Sentence transition matrix: An efficient approach that preserves sentence semantics

Myeongjun Jang [1]   **Pilsung Kang** [1]

Abstract

Sentence embedding is a significant research topic in the field of natural language processing (NLP). Generating sentence embedding vectors reflecting the intrinsic meaning of a sentence is a key factor to achieve an enhanced performance in various NLP tasks such as sentence classification and document summarization. Therefore, various sentence embedding models based on supervised and unsupervised learning have been proposed after the advent of researches regarding the distributed representation of words. They were evaluated through semantic textual similarity (STS) tasks, which measure the degree of semantic preservation of a sentence and neural network-based supervised embedding models generally yielded state-of-the-art performance. However, these models have a limitation in that they have multiple parameters to update, thereby requiring a tremendous amount of labeled training data. In this study, we propose an efficient approach that learns a transition matrix that refines a sentence embedding vector to reflect the latent semantic meaning of a sentence. The proposed method has two practical advantages; (1) it can be applied to any sentence embedding method, and (2) it can achieve robust performance in STS tasks irrespective of the number of training examples.

Keywords: *Sentence embedding, Sentence semantics, Transition matrix, Paraphrase, Natural language processing*

[1]School of Industrial Management Engineering, Korea University, Seoul, South Korea. Correspondence to: Myeongjun Jang <xkxpa@korea.ac.kr>, Pilsung Kang <pilsung_kang@korea.ac.kr>.

1. Introduction

Sentence embedding, the task of transforming a sequence of words in a sentence into a fixed-dimensional vector form reflecting the intrinsic meaning, plays an important role in the field of natural language processing (NLP). It can be considered as an essential preprocessing step that transforms unstructured textual data into structured and continuous-valued vectors that can be used as input to machine learning algorithms to conduct various NLP tasks such as machine translation (Sutskever et al., 2014; Wu et al., 2016), document classification (Kim, 2014; Conneau et al., 2017c), and sentence matching (Hu et al., 2014; Wan et al., 2016). Because performances of many NLP tasks heavily rely on the word/sentence/document embedding methods, a large number of studies have been conducted since the advent of Doc2vec (Le & Mikolov, 2014), and progressive performance improvements have been reported whenever a new embedding method was proposed.

A good sentence embedding model should yield a vector value that closely captures the intrinsic semantic meaning of the sentence. Therefore, diverse experiments predicting the similarity score of two embedded sentences have been designed to evaluate how well the embedding model satisfies such requirements. These experiments were conducted in many researches, such as SIF (Arora et al., 2017), Sent2vec (Pagliardini et al., 2017), InferSent (Conneau et al., 2017a), and Fast-Sent (Hill et al., 2016a), and models trained through supervised learning generally showed far better performances than those based on unsupervised learning. However, supervised models have a limitation that they require a sufficient number of labeled training data to learn the model. For instance, Stanford Natural Language Inference (SNLI) dataset, which contains a collection of 570k manually labeled human-written English sentence pairs (Bowman et al., 2015), was used to train the InferSent model. Because it is an English dataset, one must construct a new dataset to train the embedding model for other languages, e.g., Russian, Korean, and Japanese, which is a heavy time-

and resource-consuming work. Therefore, to increase the general usability of an embedding model, it is necessary to reduce reliance on labeled training data while closely preserving the intrinsic meaning of the sentence.

In this study, motivated by the studies that obtained successful results in cross-lingual embedding (Mikolov et al., 2013a) and word-level translation (Smith et al., 2017) by training a simple transform matrix, we propose an approach to learn the transition matrix that refines the sentence vector generated by other sentence embedding models. The contributions of this research are as follows:

- We propose an efficient method that successfully reflects sentence semantics through training the transition matrix.

- The proposed method is independent of the size of training data and can be applied to any kind of sentence embedding model.

To reflect the intrinsic meaning of a sentence completely, we define a fundamental property that a good sentence embedding model should satisfy: the property of semantic coherence (Jang & Kang, 2018), which implies that paraphrase sentences should be closely located to each other in the sentence embedding space. Next, we derive an objective function to learn the transition matrix to closely satisfy the formulated property and train the matrix using the MSCOCO caption dataset. Experimental results show that the proposed approach is practically advantageous in that it outperforms existing sentence embedding models in various semantic textual similarity (STS) tasks.

The rest of this paper is organized as follows. In Section 2, we briefly review the past researches on sentence embedding. In Section 3, we introduce the objective function for training the transition matrix and the training method. In Section 4, the effectiveness of the proposed method is demonstrated with the experimental results on STS tasks, along with their performance comparison with benchmark models. Finally, in Section 5, we conclude our present study, which leads us to some future research directions.

2. Related work

Recent researches on sentence embedding models are diverse for models based on unsupervised learning to those based on supervised learning. Unsupervised embedding models can be divided into two categories based on whether sentence sequence information (not word sequence information within a sentence) is required during training.

Doc2vec (Le & Mikolov, 2014) is a representative model that does not need sentence sequence information. Paragraph vectors-distributed bag of words (PV-DBOW) and paragraph vector-distributed memory (PV-DM), two distinct learning methods of Doc2vec, train sentence vectors based on the same objective: maximizing the probability to predict words constituting the sentence. The probability is defined as the dot product between a sentence vector and a word vector. PV-DM considers sequential information of words by employing a moving window. In this method, a sentence vector is learned to predict a word appearing after the moving window using the words within the window and the sentence vector. However, in the PV-DBOW method, words included in the window are arbitrarily selected. Therefore, this is incapable of reflecting sequential information of words in a sentence.

