Information Processing and Management 56 (2019) 102090

Contents lists available at ScienceDirect

Information Processing and Management

journal homepage: www.elsevier.com/locate/infoproman

Sentence modeling via multiple word embeddings and multi-level comparison for semantic textual similarity

Nguyen Huy Tien[a], Nguyen Minh Le[*,a], Yamasaki Tomohiro[b], Izuha Tatsuya[b]

[a] Japan Advanced Institute of Science and Technology (JAIST) Japan
[b] Toshiba Research & Development Center, Japan

ARTICLE INFO

Keywords:
Multiple word embeddings
Sentence embedding
Semantic
Similarity
Multi-level comparison

ABSTRACT

Recently, using a pretrained word embedding to represent words achieves success in many natural language processing tasks. According to objective functions, different word embedding models capture different aspects of linguistic properties. However, the Semantic Textual Similarity task, which evaluates similarity/relation between two sentences, requires to take into account of these linguistic aspects. Therefore, this research aims to encode various characteristics from multiple sets of word embeddings into one embedding and then learn similarity/relation between sentences via this novel embedding. Representing each word by multiple word embeddings, the proposed MaxLSTM-CNN encoder generates a novel sentence embedding. We then learn the similarity/relation between our sentence embeddings via Multi-level comparison. Our method M-MaxLSTM-CNN consistently shows strong performances in several tasks (i.e., measure textual similarity, identify paraphrase, recognize textual entailment). Our model does not use hand-crafted features (e.g., alignment features, Ngram overlaps, dependency features) as well as does not require pre-trained word embeddings to have the same dimension.

1. Introduction

Measuring the semantic similarity/relation of two pieces of short text plays a fundamental role in a variety of language processing tasks (i.e., plagiarism detection, question answering, and machine translation). Semantic textual similarity (STS) task is challenging because of the diversity of linguistic expression. For example, two sentences with different lexicons could have a similar meaning. Moreover, the task requires to measure similarity at several levels (e.g., word level, phrase level, sentence level). These challenges give difficulties to conventional approaches using hand-crafted features.

Recently, the emergence of word embedding techniques, which encode the semantic properties of a word into a low dimension vector, leads to the successes of many learning models in natural language processing (NLP). For example, Kalchbrenner, Grefenstette, and Blunsom (2014) randomly initialize word vectors, then tunes them during the training phase of a sentence classification task. By contrast, Huy Tien and Minh Le (2017) initialize word vectors via the pre-train word2vec model trained on Google News (Mikolov, Sutskever, Chen, Corrado, & Dean, 2013). Wieting, Bansal, Gimpel, Livescu, and Roth (2015) train a word embedding model on the paraphrase dataset PPDB, then apply the word representation for word and bi-gram similarity tasks.

Several pre-trained word embeddings are available, which are trained on various corpora under different models. Levy and Goldberg (2014) observed that different word embedding models capture different aspects of linguistic properties: a Bag-of-Words

* Corresponding author.
E-mail addresses: ntienhuy@jaist.ac.jp (N.H. Tien), nguyenml@jaist.ac.jp (N.M. Le), tomohiro2.yamasaki@toshiba.co.jp (Y. Tomohiro), tatsuya.izuha@toshiba.co.jp (I. Tatsuya).

https://doi.org/10.1016/j.ipm.2019.102090
Received 14 February 2019; Received in revised form 17 June 2019; Accepted 21 July 2019
Available online 09 August 2019
0306-4573/

contexts based model tends to reflect the domain aspect (e.g., scientist and research) while a paraphrase-relationship based model captures semantic similarities of words (e.g., boy and kid). From experiments, we also observed that the performance of a word embedding model is usually inconsistent over different datasets. This inspired us to develop a model taking advantages of various pre-trained word embeddings for measuring textual similarity/relation.

2. Research objective and contribution

Our research objective is to learn a novel word embedding capturing various linguistic properties from multiple sets of pretrained word embeddings and then measure similarity/relation between two sentences via this embedding. In this paper, we propose a convolutional neural network (CNN) to learn a multi-aspect word embedding from various pre-trained word embeddings. We then apply the max-pooling scheme and Long Short Term Memory (LSTM) on this embedding to form a sentence representation. In STS tasks, Shao (2017) shows the efficiency of the max-pooling scheme in modeling sentences from word embedding representations refined via CNN. However, the max-pooling scheme lacks the property of word order (e.g., *sentence*("Bob likes Marry") = *sentence* ("Marry likes Bob")). To address this weakness, we use LSTM as an additional scheme for modeling sentences with word order characteristics. For measuring the similarity/relation between two sentence representations, we propose Multi-level comparison which consists of word-word level, sentence-sentence level, and word-sentence level. Through these levels, our model comprehensively evaluates the similarity/relation between two sentences.

We evaluate our M-MaxLSTM-CNN model on three tasks: STS, textual entailment recognition, paraphrase identification. The advantages of M-MaxLSTM-CNN are: i) Simple but efficient for combining various pre-trained word embeddings with different dimensions; ii) using multi-level comparison shows better performances compared to using only sentence-sentence comparison; iii) does not require hand-crafted features (e.g., alignment features, Ngram overlaps, syntactic features, dependency features) compared to the state-of-the-art ECNU Tian, Zhou, Lan, and Wu (2017) on STS Benchmark dataset.

Our main contributions are as follows:

- Propose the MaxLSTM-CNN encoder for efficiently encoding sentence embeddings from multiple word embeddings.
- Propose the Multi-level comparison (M-MaxLSTM-CNN) to learn the similarity/relation between two sentences. The model achieves strong performances over various tasks.

The remainder of this paper is organized as follows: Section 3 reviews the previous research, Section 4 introduces the architecture of our model, Section 5 describes the three tasks and datasets, Section 6 describes the experimental setting, Section 7 reports and discusses the results of the experiments, and Section 8 concludes our work.

