are several parameters related to the quality of the word embeddings including token size, context window size and the dimension of embedding. We are going to do parameter analysis based on the three parameters in the following parts.

[5]http://www.wordvectors.org/index.php

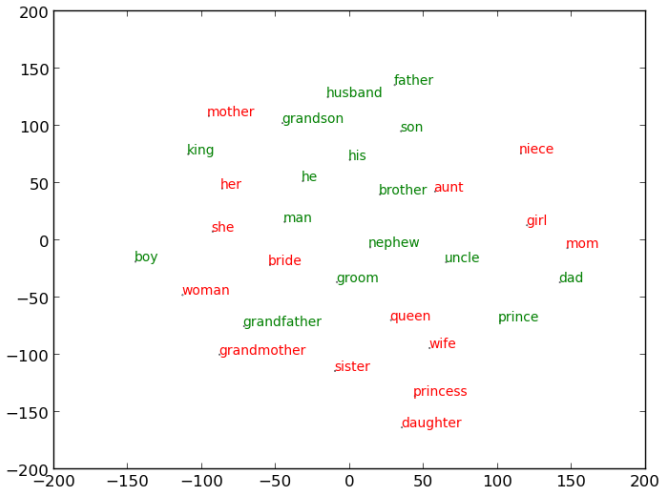

Fig. 2. The position of male and female related words in vector space of MOMC.

1) Effect of token size: In order to analyze the effect of token size, OSCCA, MOMC and MC are trained on the one-fifth, two-fifth, three-fifth, four-fifth and five-fifth of the corpus we mentioned before. All word embeddings are tested on the word similarity by using the golden standard Wordsim-353. The results are illustrated in Fig.3.

In Fig.3, the performance of all the word embeddings shows an obvious ascending tendency, which means larger corpus will generate better word embedding. At the beginning, MC and MOMC perform worse than OSCCA. However, with the increasing of token size, MC and MOMC outperform OSCCA and have a more stable increased tendency.

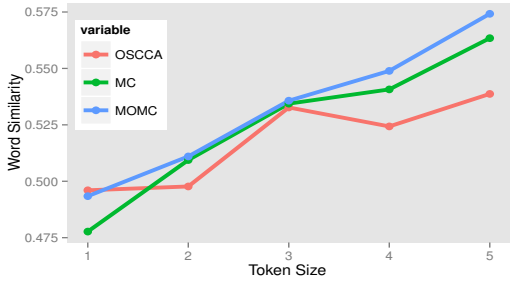

Fig. 3. Effect of token size based on word similarity test by using Wordsim-353.

2) Effect of Context Window Size: In this part, the context window size ranges from 1 to 5. All models are evaluated on the word similarity task by using Wordsim-353 as well. The result is illustrated in Fig.4.

From Fig.4, an ascending tendency is very clear accompanying with the changing of window size, which accords with our expectation. Larger window size means more semantic information. It seems that MC and MOMC are more sensitive to window size. At beginning, MC and MOMC perform worse than OSCCA. However, they outperform OSCCA on larger window size. We can not roughly conclude that larger window size means better performance because we don't test the

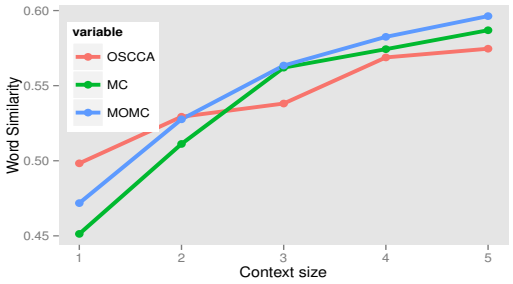

Fig. 4. Effect of context size based on word similarity test by using Wordsim-353.

models by setting some extreme condition due to the limitation of computing resource. In small scope, it is concluded that larger window size generates better word embedding.

3) Effect of Word Embedding Dimension: The word embeddings of OSCCA, MOMC and MC are trained by dynamically setting the dimension increasing from 50 to 250 step by 50. The results on word similarity are shown in Fig.5.

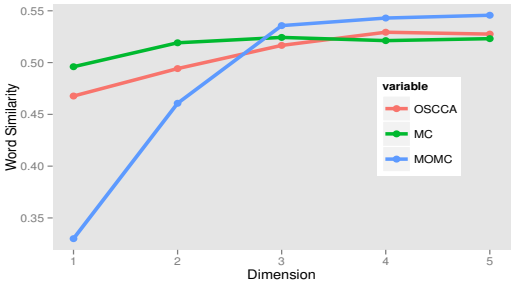

Fig. 5. Effect of dimension based on word similarity test by using Wordsim-353.