Hill et al. (2016b) proposed the sequential denoising autoencoder (SDAE), which slightly corrupts the input sentence by adding random noise. The noise is added in two different ways. First, each word is randomly dropped in a sequence according to the probability p_0. Next, for the bigrams that are not overlapped, the order of two adjacent words is permuted according to the probability p_x. Then, the embedding model, which is a recurrent neural network (RNN) with a long short-term memory (Hochreiter & Schmidhuber, 1997) (LSTM) cell, updates its parameter to generate the original sentence from the corrupted one. If $p_0 = p_x = 0$, the model becomes a sequential autoencoder (SAE), which does not add noise to the input. Hill et al. (2016b) also proposed a variant of the SDAE model that employs fixed pre-trained word vectors. This model is notated as "S(D)AE + embs" in our study.

Arora et al. (2017) proposed a simple embedding model named SIF, which computes a sentence vector as a weighted average of fixed pre-trained word embedding vectors. Despite its simplicity, SIF accomplished improved performance in STS tasks and outperformed many complex models based on RNNs if word weights are properly adjusted. Sent2vec (Pagliardini et al., 2017) has a similar characteristic with SIF, which computes a sentence vector as a weighted average of word embedding vectors. However, Sent2vec trains not only the embedding vector of words that are unigram, but also that of n-gram tokens. It is different from SIF in that it employs n-gram embedding vectors to generate sentence vectors.

Contrary to previously demonstrated sentence embed-

ding models that exploit information from corpora to learn sentence vectors, C-PHRASE (Kruszewski et al., 2015) requires external information. C-PHRASE uses information from the syntactic parse tree of each sentence. This additional knowledge is included in the training objective of C-BOW (Mikolov et al., 2013b).

SkipThought (Kiros et al., 2015) is a sentence embedding model in which sentence sequences are mandatory during the training. It has a sequence-to-sequence structure and expands the training objective of Skipgram (Mikolov et al., 2013b) for learning word embedding vectors to a sentence level. Similar to the Skipgram model, which updates word embedding vectors by predicting the surrounding words when the center word is given, the training objective of SkipThought is to generate the preceding and following sentences when a sentence is given.

FastSent (Hill et al., 2016a), similar to SkipThought, is a sentence embedding model aimed at predicting the surrounding sentences of a given sentence. FastSent learns the source word embedding $\mathbf{u}_w$ and target word embedding $\mathbf{v}_w$. When three consecutive sentences S_{i-1}, S_i, S_{i+1} are given, $\mathbf{s}_i$, the representation vector of S_i, is calculated as the sum of the source word embedding vectors:

$$\mathbf{s}_i = \sum_{w \in \mathbf{S}_i} \mathbf{u}_w. \tag{1}$$

Then, the cost function is simply defined as follows:

$$\sum_{w \in S_{i-1} \cup S_{i+1}} softmax(\mathbf{s}_i \cdot \mathbf{v}_w). \tag{2}$$

In addition, Hill et al. (2016a) also proposed a variant model (FastSent+AE) that predicts not only the adjacent sentences but also the center sentence S_i. FastSent with such a simple structure, takes much less training time than SkipThought.

Siamese C-BOW (Kenter et al., 2016) shares a common concept with SIF and Sent2vec: defining a sentence vector as the average of word embedding vectors. In addition, it is similar to SkipThought and FastSent in that it is also trained to predict surrounding sentences when the center sentence is given. However, Siamese C-BOW employs a Siamese neural network (Koch, 2015) structure, which is significantly different from the sentence embedding models described above.

InferSent (Conneau et al., 2017a) is a sentence embedding model trained through supervised tasks. Inspired by previous researches in computer vision, where a large number of models are pre-trained through a classification task based on the ImageNet (Deng et al.,

2009) dataset, Conneau et al. (2017a) performed a research to determine the effectiveness of supervised tasks in the learning of a sentence embedding model in the field of NLP. Through experiments, Conneau et al. (2017a) concluded that a sentence embedding model having a bidirectional LSTM structure, trained on the SNLI dataset, yielded state-of-the-art performance in various NLP tasks.

All sentence embedding models using sentence sequence information, described above, require a specific dataset to train the model. For instance, SkipThought and FastSent used the Book Corpus dataset (Zhu et al., 2015) and InferSent used the SNLI dataset to train the embedding models. In the case of sentence embedding models independent of sentence sequence information, a large document dataset, such as Wikipedia sentences, was also used. Furthermore, these models have a limitation in completely reflecting the traits of paraphrase sentences conveying the same meaning but having a different word usage, because these models generate sentence vectors based on word embedding vectors. In this study, we propose an efficient approach that successfully preserves sentence semantics by employing only a small amount of paraphrase sentences.

3. An efficient transition matrix for generating sentence embedding vectors

In this section, we first define a fundamental property that a good sentence embedding model should satisfy. Next, we describe the method to learn the transition matrix from the derived property. Finally, we demonstrate the processing method for training data when learning the transition matrix.

3.1. Property of semantic coherence

Semantic coherence refers to a fundamental property that a good sentence embedding model should satisfy: if two sentences have similar meaning, they should be located close to each other in the embedding space. The semantic coherence of a sentence embedding model can be evaluated as follows:

Definition: The degree of semantic coherence of a sentence embedding model is proportional to the similarity between the embedding vectors of paraphrase sentences generated by the sentence embedding model.