3. Related work

Most prior research on modeling textual similarity relied on feature engineering. Wan, Dras, Dale, and Paris (2006) extract *n*-gram overlap features and dependency-based features, while Madnani, Tetreault, and Chodorow (2012) employ features based on machine translation metrics. Mihalcea, Corley, Strapparava et al. (2006) propose a method using corpus-based and knowledge-based measures of similarity. Das and Smith (2009) design a model which incorporates both syntax and lexical semantics using dependency grammars. Ji and Eisenstein (2013) combine the fine-grained n-gram overlap features with the latent representation from matrix factorization. Xu, Ritter, Callison-Burch, Dolan, and Ji (2014) develop a latent variable model which jointly learns paraphrase relations between word and sentence pairs. Using Dependency trees, Sultan, Bethard, and Sumner (2014) propose a robust monolingual aligner and successfully applied it for STS tasks. Ferreira, Lins, Simske, Freitas, and Riss (2016) present a novel sentence representation at three layers: lexical, syntactical and semantic. Through the proposed statistical-semantic similarity measurement, the approach achieves strong performances in semantic textual similarity, redundancy elimination in multi-document summarization. According to AL-Smadi, Jaradat, AL-Ayyoub, and Jararweh (2017), these features (i.e., lexical, syntactic, and semantic) also achieve a competitive performance in Arabic news tweets. Ferreira et al. (2018) also employ the three-layer representation with different machine learning methods for identifying paraphrase. Although not achieving the state-of-the-art results in general terms, the model handles the problems of meaning and word order. Jiang, Bai, Zhang, and Hu (2017) and Qu, Fang, Bai, and Jiang (2018) propose some semantic computation approaches using features based on the structure of Wikipedia. These models are efficient in determining semantic similarity between concepts and have a better human correlation than previous methods such as Word2Vec and NASARI (Camacho-Collados, Pilehvar, & Navigli, 2016).

The recent emergence of deep learning models has provided an efficient way to learn continuous vectors representing words/sentences. By using a neural network in the context of a word prediction task, Bengio, Ducharme, Vincent, and Jauvin (2003) and (Mikolov, Chen, Corrado, & Dean, 2013) generate word embedding vectors carrying semantic meanings. The embedding vectors of words which share similar meanings are close to each other. To capture the morphology of words, Bojanowski, Grave, Joulin, and Mikolov (2017) enrich the word embedding with character n-grams information. Closest to this approach, Wieting, Bansal, Gimpel, and Livescu (2016a) also propose to represent a word or sentence using a character n-gram count vector. However, the objective function for learning these embeddings is based on paraphrase pairs.

For modeling sentences, composition approach attracted many studies. Yessenalina and Cardie (2011) model each word as a matrix and used iterated matrix multiplication to present a phrase. Tai, Socher, and Manning (2015) design a Dependency Tree-Structured LSTM for modeling sentences. This model outperforms the linear chain LSTM in STS tasks. Convolutional neural network

(CNN) has recently been applied efficiently for semantic composition (Kalchbrenner et al., 2014; Kim, 2014; Shao, 2017). This technique uses convolutional filters to capture local dependencies in term of context windows and applies a pooling layer to extract global features. He, Gimpel, and Lin (2015) use CNN to extract features at multiple level of granularity. The authors then compare their sentence representations via multiple similarity metrics at several granularities. Gan et al. (2017) propose a hierarchical CNN-LSTM architecture for modeling sentences. In this approach, CNN is used as an encoder to encode a sentence into a continuous representation, and LSTM is used as a decoder. Conneau, Kiela, Schwenk, Barrault, and Bordes (2017) train a sentence encoder on a textual entailment recognition database using a BiLSTM-Maxpooling network. This encoder achieves competitive results on a wide range of transfer tasks. To enhance the conventional word embedding representation for capturing contextual information, Huy and Nguyen (2018) proposed a N-gram word embedding via convolutional filters. This approach achieves a robust performance in multi-lingual sentiment analysis. As trained under one pre-trained embedding, these above approaches depend on the objective function of that embedding.

At SemEval-2017 STS task, hybrid approaches obtain strong performances. Wu, Huang, Jian, Guo, and Su (2017) train a linear regression model with WordNet, alignment features and the word embedding *word2vec*.[1] Tian et al. (2017) develop an ensemble model with multiple boosting techniques (i.e., Random Forest, Gradient Boosting, and XGBoost). This model incorporates traditional features (i.e., n-gram overlaps, syntactic features, alignment features, bag-of-words) and sentence modeling methods (i.e., Averaging Word Vectors, Projecting Averaging Word Vectors, LSTM).

Multichannel Variable-Size Convolution (MVCNN) (Yin & Schütze, 2015) and Multi-Group Norm Constraint CNN (MGNC-CNN) (Zhang, Roller, & Wallace, 2016) are close to our approach. In MVCNN, the authors use variable-size convolution filters on various pre-trained word embeddings for extracting features. However, MVCNN requires word embeddings to have the same size. In MGNC-CNN, the authors apply independently CNN on each pre-trained word embedding for extracting features and then concatenate these features for sentence classification. By contrast, our M-MaxLSTM-CNN model jointly applies CNN on all pre-trained word embeddings to learn a multi-aspect word embedding. From this word representation, we encode sentences via the max-pooling and LSTM. To learn the similarity/relation between two sentences, we employ Multi-level comparison.

4. Model description

Our model (shown in Fig. 1) consists of three main components: i) Learning a multi-aspect word embedding (Section 3.1); ii) modeling sentences from this embedding (Section 3.2); iii) measuring the similarity/relation between two sentences via Multi-level comparison (Section 3.3).

4.1. Multi-aspect word embedding

Given a word w, we transfer it into a word vector e_w^{concat} via K pre-trained word embeddings as follows:

$$e_w^{concat} = e_w^1 \oplus e_w^2 \oplus \dots \oplus e_w^K \tag{1}$$

where $\oplus$ is concatenation operator, e_w^i is the word embedding vector of w in the ith pre-trained embedding.

To learn a multi-aspect word embedding e_w^{multi} from the representation e_w^{concat}, we design H convolutional filters. Each filter r_i is denoted as a weight vector with the same dimension as e_w^{concat} and a bias value b_{r_i}. The e_w^{multi} is obtained by applying these filters on the e_w^{concat} as follows:

$$e_w^{r_i} = \sigma(e_w^{concat} r_i^T + b_{r_1}) \tag{2}$$

$$e_w^{multi} = [e_w^{r_1}, e_w^{r_2}, \dots, e_w^{r_H}] \tag{3}$$

where σ denotes a logistic sigmoid function.