In Fig.5, all models follow a tendency of ascending before and stable later. It seems that MOMC is very sensitive to dimension and starts with a low position. An empirical explanation is given to illustrate this phenomenon. From anterior section, it is easy to find that the matrices used in MOMC for random SVD decomposition are much larger than other models which means if the dimension is set too low, we may lose more information than other models when the random SVD is conducted. It convinces us that the dimension of MOMC should be set larger than other models. From the figure, it is fine to choose dimension of word embedding from 200 to 250.

V. Conclusion

Traditional word embedding models neglect meaningful internal semantic structures such as morphemes when extracting semantic and syntactic information from large corpus. To address this problem, we propose two models including MOMC and MC by exploiting the morphological information based on spectral methods. In MOMC, the morphological information is viewed as a unit feature. On the contrary, the morphological information is viewed as a global feature and utilized to supplement the context information during the training process of MC. The experiments' results on word similarity and sentiment classification show that MOMC and MC outperform the baselines on both tasks. The morpheme-rich language test also demonstrates the effectiveness of MOMC and MC, which outperform OSCCA to a great extent. In summary, both semantic-similar words and morphological-similar words have a trend to group together in vector space of MOMC and MC. The property can not only improve the semantic similarity but also elevate morphological similarity of word embeddings.

References

[1] C. D. Manning, P. Raghavan, H. Schütze *et al.*, *Introduction to information retrieval*. Cambridge university press Cambridge, 2008, vol. 1, no. 1.

[2] Y. Liu, Z. Liu, T.-S. Chua, and M. Sun, "Topical word embeddings," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2015, pp. 2418–2424.

[3] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[4] P. S. Dhillon, D. P. Foster, and L. H. Ungar, "Eigenwords: Spectral word embeddings," *The Journal of Machine Learning Research*, vol. 16, no. 1, pp. 3035–3078, 2015.

[5] X. Chen, L. Xu, Z. Liu, M. Sun, and H. Luan, "Joint learning of character and word embeddings," in *Proceedings of the International Joint Conference on Artificial Intelligence*, 2015, pp. 1236–1242.

[6] J. Xu, J. Liu, L. Zhang, Z. Li, and H. Chen, "Improve chinese word embeddings by exploiting internal structure," in *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016, pp. 1041–1050.

[7] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proceedings of the Advances in Neural Information Processing Systems*, 2013, pp. 3111–3119.

[8] N. Halko, P.-G. Martinsson, and J. A. Tropp, "Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions," *SIAM review*, vol. 53, no. 2, pp. 217–288, 2011.

[9] Y. Kim, Y. Jernite, D. Sontag, and A. M. Rush, "Character-aware neural language models," *arXiv preprint arXiv:1508.06615*, 2015.

[10] H. Rubenstein and J. B. Goodenough, "Contextual correlates of synonymy," *Communications of the ACM*, vol. 8, no. 10, pp. 627–633, 1965.

[11] G. Halawi, G. Dror, E. Gabrilovich, and Y. Koren, "Large-scale learning of word relatedness with constraints," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2012, pp. 1406–1414.

[12] T. Luong, R. Socher, and C. D. Manning, "Better word representations with recursive neural networks for morphology." in *Proceedings of the SIGNLL Conference on Computational Natural Language Learning*, 2013, pp. 104–113.

[13] L. Finkelstein, E. Gabrilovich, Y. Matias, E. Rivlin, Z. Solan, G. Wolfman, and E. Ruppin, "Placing search in context: The concept revisited," in *Proceedings of the International Conference on World Wide Web*. ACM, 2001, pp. 406–414.

[14] E. Agirre, E. Alfonseca, K. Hall, J. Kravalova, M. Paşca, and A. Soroa, "A study on similarity and relatedness using distributional and wordnet-based approaches," in *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2009, pp. 19–27.

[15] E. Bruni, N.-K. Tran, and M. Baroni, "Multimodal distributional semantics," *Journal of Artificial Intelligence Research*, vol. 49, no. 1-47, 2014.

[16] M. Faruqui and C. Dyer, "Non-distributional word vector representations," *arXiv preprint arXiv:1506.05230*, 2015.

[17] R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, vol. 1631, 2013, p. 1642.

[18] M. Faruqui and C. Dyer, "Community evaluation and exchange of word vectors at wordvectors.org," in *Proceedings of the Meeting of the Association for Computational Linguistics: System Demonstrations*, 2014, pp. 19–24.