The above definition can be expressed mathematically. Assume that the set **I** is a vector set of input sentences

and the set $\mathbf{P}$ is a vector set of paraphrase sentences corresponding to the input sentences:

$$\mathbf{I} = (\mathbf{i}_1, \mathbf{i}_2, ..., \mathbf{i}_n), \quad \mathbf{P} = (\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_n), \qquad (3)$$

where $(\mathbf{i}_k, \mathbf{p}_k)$ is a pair of paraphrase sentence vectors. Then, the similarity matrix of elements belonging to each set can be computed as follows:

$$\hat{\mathbf{S}} = \mathbf{I}\mathbf{P}^T. \qquad (4)$$

The normalized similarity matrix $\mathbf{N}$ can be obtained by the element-wise multiplication of $\mathbf{N}$ with $\hat{\mathbf{S}}$ as follows:

$$\mathbf{N}_{ij} = \frac{1}{||\mathbf{i}_i||||\mathbf{p}_j||}, \quad \mathbf{S} = \mathbf{N} \times \hat{\mathbf{S}}. \qquad (5)$$

The tupple $(\mathbf{i}_i, \mathbf{p}_j)$ is a pair of sentence vectors sharing a similar intrinsic meaning when $i = j$. However, when $i \neq j$, the sentence vectors have different meanings. Therefore, the similarity of two sentence vectors $\mathbf{i}_i$ and $\mathbf{p}_j$ should be close to 1 when $i = j$ and should be minimized when $i \neq j$. Hence, the similarity matrix $\mathbf{S}$ of two vector sets $\mathbf{I}$ and $\mathbf{P}$ should satisfy the following two conditions if the sentence embedding model fulfills the property of semantic coherence.

Condition 1: $\mathbf{d} = diag(\mathbf{S}) \approx \vec{1}$,
Condition 2: $|\mathbf{S} - diag(\mathbf{d})|$ should be minimized.

3.2. Sentence transition matrix

In this study, we attempted to devise an approach satisfying the semantic coherence by learning a minimal number of weights. To achieve this, we trained the transition matrix, which elaborates the pre-generated sentence vectors to meet the conditions described above. Assume that $\mathbf{I}^M$ and $\mathbf{P}^M$ refer to the sentence vector sets generated by the sentence embedding model M of the input sentence and the corresponding paraphrase sentences, respectively.

$$\mathbf{I}^M = (\mathbf{i}_1^M, \mathbf{i}_2^M, ..., \mathbf{i}_n^M), \quad \mathbf{P}^M = (\mathbf{p}_1^M, \mathbf{p}_2^M, ..., \mathbf{p}_n^M). \quad (6)$$

The main purpose of this study is to train the transition matrix $\mathbf{W}$ using $\mathbf{I}^M$ and $\mathbf{P}^M$. First, each set of sentence vectors is multiplied by the transition matrix as follows:

$$\hat{\mathbf{I}}^M = \mathbf{W}\mathbf{I}^M, \quad \hat{\mathbf{P}}^M = \mathbf{W}\mathbf{P}^M. \qquad (7)$$

Then, the similarity matrix of the sentence vectors to which the transition matrix is applied is computed as follows:

$$\hat{\mathbf{S}}^M = \hat{\mathbf{I}}^M(\hat{\mathbf{P}}^M)^T, \quad \mathbf{S}^M = \mathbf{N} \times \hat{\mathbf{S}}^M,$$
$$\text{where } \mathbf{N}_{ij} = \frac{1}{||\mathbf{i}_i^M||||\mathbf{p}_j^M||}. \qquad (8)$$

Next, from the formulated conditions of the semantic coherence property, the diagonal and non-diagonal losses are defined as follows:

$$\text{diagonal\_loss} = average(|diag(\mathbf{S}^M - \vec{1})|),$$
$$\text{non-diagonal\_loss} = average(|\mathbf{S}^M - diag(diag(\mathbf{S}^M))|). \qquad (9)$$

Finally, the final training loss is defined as follows:

$$\begin{aligned} \text{loss} = &\lambda \times \text{non-diagonal\_loss} \\ &+ (1 - \lambda) \times \text{diagonal\_loss}, \end{aligned} \qquad (10)$$

where λ is the user-specific hyperparameter that controls the trade-off between the two losses. The sentence vector for sentence x after training the transition matrix is computed as follows:

$$\mathbf{sv}_x = \mathbf{W} \cdot \mathrm{M}(x), \qquad (11)$$

where $\mathrm{M}(x)$ is the vector value of the sentence x generated by model M.

Model M can be any kind of sentence embedding model. However, we defined the average of pre-trained word vectors as a sentence vector to show that the performance improvement could be obtained by a simple method. We used two kinds of pre-trained word vectors: GloVe vectors (Pennington et al., 2014) and Google Word2vec[1].

3.3. Training transition matrix

Composing training dataset: To train the transition matrix, we employed the MSCOCO 2017 training dataset (Lin et al., 2014), which is widely used as a paraphrase dataset in many researches (Prakash et al., 2016; Gupta et al., 2017). This dataset contains a minimum of five captions per image for 118,284 images. The total number of unique captions is 591,753. To train the transition matrix proposed in this study, $\mathbf{i}_i$ and $\mathbf{p}_j$ should be mutually exclusive except for the case when $i = j$. In other words, in a mini-batch, only one paraphrase sentence should be included for one corresponding sentence when training the transition matrix. Hence, we constructed the training dataset as explained below.

First, note that it is possible to create ten unique sentence pairs for each image because at least five captions are provided for each image. Therefore, for all images, if we include only one sentence pair per image in one sub-training set, then the sub-training set does not contain sentence pairs with overlapping meaning and totally ten sub-training sets can be constructed.