The next section explains how to model a sentence from its multiple-aspect word embeddings.

4.2. Sentence modeling

Given an input sentence $s = [w_1, w_2, \dots, w_n]$, we obtain a sequence of multiple-aspect word embeddings $s^{multi} = [e_{w1}^{multi}, e_{w2}^{multi}, \dots, e_{w_n}^{multi}]$ using Eqs. (1)–(3). For modeling the sentence from the representation s^{multi}, we use two schemes: max-pooling and LSTM.

Max-pooling scheme: To construct a max-pooling sentence embedding $e_s^{\max}$, the most potential features are extracted from the representation s^{multi} as follows:

$$e_s^{max}[i] = max(e_{w_1}^{multi}[i], e_{w_2}^{multi}[i], \dots, e_{w_n}^{multi}[i]) \tag{4}$$

where $e_{w_k}^{multi}[i]$ is the ith element of $e_{w_k}^{multi}$.

LSTM scheme: From Eq. (4), we find that the max-pooling scheme ignores the property of word order. Therefore, we construct a LSTM sentence embedding e_s^{lstm} to support the sentence embedding $e_s^{\max}$. The representation s^{multi} is transformed to a fix-length vector

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

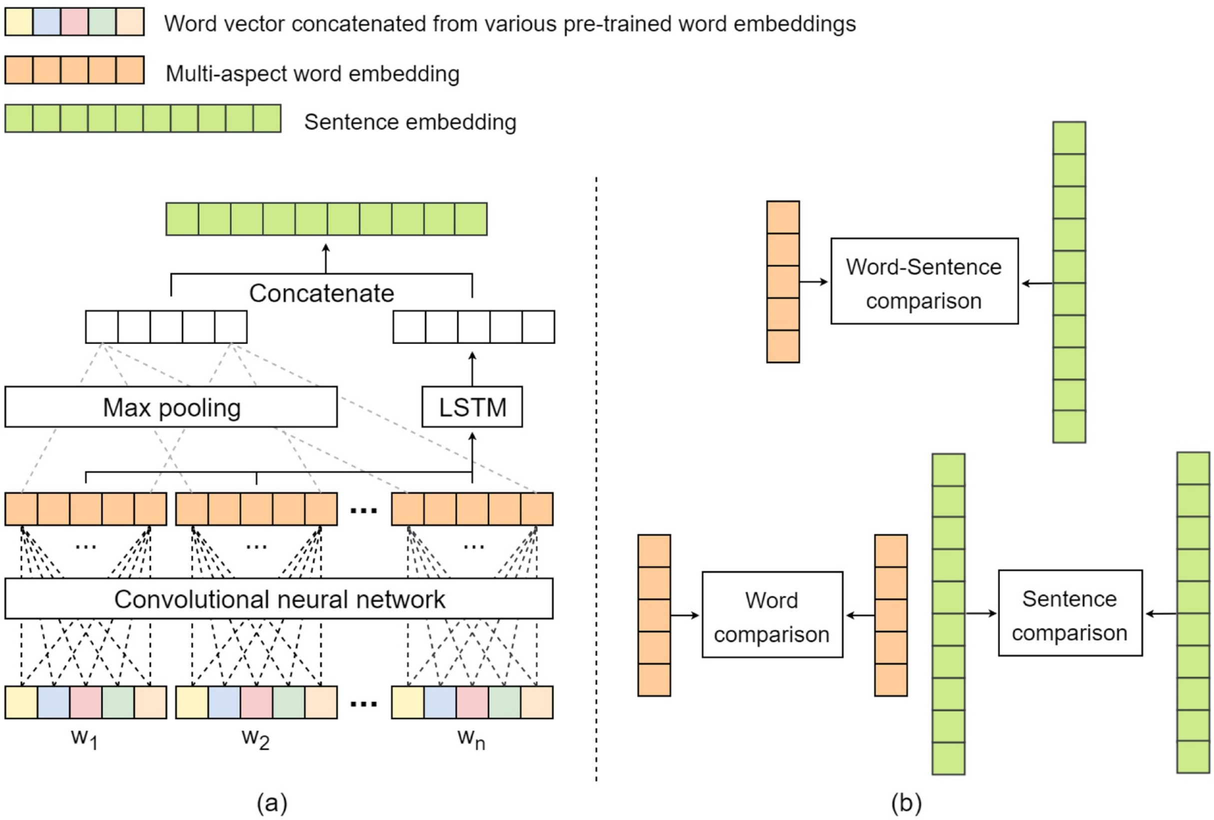

Fig. 1. The proposed M-MaxLSTM-CNN model: (a) MaxLSTM-CNN encoder; (b) Multi-level comparison.

by recursively applying a LSTM unit to each input $e_{w_t}^{multi}$ and the previous step h_{t-1}. At each time step t, the LSTM unit with l-memory dimension defines six vectors in $\mathbb{R}^l$: input gate i_t, forget gate f_t, output gate o_t, tanh layer u_t, memory cell c_t and hidden state h_t as follows (from Tai et al., 2015):

$$i_t = \sigma(W_i e_{w_t}^{multi} + U_i h_{t-1} + b_i) \tag{5}$$

$$f_t = \sigma(W_f e_{w_t}^{multi} + U_f h_{t-1} + b_f) \tag{6}$$

$$o_t = \sigma(W_o e_{w_t}^{multi} + U_o h_{t-1} + b_o) \tag{7}$$

$$u_t = \tanh(W_u e_{w_t}^{multi} + U_u h_{t-1} + b_u) \tag{8}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot u_t \tag{9}$$

$$h_t = o_t \odot \tanh(c_t) \tag{10}$$

$$e_s^{lstm} = h_n \tag{11}$$

where σ, $\odot$ respectively denote a logistic sigmoid function and element-wise multiplication; W_i, U_i, b_i are respectively two weights matrices and a bias vector for input gate i. The denotation is similar to forget gate f, output gate o, tanh layer u, memory cell c and hidden state h.

Finally, the sentence embedding e_s is obtained by concatenating the two sentence embeddings $e_s^{\max}$ and e_s^{lstm}:

$$e_s = e_s^{max} \oplus e_s^{lstm} \tag{12}$$

4.3. Multi-level comparison

In this section, we describe the process for evaluating the similarity/relation between two sentences. We compare two sentences via three levels: word-word, sentence-sentence and word-sentence.

4.3.1. Word-word comparison

Given two input sentences s_1 and s_2, we encode them into two sequences of multi-aspect word embeddings s_1^{multi} and s_2^{multi} (Section 3.2). We then compute a word-word similarity vector sim^{word} as follows:

$$A_{ij} = \frac{s_1^{multi}[i] \cdot s_2^{multi}[j]}{||s_1^{multi}[i]||\,||s_2^{multi}[j]||} \tag{13}$$

$$sim^{word} = \sigma(W^{word} g(A) + b^{word}) \tag{14}$$

where $s_t^{multi}[i]$ is the ith multi-aspect word embedding of sentence s_t; $g()$ is a function to flatten a matrix into a vector; and W^{word} and b^{word} are a weight matrix and a bias parameter, respectively.

4.3.2. Sentence-sentence comparison

Given two input sentences s_1 and s_2, we encode them into two sentence embeddings e_{s_1} and e_{s_2} (Section 3.1 and 3.2). To compute the similarity/relation between the two embeddings, we introduce four comparison metrics:

Cosine similarity:

$$d_{cosine} = \frac{e_{s_1} \cdot e_{s_1}}{||e_{s_1}||\,||e_{s_2}||} \tag{15}$$

Multiplication vector & Absolute difference:

$$d_{mul} = e_{s_1} \odot e_{s_2} \tag{16}$$

$$d_{abs} = |e_{s_1} - e_{s_2}| \tag{17}$$

where $\odot$ is element-wise multiplication.

Neural difference:

$$x = e_{s_1} \oplus e_{s_2} \tag{18}$$

$$d_{neu} = W^{neu} x + b^{neu} \tag{19}$$

where W^{neu} and b^{neu} are respectively a weight matrix and a bias parameter.

As a result, we have a sentence-sentence similarity vector sim^{sent} as follows:

$$d^{sent} = d_{cosine} \oplus d_{mul} \oplus d_{abs} \oplus d_{neu} \tag{20}$$

$$sim^{sent} = \sigma(W^{sent} d^{sent} + b^{sent}) \tag{21}$$

where W^{sent} and b^{sent} are respectively a weight matrix and a bias parameter.

4.3.3. Word-sentence comparison

Given a sentence embedding e_{s_1} and a sequence of multi-aspect word embeddings s_2^{multi}, we compute a word-sentence similarity matrix $sim_{s_1}^{ws}$ as follows:

$$e_{s_1}^{ws}[i] = e_{s_1} \oplus s_2^{multi}[i] \tag{22}$$

$$sim_{s_1}^{ws}[i] = \sigma\left(W^{ws} e_{s_1}^{ws}[i] + b^{ws}\right) \tag{23}$$

where $s_2^{multi}[i]$ is the multi-aspect word embedding of the ith word in sentence s_2; W^{ws} and b^{ws} are respectively a weight matrix and a bias parameter.

As a result, we have a word-sentence similarity vector sim^{ws} for the two sentences as follows:

$$sim^{ws} = \sigma\left(W^{ws'}\left[g\left(sim_{s_1}^{ws}\right) \oplus g\left(sim_{s_2}^{ws}\right)\right] + b^{ws'}\right) \tag{24}$$

where $g()$ is a function to flatten a matrix into a vector; $W^{ws'}$ and $b^{ws'}$ are respectively a weight matrix and a bias parameter.

Finally, we compute a target score/label of a sentence pair as follows:

$$sim = sim^{word} \oplus sim^{sent} \oplus sim^{ws} \tag{25}$$

$$h_s = \sigma(W^{l1} sim + b^{l1}) \tag{26}$$

$$\hat{y} = softmax(W^{l2} h_s + b^{l2}) \tag{27}$$

where W^{l1}, W^{l2}, b^{l1} and b^{l2} are model parameters; $\hat{y}$ is a predicted target score/label.

5. Tasks & datasets

We evaluate our model on three tasks:

Table 1
Statistic of datasets. $|V|$, l denote the vocabulary size, and the average length of sentences respectively.

Dataset	Train	Validation	Test	l	$\|V\|$
STSB	5749	1500	1379	11	15,997
SICK	4500	500	4927	9	2,312
MRPC	3576	500	1725	21	18,003

- **Textual entailment recognition**: Given a pair of sentences, we predict a directional relation between the sentences (entailment/contradiction/neutral). We evaluate this task on **SICK** dataset (Marelli et al., 2014). It was collected for the 2014 SemEval competition and includes examples of the lexical, syntactic and semantic phenomena and ignores other aspects of existing sentential datasets (i.e., idiomatic multiword expressions, named entities, telegraphic language).
- **Semantic textual similarity**: Given a pair of sentences, we measure a semantic similarity score of this pair. We use two datasets for this task:
  - **STSB** (Cer, Diab, Agirre, Lopez-Gazpio, & Specia, 2017): Comprises a careful selection of the English data sets used in SemEval and *SEM STS shared tasks from 2012 to 2017. This dataset cover three genres: Image captions, news headlines and user forums. Each sentence pair is annotated with a relatedness score $\in [0, 5]$.
  - **SICK** (Marelli et al., 2014): Each sentence pair is annotated with a relatedness score $\in [1, 5]$.
- **Paraphrase identification**: Given a pair of sentences, we predict a binary label indicating whether the two sentences are paraphrases. Microsoft Research Paraphrase Corpus (MRPC) (Dolan, Quirk, & Brockett, 2004) is used for this task. It includes pairs of sentences extracted from news source on the web.

Table 1 shows the statistic of the three datasets. Because of not dealing with name entities and multi-word idioms, the vocabulary size of SICK is quite small compared to the others.