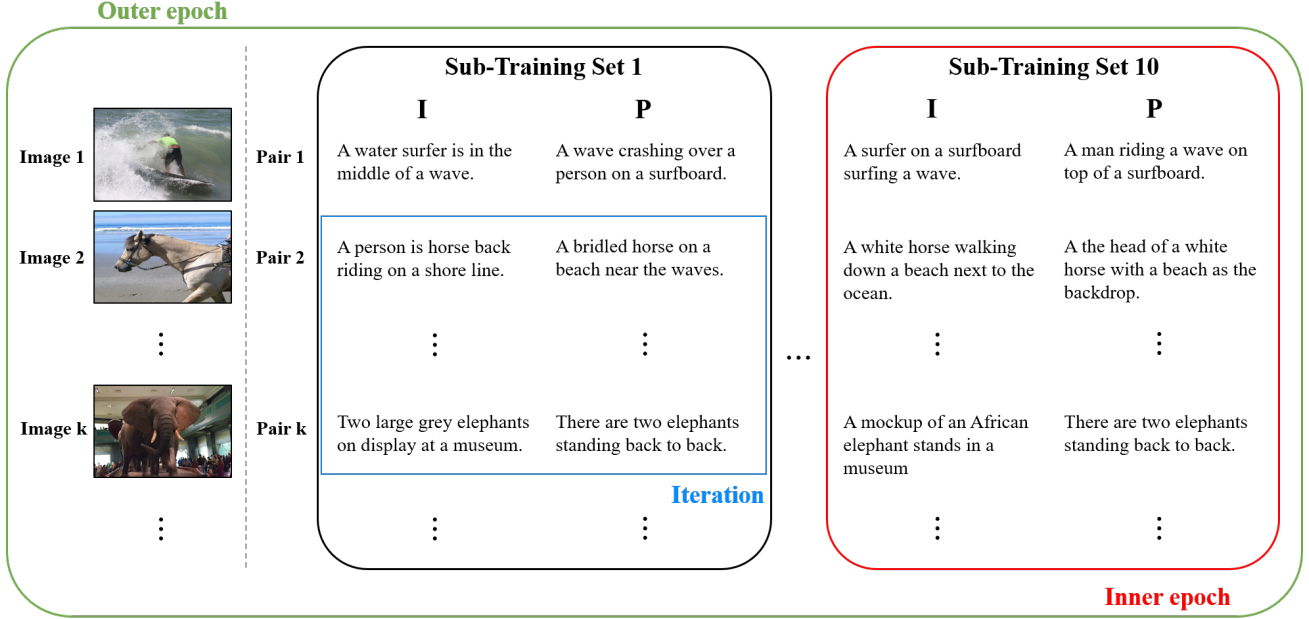[1]https://code.google.com/archive/p/word2vec/

Figure 1. Description of training data composition

In our experiment, we denoted the training of one sub-training set as *inner epoch* and of all sub-training sets as *outer epoch*. The description of training data composition is provided in Figure 1.

Training option: Both GloVe vectors and Google Word2vec have word vectors of 300 dimensions. Therefore, the transition matrix has a size of 300×300. We employed Xavier initialization (Glorot & Bengio, 2010) and weight updates were performed with a mini-batch size of 512. We trained the transition matrix for five outer epochs using the RMSprop optimizer[2].

4. Experiments

4.1. Textual similarity task

Data set: To evaluate how well the proposed approach reflects the intrinsic meaning of sentences, we conducted STS tasks (2012–2016) (Agirre et al., 2012; 2013; 2014; 2015; 2016) and SemEval 2014 semantic relatedness task (SICK) (Marelli et al., 2014). The objective of these tasks is to predict the similarity score of two given sentences. Their performance is evaluated by comparing the predicted score with the ground truth, which is the similarity score determined by human judgments. The evaluation metric is Pearson's r (Pearson, 1895) and Spearman's ρ (Spearman, 1904). Similar to previous researches, we defined the

_____
[2]http://www.cs.toronto.edu/ tij-men/csc321/slides/lecture_slides_lec6.pdf

predicted similarity score as the cosine similarity between two sentence vectors.

Experiment design: We conducted the experiments in three different levels based on the number of training data to investigate the relationship between the performance and size of training data. The first level is the case of using all available data. The second and third levels are the cases where only 50 % and 10 %, respectively, of the total training data are used.

In the proposed approach, the hyperparameter λ is a key value affecting the model performance. In this experiment, we empirically determined λ from a sufficient number of candidate values. The λ values for GloVe vectors and Google Word2vec are set to 0.7 and 0.9, respectively.

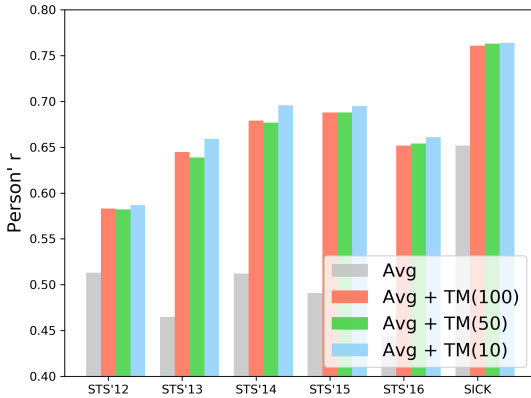### 4.2. Effect of the transition matrix on performance

We first evaluated the effect of the transition matrix by comparing the performances before and after applying the transition matrix to sentence vectors that are defined as the average of pre-trained word vectors. The result is summarized in Table 1 and Figure 2. The notation "TM" denotes that the transition matrix is applied, and the number in the parentheses indicates the percentage of training data used. The evaluation metric recorded in Table 1 is Pearson's r.

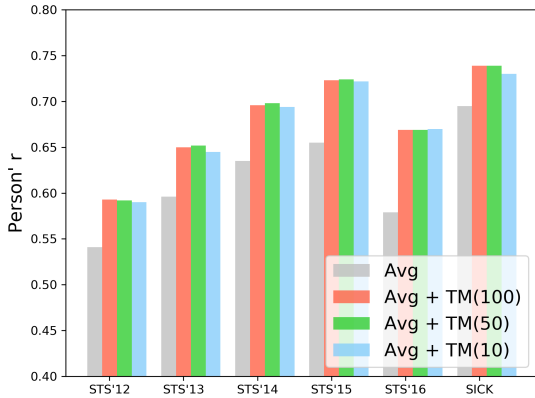Experimental results show that the proposed transi-

Table 1. Result of semantic textual similarity tasks

	Model	STS 12	STS 13	STS 14	STS 15	STS 16	SICK
	Avg	0.513	0.465	0.512	0.491	0.444	0.652
Glove	Avg+TM(100)	0.583	0.645	0.679	0.688	0.652	0.761
	Avg+TM(50)	0.582	0.639	0.677	0.688	0.654	0.763
	Avg+TM(10)	**0.587**	**0.659**	**0.696**	**0.695**	**0.661**	**0.764**
	Avg	0.541	0.596	0.635	0.655	0.579	0.695
Google	Avg+TM(100)	**0.593**	0.650	0.696	0.723	0.669	**0.739**
Word2vec	Avg+TM(50)	0.592	**0.652**	**0.698**	**0.724**	0.669	**0.739**
	Avg+TM(10)	0.590	0.645	0.694	0.722	**0.670**	0.730

(a) Glove

(b) Google Word2vec

Figure 2. Effect of transition matrix

tion matrix remarkably improves the performance in both the pre-trained word vectors. The training time was about 130, 60, and 13 s when 100, 50, and 10 %, respectively, of the training data was used to train **W**.In the case of Google Word2vec, the performance improvement is $5 \sim 16$ %, while the transition matrix with Glove vector shows a very high performance improvement of $13 \sim 49$ %. Both the transition matrices do not show significant performance differences according to the amount of training data. Therefore, it can be concluded that the performance of a transition matrix is independent of the amount of paraphrase data used in training.

4.3. Comparison with other sentence embedding models

After confirming the effectiveness of the transition matrix, we compared its performance with previously proposed sentence embedding models. The benchmark models are as follows:

- **Embedding models using sentence sequence information**: S(D)AE, PV-DBOW, PV-DM, Sent2vec, C-PHRASE, and SIF.

- **Embedding models not using sentence sequence information**: SkipThought, FastSent, and Siamese C-BOW.

- **Supervised task-based sentence embedding model**: InferSent

The tasks used for comparison are the STS 2014 and SICK relatedness tasks, both of which were used in the benchmark models. The benchmark results were obtained by Pagliardini et al. (2017), Arora et al. (2017), and Conneau et al. (2017a). For the models discussed above, the result of Pagliardini et al. (2017) contains the model defining a sentence vector as the simple average of word embeddings trained by the C-BOW (Mikolov et al., 2013b) method. It also includes the result of TF-IDF representation, which is the weighted word count of the 200,000 most common words. We underlined the best performance for the dataset, while

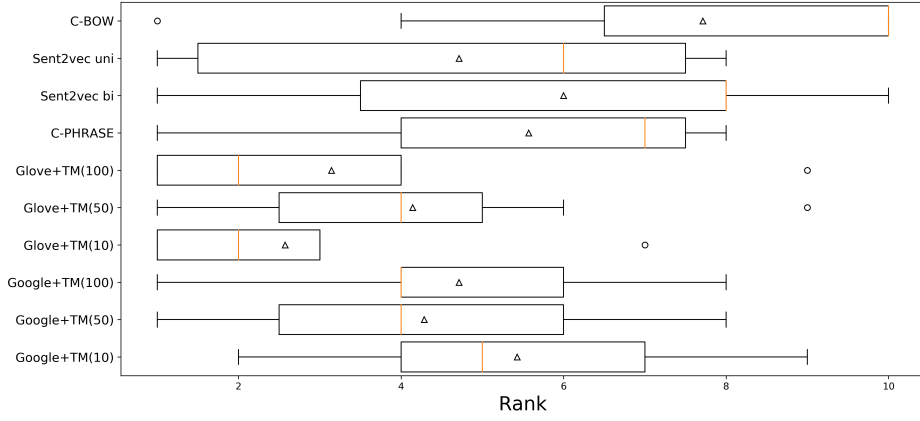*Table 2.* Result of semantic textual similarity tasks