6. Experimental setting

6.1. Pre-trained word embeddings

We study five pre-trained word embeddings for our model:

- **word2vec** is trained on Google News dataset (100 billion tokens). The model contains 300-dimensional vectors for 3 million words and phrases. word2vec is one of the first word embedding models and frequently employed in NLP deep learning researches.
- **fastText** is learned via skip-gram with subword information on Wikipedia text. The embedding representations in fastText are 300-dimensional vectors. A word is represented as the sum of n-gram vectors. This characteristic enhances the limitation of languages with large vocabularies and many rare words.
- **GloVe** is a 300-dimensional word embedding model learned on aggregated global word-word co-occurrence statistics from Common Crawl (840 billion tokens). Glove's advantage is to leverage statistical information by training only on the nonzero elements in a word-word co-occurrence matrix, rather than on the entire sparse matrix or on individual context windows in a large corpus.
- **SL999** is trained under the skip-gram objective with negative sampling on word pairs from the paraphrase database **PPDB**. This 300-dimensional embedding model is tuned on SimLex-999 dataset (Hill, Reichart, & Korhonen, 2016). The difference of SL999 from the others is to be trained on a paraphrase corpus, not a context corpus.
- **Baroni** uses a context-predict approach to learn a 400-dimensional semantic embedding model. It is trained on 2.8 billion tokens constructed from ukWaC, the English Wikipedia and the British National Corpus. Baroni is used for experiments to show the our model's efficiency in employing multiple word embeddings with different dimensions.

6.2. Model configuration

In all of the tasks, we used the same model configuration as follows:

- Convolutional filters: we used 1600 filters. It is also the dimension of the word embedding concatenated from the five pre-trained word embeddings.
- LSTM dimension: We also selected 1600 for LSTM dimension.
- Neural similarity layers: The dimension of b^{word}, b^{sent}, b^{ws} and $b^{ws'}$ are respectively 50, 5, 5 and 100.
- Penultimate fully-connected layer: Has the dimension of 250 and is followed by a drop-out layer ($p = 0.5$).

We conducted a grid search on 30% of STSB dataset to select these optimal hyper-parameters.

6.3. Training setting

6.3.1. Textual entailment recognition & paraphrase identification

In these tasks, we use the cross-entropy objective function and employ AdaDelta as the stochastic gradient descent (SGD) update rule with mini-batch size as 30. Details of Adadelta method can be found in Zeiler (2012). During the training phase, the pre-trained word embeddings are fixed.

6.3.2. Semantic textual similarity

To compute a similarity score of a sentence pair in the range [1, K], where K is an integer, we replace Eq. (27) with the equations in Tai et al. (2015) as follows:

$$\hat{p}_\theta = softmax(W^{l2}h_s + b^{l2}) \tag{28}$$

$$\hat{y} = r^T\hat{p}_\theta \tag{29}$$

where W^{l1}, W^{l2}, b^{l1} and b^{l2} are parameters; $r^T = [1, 2, ..., K]$; $\hat{y}$ is a predicted similarity score.

A sparse target distribution p which satisfies $y = r^T p$ is computed as:

$$p_i = \begin{cases} y - \lfloor y \rfloor, & i = \lfloor y \rfloor + 1 \\ \lfloor y \rfloor - y + 1, & i = \lfloor y \rfloor \\ 0 & otherwise \end{cases} \tag{30}$$

for $i \in [1, K]$, and y is the similarity score.

To train the model, we minimize the regularized KL-divergence between p and $\hat{p}_\theta$:

$$J(\theta) = \frac{1}{m}\sum_{k=1}^{m} KL(p^{(k)} \| \hat{p}_\theta^{(k)}) \tag{31}$$

where m is the number of training pairs and θ denotes the model parameters. The gradient descent optimization Adadelta is used to learn the model parameters. We also use mini-batch size as 30 and keep the pre-trained word embeddings fixed during the training phase. We evaluate our models through Pearson correlation r.

7. Experiments and discussion

This section describes two experiments: i) Compare our model against recent systems; ii) evaluate the efficiency of using multiple pre-trained word embeddings.

7.1. Overall evaluation

Besides existing methods, we also compare our model with several sentence modeling approaches using multiple pre-trained word embeddings:

- **Word average**:

$$e_s = \frac{1}{n}\sum_{i=1}^{n} e_{w_i}^{concat} \tag{32}$$

  where e_s is the sentence embedding of a n-words sentence, and $e_{w_i}^{concat}$ is from Eq. (1)
- **Project average**:

$$e_s = \sigma(W(\frac{1}{n}\sum_{i=1}^{n} e_{w_i}^{concat}) + b) \tag{33}$$

  where W is a 1600 × 1600 weight matrix, and b is a 1600 bias vector.
- **LSTM**: Apply Eqs. (5)–(11) on $e_{w_i}^{concat}$ to construct the 1600-dimension e_s sentence embedding.
- **Max-CNN**: Apply Eqs. (2)–(4) on $e_{w_i}^{concat}$ to construct the 1600-dimension e_s sentence embedding.

We report the results of these methods in Table 2. Overall, our M-MaxLSTM-CNN shows competitive performances in these tasks. Especially in the STS task, M-MaxLSTM-CNN outperforms the state-of-the-art methods on the two datasets. Because STSB includes complicated samples compared to SICK, the performances of methods on STSB are quite lower. In STSB, the prior top performance methods use ensemble approaches mixing hand-crafted features (word alignment, syntactic features, N-gram overlaps) and neural sentence representations, while our approach is only based on a neural sentence modeling architecture. In addition, we observed that InferSent shows the strong performance on SICK-R but quite low on STSB while our model consistently obtains the strong performances on both of the datasets. InferSent uses transfer knowledge on textual entailment data, consequently it obtains the strong performance on this entailment task. We also compared with the previous approaches using multiple word embeddings (i.e., MGNC-

Table 2
Test set results with Pearson's *r* score × 100 for STS tasks, and accuracy for other tasks. Boldface values show the highest scores in each dataset. SICK-R and SICK-E denote the STS task and the entailment task in SICK dataset respectively.