Model	STS 2014							SICK
	News	Forum	WordNet	Twitter	Images	Headlines	Avg	Test+train
SAE	.17/.16	.12/.12	.30/.23	.28/.22	.49/.46	.13/.11	.27/.23	.32/.31
SAE+embs	.52/.54	.22/.23	.60/.55	.60/.60	.64/.64	.41/.41	.53/.52	.47/.49
SDAE	.07/.04	.11/.13	.33/.24	.44/.42	.44/.38	.36/.36	.33/.30	.46/.46
SDAE+embs	.51/.54	.29/.29	.56/.50	.57/.58	.59/.59	.43/.44	.51/.51	.46/.46
PV-DBOW	.31/.34	.32/.32	.53/.50	.43/.46	.46/.44	.39/.41	.42/.43	.42/.46
PV-DM	.42/.46	.33/.34	.51/.48	.54/.57	.32/.30	.46/.47	.44/.45	.44/.40
C-BOW	.57/.61	.43/.44	.72/.69	**.71/.75**	.71/.73	.55/.59	.64/.66	.60/.69
Uni TF-IDF	.48/.48	.40/.38	.60/.59	.63/.65	.72/.74	.49/.49	.58/.58	.52/.58
Sent2vec uni	.62/.67	**.49/.49**	.75/.72	**.70/.75**	**.78**/.82	**.61/.63**	**.68/.70**	.61/.70
Sent2vec bi	.62/.67	**.49/.49**	.71/.68	**.70/.75**	.75/.79	.59/.62	**.66**/.69	**.62**/.70
C-PHRASE	.69/.71	.43/.41	.76/.73	.60/.65	.75/.79	**.60/.65**	**.66**/.68	.60/.72
SIF (Glove)	-	-	-	-	-	-	- /.69	- /.72
SkipThought	.44/.45	.14/.15	.39/.34	.42/.43	.55/.60	.43/.44	.42/.43	.57/.60
FastSent	.58/.59	.41/.36	.74/.70	.63/.66	.74/.78	.57/.59	.64/.65	.61/.72
FastSent+AE	.56/.59	.41/.40	.69/.64	.70/.74	.63/.65	.58/.60	.62/.65	.60/.65
Siamse C-BOW	.58/.59	.42/.41	.66/.61	.71/.73	.63/.65	**.63/.64**	.63/.63	-
InferSent	-	-	-	-	-	-	**.67/.70**	-
Glove avg	.66/.66	.29/.22	.63/.56	.59/.57	.57/.58	.45/.46	.53/.51	.55/.65
Google w2v avg	.64/.69	.31/.31	.76/.73	.66/.70	.68/.71	.52/.58	.61/.64	.60/.70
Glove avg+TM(100)	**.70/.73**	.43/.44	.74/.73	.57/.64	**.77/.84**	.58/**.63**	.64/.68	.61/**.76**
Glove avg+TM(50)	**.70/.72**	.43/.44	.73/.72	.58/.64	**.77**/.83	.58/**.63**	.64/.68	.61/**.76**
Glove avg+TM(10)	**.71/.73**	**.45/.45**	.78/.77	.61/.67	**.78/.84**	.58/**.63**	**.66/.70**	**.62/.76**
Google w2v avg+TM(100)	.62/.69	.37/.38	**.81/.81**	.66/.73	**.77**/.83	.54/.61	.65/**.70**	**.63**/.74
Google w2v avg+TM(50)	.63/.69	.37/.38	**.81/.81**	.66/.73	**.77/.84**	.54/.61	.65/**.70**	**.63**/.74
Google w2v avg+TM(10)	.63/.69	.36/.38	**.81/.80**	.67/.73	**.77**/.83	.54/.60	.65/.69	**.62**/.73

the top three performances are shown in bold. The order of recording the results is Spearman's ρ / Pearson's r. The results are summarized in Table 2.
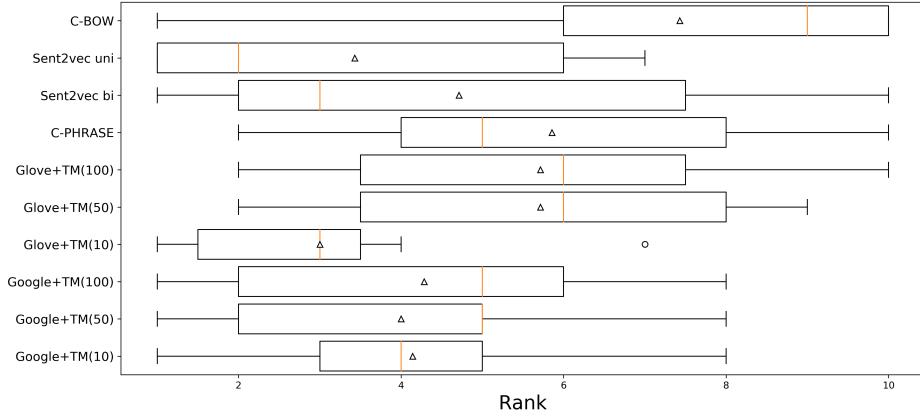
The experimental results show that the proposed approach is ranked among the top three for most of the datasets despite its low computational complexity. We further compared the performance ranks of the proposed models with three selected benchmark models: Sent2vec, C-BOW, and C-PHRASE. The selection criteria are: (1) the model was tested for both STS 14 and SICK experiments, and (2) it performed the best for at least one dataset. For each dataset, we recorded the performance ranks of the models and then compared the distribution of the ranks and its average. Figure 3 shows the box plots of the performance ranks for the selected ten models. The triangles denote the average performance ranks. The results show that the model which used Glove vectors and 10 % of the total paraphrase sentences yielded the lowest average rank in both Pearson's r and Spearman's ρ. In addition, in

the case of Pearson's r, most of our proposed models resulted in lower average ranks than those of Sent2vec, which showed the best performance among the benchmark models.

Comparison with SIF: SIF has characteristics similar to our approach, i.e., it also computes a sentence representation vector from fixed pre-trained word vectors in a simple manner. In Table 2, we were not able to compare the performance with SIF in detail because Arora et al. (2017) did not specify the experimental result for the individual datasets of the STS 2014 task. However, because Arora et al. (2017) performed STS 2012–2015 tasks and the SICK semantic relatedness task, we compared the performance of our approach with SIF in more detail, as shown in Table 3. We recorded the Pearson's r of the best model for each pre-trained word vector. The experimental results reveal that our proposed approach showed a better performance than SIF for all the datasets. We also observed that the average performance improvement

(a) Pearson's r

(b) Spearman's ρ

Figure 3. Box plot of performance rank for each model

Table 3. Detailed comparison with SIF

Model	STS 12	STS 13	STS 14	STS 15	SICK
SIF (Glove + WR)	56.2	56.6	68.5	71.7	72.2
Glove+TM(10)	58.7	**65.9**	69.9	66.1	**76.4**
Goggle w2v+TM(50)	**59.2**	65.2	**69.8**	**72.4**	73.9

ratio was 6 % and the performance was enhanced to a maximum of 16.5 % compared to that of SIF.

5. Conclusion

Sentence embedding, which transforms an unstructured textual data into a structured vector form, is a fundamental and imperative method in the field of NLP. Generating sentence vectors that preserve the semantic meaning of a sentence is the key component to achieve an improved performance in various NLP tasks. Hence, various sentence embedding models have been proposed, which show an enhanced performance in various NLP tasks such as document classification, sentiment analysis, and semantic similarity task.

In this study, we first defined the property of semantic coherence that closely preserves the sentence semantics. Subsequently, we derived the objective function

to train the transition matrix, which refines sentence representation vectors to satisfy the derived property. Finally, the transition matrix is trained by employing the MSCOCO caption dataset, which is widely used as a paraphrase dataset.

The proposed approach was evaluated through various semantic textual similarity tasks. Despite its low computational complexity, our approach showed significant improvements in various STS tasks. In addition, compared to the previously proposed benchmark models, our approach showed the best performance in many datasets and one of the best performances in almost every dataset.

Nonetheless, the proposed method has a limitation that paraphrase data is required to train the transition matrix. Although the experimental results show that the performance of a transition matrix is independent of the amount of training data, it can be a critical issue when analyzing the languages which are difficult to obtain paraphrase data. Therefore, similar to researches on unsupervised machine translation (Artetxe et al., 2017; Lample et al., 2017; 2018) and unsupervised cross-lingual embedding (Wada & Iwata, 2018; Xu et al., 2018; Conneau et al., 2017b), both of which do not require labeled data, a research to generate sentence vectors preserving the latent meaning of sentences without paraphrase data should be developed.

References

Agirre, Eneko, Diab, Mona, Cer, Daniel, and Gonzalez-Agirre, Aitor. Semeval-2012 task 6: A pilot on semantic textual similarity. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, pp. 385–393. Association for Computational Linguistics, 2012.

Agirre, Eneko, Cer, Daniel, Diab, Mona, Gonzalez-Agirre, Aitor, and Guo, Weiwei. * sem 2013 shared task: Semantic textual similarity. In *Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, volume 1, pp. 32–43, 2013.

Agirre, Eneko, Banea, Carmen, Cardie, Claire, Cer, Daniel, Diab, Mona, Gonzalez-Agirre, Aitor, Guo, Weiwei, Mihalcea, Rada, Rigau, German, and Wiebe, Janyce. Semeval-2014 task 10: Multilingual semantic textual similarity. In *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)*, pp. 81–91, 2014.

Agirre, Eneko, Banea, Carmen, Cardie, Claire, Cer, Daniel, Diab, Mona, Gonzalez-Agirre, Aitor, Guo, Weiwei, Lopez-Gazpio, Inigo, Maritxalar, Montse, Mihalcea, Rada, et al. Semeval-2015 task 2: Semantic textual similarity, english, spanish and pilot on interpretability. In *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, pp. 252–263, 2015.

Agirre, Eneko, Banea, Carmen, Cer, Daniel, Diab, Mona, Gonzalez-Agirre, Aitor, Mihalcea, Rada, Rigau, German, and Wiebe, Janyce. Semeval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pp. 497–511, 2016.

Arora, Sanjeev, Liang, Yingyu, and Ma, Tengyu. A simple but tough-to-beat baseline for sentence embeddings. *International conference on Learning Representations*, 2017.

Artetxe, Mikel, Labaka, Gorka, Agirre, Eneko, and Cho, Kyunghyun. Unsupervised neural machine translation. *arXiv preprint arXiv:1710.11041*, 2017.

Bowman, Samuel R, Angeli, Gabor, Potts, Christopher, and Manning, Christopher D. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*, 2015.

Conneau, Alexis, Kiela, Douwe, Schwenk, Holger, Barrault, Loic, and Bordes, Antoine. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364*, 2017a.

Conneau, Alexis, Lample, Guillaume, Ranzato, Marc'Aurelio, Denoyer, Ludovic, and Jégou, Hervé. Word translation without parallel data. *arXiv preprint arXiv:1710.04087*, 2017b.

Conneau, Alexis, Schwenk, Holger, Barrault, Loïc, and Lecun, Yann. Very deep convolutional networks for text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, volume 1, pp. 1107–1116, 2017c.

Deng, Jia, Dong, Wei, Socher, Richard, Li, Li-Jia, Li, Kai, and Fei-Fei, Li. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 248–255. IEEE, 2009.

Glorot, Xavier and Bengio, Yoshua. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 249–256, 2010.

Gupta, Ankush, Agarwal, Arvind, Singh, Prawaan, and Rai, Piyush. A deep generative framework for paraphrase generation. *arXiv preprint arXiv:1709.05074*, 2017.