Method	STSB	SICK-R	SICK-E	MRPC
Ensemble models/Feature engineering				
DT_TEAM (Maharjan, Banjade, Gautam, Tamang, & Rus, 2017)	79.2	–	–	–
ECNU (Tian et al., 2017)	81	–	–	–
BIT (Wu et al., 2017)	80.9	–	–	–
TF-KLD (Ji & Eisenstein, 2013)	–	–	–	**80.41/85.96**
Neural representation models (NNM) with one embedding				
Multi-Perspective CNN (He et al., 2015)	–	86.86	–	78.6/84.73
InferSent (Conneau et al., 2017)	75.8	88.4	**86.1**	76.2/83.1
GRAN (Wieting & Gimpel, 2017)	76.4	86	–	–
Paragram-Phrase (Wieting et al., 2016b)	73.2	86.84	85.3	–
HCTI (Shao, 2017)	78.4	–	–	–
NNM with the five embeddings using sentence-sentence comparison (S)				
S-Word Average	71.06	81.18	80.88	71.48/81.1
S-Project Average	75.12	86.53	85.12	75.48/82.47
MGNC-CNN (Zhang et al., 2016)	72.11	82.1	81.72	74.91/82.79
MVCNN (Yin & Schütze, 2015)	72.41	82.7	82.12	75.4/82.1
S-LSTM	77.14	85.15	85.6	70.43/79.71
S-Max-CNN	81.87	88.3	84.33	76.35/83.75
S-MaxLSTM-CNN	82.2	88.47	84.9	77.91/84.31
NNM with the five embeddings using Multi-level comparison (M)				
M-Max-CNN	82.11	88.45	84.7	76.75/83.64
M-MaxLSTM-CNN	**82.45**	**88.76**	84.95	78.1/84.5

CNN and MVCNN). In MVCNN, we do not employ the Baroni word embeddings as its size is different from the others. Although these models are strong on sentence classification which requires to extract salient features for predicting targets, they do not work efficiently on capturing relationship between two sentences which requires the whole meaning of two sentences.

According to Wieting, Bansal, Gimpel, and Livescu (2016b), using Word Average as the compositional architecture outperforms the other architectures (e.g., Project Average, LSTM) for STS tasks. In a multiple word embeddings setting, however, Word Average does not show its efficiency. Each word embedding model has its own architecture as well as objective function. These factors makes the vector spaces of word embeddings are different. Therefore, we intuitively need a step to learn or refine a representation from a set of pre-trained word embeddings rather than only averaging them. Because Project Average model, LSTM model and Max-CNN model have their parameters for learning sentence embeddings, they significantly outperform Word Average model.

We observed that MaxLSTM-CNN outperforms Max-CNN in both of the settings (i.e., sentence-sentence comparison, Multi-level comparison). As mentioned in Section 1, Max-CNN ignores the property of word order. Therefore, our model achieves improvement compared to Max-CNN by additionally employing LSTM for capturing this property.

We only applied Multi-level comparison on Max-CNN and MaxLSTM-CNN because these encoders generate multi-aspect word embeddings. The experimental results prove the efficiency of using Multi-level comparison. In the textual entailment dataset SICK-E, the task mainly focuses on interpreting the meaning of a whole sentence pair rather than comparing word by word. Therefore, the performance of Multi-level comparison is quite similar to sentence-sentence comparison in the SICK-E task. This is also the reason why LSTM, which captures global relationships in sentences, has the strong performance in this task. In Section 6.3, we provide a deeper analysis explaining why LSTM's performance is better than our model's in SICK-E.

7.2. Evaluation of exploiting multiple pre-trained word embeddings

In this section, we evaluate the efficiency of using multiple pre-trained word embeddings. We compare our multiple pre-trained word embeddings model against models using only one pre-trained word embedding. The same objective function and Multi-level comparison are applied for these models. In case of using one pre-trained word embedding, the dimension of LSTM and the number of convolutional filters are set to the length of the corresponding word embedding. Table 3 shows the experimental results of this comparison. Because the approach using five word embeddings outperforms the approaches using two, three, or four word embeddings, we only report the performance of using five word embeddings. We also report $|V|_{avai}$ which is the proportion of vocabulary available in a pre-trained word embedding. SICK dataset ignores idiomatic multi-word expressions, and named entities, consequently the $|V|_{avai}$ of SICK is quite high.

We observed that no word embedding has strong results on all the tasks. Although trained on the paraphrase database and having the highest $|V|_{avai}$, the SL999 embedding could not outperform the Glove embedding in SICK-R. HCTI (Shao, 2017), which is the current state-of-the-art in the group of neural representation models on STSB, also used the Glove embedding. However, the performance of HTCI in STSB (78.4) is lower than our model using the Glove embedding. In SICK-R, InferSent (Conneau et al., 2017) achieves a strong performance (88.4) using the Glove embedding with transfer knowledge, while our model with only the Glove embedding achieves a performance close to the performance of InferSent. These results confirm the efficiency of Multi-level comparison.

Table 3
Evaluation of exploiting multiple pre-trained word embeddings. $|V|_{avai}$ is the proportion of vocabulary available in a word embedding. In case of using all word embeddings, $|V|_{avai}$ denotes the proportion of vocabulary where each word is available in at least one word embedding.

Word embedding	STSB		SICK-R & SICK-E			MRPC	
	Pearson	$\|V\|_{avai}$(%)	Pearson	Acc	$\|V\|_{avai}$(%)	Acc/F1	$\|V\|_{avai}$(%)
word2Vec	78.9	75.64	87.27	84.09	98.53	75.42/82.13	67.81
fastText	79.95	84.27	87.59	83.42	99.18	74.31/81.75	79.04
Glove	80.1	91.71	88.21	84.71	99.78	74.9/82.782	89.85
SL999	80.31	94.76	87.26	84.55	99.83	76.46/83.13	94.19
Baroni	79.81	90.54	86.9	83.99	98.83	74.84/82.4	87.92
Glove + SL999	81.14	95.07	88.28	84.45	99.83	76.17/83.01	94.29
Glove + SL999 + fastText	81.73	95.45	88.38	84.91	99.83	76.46/83.22	94.83
Glove + SL999 + fastText + Baroni	82.16	95.65	88.74	84.94	99.83	76.63/82.99	95.06
All	**82.45**	95.65	**88.76**	**84.95**	99.83	**78.1/84.5**	95.97

In STSB and MRPC, as employing the five pre-trained embeddings, the $|V|_{avai}$ is increased. This factor limits the number of random values when initializing word embedding representations because a word out of a pre-trained word embedding is assigned a random word embedding representation. In other words, a word out of a pre-trained word embedding is assigned a random semantic meaning. Therefore, the increase of the $|V|_{avai}$ improves the performance of measuring textual similarity. In STSB and MRPC, our multiple pre-trained word embedding achieves a significant improvement in performance compared against using one word embedding. In SICK-R and SICK-E, although the $|V|_{avai}$ is not increased when employing five pre-trained embeddings, the performance of our model is improved. This fact shows that our model learned an efficient word embedding via these pre-trained word embeddings.