Hill, Felix, Cho, Kyunghyun, and Korhonen, Anna. Learning distributed representations of sentences from unlabelled data. *arXiv preprint arXiv:1602.03483*, 2016a.

Hill, Felix, Cho, Kyunghyun, and Korhonen, Anna. Learning distributed representations of sentences from unlabelled data. *arXiv preprint arXiv:1602.03483*, 2016b.

Hochreiter, Sepp and Schmidhuber, Jürgen. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Hu, Baotian, Lu, Zhengdong, Li, Hang, and Chen, Qingcai. Convolutional neural network architectures for matching natural language sentences. In *Advances in neural information processing systems*, pp. 2042–2050, 2014.

Jang, Myeongjun and Kang, Pilsung. Paraphrase thought: Sentence embedding module imitating human language recognition. *arXiv preprint arXiv:1808.05505*, 2018.

Kenter, Tom, Borisov, Alexey, and de Rijke, Maarten. Siamese cbow: Optimizing word embeddings for sentence representations. *arXiv preprint arXiv:1606.04640*, 2016.

Kim, Yoon. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

Kiros, Ryan, Zhu, Yukun, Salakhutdinov, Ruslan R, Zemel, Richard, Urtasun, Raquel, Torralba, Antonio, and Fidler, Sanja. Skip-thought vectors. In *Advances in neural information processing systems*, pp. 3294–3302, 2015.

Koch, Gregory. Siamese neural networks for one-shot image recognition. In *ICML Deep Learning workshop*, volume 2, 2015.

Kruszewski, Germán, Lazaridou, Angeliki, Baroni, Marco, et al. Jointly optimizing word representations for lexical and sentential tasks with the c-phrase model. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pp. 971–981, 2015.

Lample, Guillaume, Denoyer, Ludovic, and Ranzato, Marc'Aurelio. Unsupervised machine translation using monolingual corpora only. *arXiv preprint arXiv:1711.00043*, 2017.

Lample, Guillaume, Ott, Myle, Conneau, Alexis, Denoyer, Ludovic, and Ranzato, Marc'Aurelio. Phrase-based & neural unsupervised machine translation. *arXiv preprint arXiv:1804.07755*, 2018.

Le, Quoc and Mikolov, Tomas. Distributed representations of sentences and documents. In *International Conference on Machine Learning*, pp. 1188–1196, 2014.

Lin, Tsung-Yi, Maire, Michael, Belongie, Serge, Hays, James, Perona, Pietro, Ramanan, Deva, Dollár, Piotr, and Zitnick, C Lawrence. Microsoft coco: Common objects in context. In *European conference on computer vision*, pp. 740–755. Springer, 2014.

Marelli, Marco, Bentivogli, Luisa, Baroni, Marco, Bernardi, Raffaella, Menini, Stefano, and Zamparelli, Roberto. Semeval-2014 task 1: Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment. In *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)*, pp. 1–8, 2014.

Mikolov, Tomas, Le, Quoc V, and Sutskever, Ilya. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*, 2013a.

Mikolov, Tomas, Sutskever, Ilya, Chen, Kai, Corrado, Greg S, and Dean, Jeff. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pp. 3111–3119, 2013b.

Pagliardini, Matteo, Gupta, Prakhar, and Jaggi, Martin. Unsupervised learning of sentence embeddings using compositional n-gram features. *arXiv preprint arXiv:1703.02507*, 2017.

Pearson, Karl. Note on regression and inheritance in the case of two parents. *Proceedings of the Royal Society of London*, 58:240–242, 1895.

Pennington, Jeffrey, Socher, Richard, and Manning, Christopher. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

Prakash, Aaditya, Hasan, Sadid A, Lee, Kathy, Datla, Vivek, Qadir, Ashequl, Liu, Joey, and Farri, Oladimeji. Neural paraphrase generation with stacked residual lstm networks. *arXiv preprint arXiv:1610.03098*, 2016.

Smith, Samuel L, Turban, David HP, Hamblin, Steven, and Hammerla, Nils Y. Offline bilingual word vectors, orthogonal transformations and the inverted softmax. *arXiv preprint arXiv:1702.03859*, 2017.

Spearman, Charles. The proof and measurement of association between two things. *The American journal of psychology*, 15(1):72–101, 1904.

Sutskever, Ilya, Vinyals, Oriol, and Le, Quoc V. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pp. 3104–3112, 2014.

Wada, Takashi and Iwata, Tomoharu. Unsupervised cross-lingual word embedding by multilingual neural language models. *arXiv preprint arXiv:1809.02306*, 2018.

Wan, Shengxian, Lan, Yanyan, Guo, Jiafeng, Xu, Jun, Pang, Liang, and Cheng, Xueqi. A deep architecture for semantic matching with multiple positional sentence representations. In *AAAI*, volume 16, pp. 2835–2841, 2016.

Wu, Yonghui, Schuster, Mike, Chen, Zhifeng, Le, Quoc V, Norouzi, Mohammad, Macherey, Wolfgang, Krikun, Maxim, Cao, Yuan, Gao, Qin, Macherey, Klaus, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

Xu, Ruochen, Yang, Yiming, Otani, Naoki, and Wu, Yuexin. Unsupervised cross-lingual transfer of word embedding spaces. *arXiv preprint arXiv:1809.03633*, 2018.

Zhu, Yukun, Kiros, Ryan, Zemel, Rich, Salakhutdinov, Ruslan, Urtasun, Raquel, Torralba, Antonio, and Fidler, Sanja. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pp. 19–27, 2015.