7.3. Quality analysis

We manually inspect some samples to analysis the advantages and disadvantages of our model (listed in Table 4). To answer sample #1, our multi-word embeddings model well evaluates the sharing meaning between words *(man ~ person* and *horse ~ animal*) compared to single-word embedding models. This capability is a fundamental requirement for STS tasks. However, totally basing on this measurement is not enough for textual entailment tasks. As in sample #2, the rule from sample #1 could not be applied (i.e., *people* and *spectator* are not interchange in the context of sample #2). The degree of word similarity has to consider contextual information. A context-based similarity evaluator is the promising approach for textual entailment tasks. That is the reason why LSTM focusing on comparing the whole context meanings rather than each word has a strong performance in this task.

In sample #3 and #4, we observed these two pairs share some phrases (e.g., *your best bet, a good idea*). Although having the same phrases, these pairs are assigned low similarity scores by human, which contradicts our model. In these samples, each word or phrase contributes to its sentence at different degrees. For example, *"microwave"* and *"research"* are more important than *"your best bet"*. The word "just" usually does not contribute so much to its sentence meaning. However, in sample #4, it changes the whole meaning of the sentence. Therefore, the role or contribution of each word in a sentence should be considered for evaluating sentence similarity or textual entailment.

Compared to STS tasks, Paraphrase identification has a little different rule which requires a paraphrase sentence pair to share the same meaning and use different words, phrases or forms. In sample #5, the sentence pair shares the form and almost words, so they are not called paraphrases. As a result, similarity measurement models without the constrain of using different forms, words fail to handle the cases as sample #5. In addition, some cases require extra knowledge as sample #6. Without prior-knowledge, it is hard to link *"victims"* to *"family"*. These challenges make the paraphrase tasks difficult.

Table 4
Some typical samples for quality analysis.

No	Sentence pair	Proposed	True
1	A man is riding a horse the person is riding the animal	Entailment	Entailment
2	Two spectators are kickboxing and some people are watching two people are kickboxing and spectators are watching	Entailment	Neutral
3	Microwave would be your best bet. your best bet is research.	2.8	0
4	It 's not a good idea. it 's not just a good idea, it 's an excellent idea.	3.9	1
5	" this is america, my friends, and it should not happen here, " he said to loud applause. " this is america, my friends, and it should not happen here. "	Paraphrase	Not paraphrase
6	The victims were last seen at church last sunday ; their bodies were discovered tuesday. the family was last seen july 6, and their bodies were found tuesday.	Not paraphrase	Paraphrase

8. Conclusion

In this work, we study an approach employing multiple pre-trained word embeddings and Multi-level comparison for measuring semantic textual relation. The proposed M-MaxLSTM-CNN architecture consistently obtains strong performances on several tasks. Compared to the state-of-the art methods in STS tasks, our model does not require handcrafted features (e.g., word alignment, syntactic features) as well as transfer learning knowledge. In addition, it allows using several pre-trained word embeddings with different dimensions.

In future work, we could apply our multiple word embeddings approach for transfer learning tasks. This strategy allows making use of pre-trained word embeddings as well as available resources.

Acknowledgments

This work was done while Nguyen Tien Huy was an intern at Toshiba Research Center.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at 10.1016/j.ipm.2019.102090 .

References

AL-Smadi, M., Jaradat, Z., AL-Ayyoub, M., & Jararweh, Y. (2017). Paraphrase identification and semantic text similarity analysis in arabic news tweets using lexical, syntactic, and semantic features. *Information Processing & Management, 53*(3), 640–652.

Bengio, Y., Ducharme, R., Vincent, P., & Jauvin, C. (2003). A neural probabilistic language model. *Journal of Machine Learning Research, 3*(Feb), 1137–1155.

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association of Computational Linguistics, 5*, 135–146.

Camacho-Collados, J., Pilehvar, M. T., & Navigli, R. (2016). Nasari: Integrating explicit knowledge and corpus statistics for a multilingual representation of concepts and entities. *Artificial Intelligence, 240*, 36–64.

Cer, D., Diab, M., Agirre, E., Lopez-Gazpio, I., & Specia, L. (2017). *SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. Proceedings of the 11th international workshop on semantic evaluation (SemEval-2017)*. Vancouver, Canada: Association for Computational Linguistics1–14.

Conneau, A., Kiela, D., Schwenk, H., Barrault, L., & Bordes, A. (2017). Supervised learning of universal sentence representations from natural language inference data. *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*.

Das, D., & Smith, N. A. (2009). *Paraphrase identification as probabilistic quasi-synchronous recognition. Proceedings of the joint conference of the 47th annual meeting of the acl and the 4th international joint conference on natural language processing of the AFNLP: Volume 1-volume 1*468–476 Association for Computational Linguistics

Dolan, B., Quirk, C., & Brockett, C. (2004). *Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. Proceedings of coling 2004*. Geneva, Switzerland: COLING350–356.

Ferreira, R., Cavalcanti, G. D., Freitas, F., Lins, R. D., Simske, S. J., & Riss, M. (2018). Combining sentence similarities measures to identify paraphrases. *Computer Speech & Language, 47*, 59–73. https://doi.org/10.1016/j.csl.2017.07.002.

Ferreira, R., Lins, R. D., Simske, S. J., Freitas, F., & Riss, M. (2016). Assessing sentence similarity through lexical, syntactic and semantic analysis. *Computer Speech & Language, 39*, 1–28. https://doi.org/10.1016/j.csl.2016.01.003.

Gan, Z., Pu, Y., Henao, R., Li, C., He, X., & Carin, L. (2017). *Learning generic sentence representations using convolutional neural networks. Proceedings of the 2017 conference on empirical methods in natural language processing*2380–2390.

He, H., Gimpel, K., & Lin, J. J. (2015). *Multi-perspective sentence similarity modeling with convolutional neural networks. Proceedings of the 2015 conference on empirical methods in natural language processing (EMNLP)*1576–1586.

Hill, F., Reichart, R., & Korhonen, A. (2016). Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*.

Huy, N. T., & Nguyen, M. L. (2018). Multilingual opinion mining on youtube a convolutional n-gram bilstm word embedding. *Information Processing & Management, 54*(3), 451–462.

Huy Tien, N., & Minh Le, N. (2017). *An ensemble method with sentiment features and clustering support. Proceedings of the eighth international joint conference on natural language processing (volume 1: Long papers)*. Taipei, Taiwan: Asian Federation of Natural Language Processing644–653.

Ji, Y., & Eisenstein, J. (2013). *Discriminative improvements to distributional sentence similarity. Proceedings of the 2013 conference on empirical methods in natural language processing*. Association for Computational Linguistics891–896.

Jiang, Y., Bai, W., Zhang, X., & Hu, J. (2017). Wikipedia-based information content and semantic similarity computation. *Information Processing & Management, 53*(1), 248–265.

Kalchbrenner, N., Grefenstette, E., & Blunsom, P. (2014). *A convolutional neural network for modeling sentences. Proceedings of the 52nd annual meeting of the association for computational linguistics (volume 1: Long papers)*. Association for Computational Linguistics655–665. https://doi.org/10.3115/v1/P14-1062.

Kim, Y. (2014). *Convolutional neural networks for sentence classification. Proceedings of the 2014 conference on empirical methods in natural language processing (emnlp)*. Association for Computational Linguistics1746–1751. https://doi.org/10.3115/v1/D14-1181.

Levy, O., & Goldberg, Y. (2014). *Dependency-based word embeddings. Proceedings of the 52nd annual meeting of the association for computational linguistics*302–308.

Madnani, N., Tetreault, J., & Chodorow, M. (2012). *Re-examining machine translation metrics for paraphrase identification. Proceedings of the 2012 conference of the north american chapter of the association for computational linguistics: Human language technologies*. Association for Computational Linguistics182–190.

Maharjan, N., Banjade, R., Gautam, D., Tamang, L. J., & Rus, V. (2017). *Dt_team at semeval-2017 task 1: Semantic similarity using alignments, sentence-level embeddings and gaussian mixture model output. Proceedings of the 11th international workshop on semantic evaluation (semeval-2017)*120–124.

Marelli, M., Bentivogli, L., Baroni, M., Bernardi, R., Menini, S., & Zamparelli, R. (2014). *SemEval-2014 task 1: Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment. Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)*. Dublin, Ireland: Association for Computational Linguistics1–8.

Mihalcea, R., Corley, C., Strapparava, C., et al. (2006). *Corpus-based and knowledge-based measures of text semantic similarity. Aaai6. Aaai* 775–780.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). *Efficient estimation of word representations in vector space. Proceedings of workshop at the international conference on learning representations*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). *Distributed representations of words and phrases and their compositionality. Advances in neural information processing systems*3111–3119.

Qu, R., Fang, Y., Bai, W., & Jiang, Y. (2018). Computing semantic similarity based on novel models of semantic representation using wikipedia. *Information Processing & Management, 54*(6), 1002–1021.

Shao, Y. (2017). *Hcti at semeval-2017 task 1: Use convolutional neural network to evaluate semantic textual similarity. Proceedings of the 11th international workshop on semantic evaluation (semeval-2017)*130–133.

Sultan, M. A., Bethard, S., & Sumner, T. (2014). Back to basics for monolingual alignment: Exploiting word similarity and contextual evidence. *Transactions of the Association of Computational Linguistics, 2*, 219–230.

Tai, K. S., Socher, R., & Manning, C. D. (2015). *Improved semantic representations from tree-structured long short-term memory networks. 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)*. Association for Computational Linguistics1556–1566.

Tian, J., Zhou, Z., Lan, M., & Wu, Y. (2017). *Ecnu at semeval-2017 task 1: Leverage kernel-based traditional nlp features and neural networks to build a universal model for multilingual and cross-lingual semantic textual similarity. Proceedings of the 11th international workshop on semantic evaluation (semeval-2017)*191–197.

Wan, S., Dras, M., Dale, R., & Paris, C. (2006). *Using dependency-based features to take the œpara-farceg out of paraphrase*.

Wieting, J., Bansal, M., Gimpel, K., & Livescu, K. (Bansal, Gimpel, Livescu, 2016a). *Charagram: Embedding words and sentences via character n-grams. Proceedings of the 2016 conference on empirical methods in natural language processing*1504–1515.

Wieting, J., Bansal, M., Gimpel, K., & Livescu, K. (Bansal, Gimpel, Livescu, 2016b). Towards universal paraphrastic sentence embeddings. *The International Conference on Learning Representations*.

Wieting, J., Bansal, M., Gimpel, K., Livescu, K., & Roth, D. (2015). From paraphrase database to compositional paraphrase model and back. *Transactions of the ACL (TACL)*.

Wieting, J., & Gimpel, K. (2017). *Revisiting recurrent networks for paraphrastic sentence embeddings. ACL (1)*. Association for Computational Linguistics2078–2088.

Wu, H., Huang, H., Jian, P., Guo, Y., & Su, C. (2017). *Bit at semeval-2017 task 1: Using semantic information space to evaluate semantic textual similarity. Proceedings of the 11th international workshop on semantic evaluation (semeval-2017)*77–84.

Xu, W., Ritter, A., Callison-Burch, C., Dolan, W. B., & Ji, Y. (2014). Extracting lexically divergent paraphrases from twitter. *Transactions of the Association of Computational Linguistics, 2*, 435–448.

Yessenalina, A., & Cardie, C. (2011). *Compositional matrix-space models for sentiment analysis. Proceedings of the conference on empirical methods in natural language processing*172–182 Association for Computational Linguistics

Yin, W., & Schütze, H. (2015). *Multichannel variable-size convolution for sentence classification. Proceedings of the nineteenth conference on computational natural language learning*. Association for Computational Linguistics204–214.

Zeiler, M. D. (2012). ADADELTA: An adaptive learning rate method. *CoRR, abs/1212.5701*.

Zhang, Y., Roller, S., & Wallace, B. C. (2016). *Mgnc-cnn: A simple approach to exploiting multiple word embeddings for sentence classification. Proceedings of the 2016 conference of the north american chapter of the association for computational linguistics: Human language technologies*. Association for Computational Linguistics1522–1527. https://doi.org/10.18653/v1/N16-1178.