Pairwise Word Interaction Modeling with Deep Neural Networks for Semantic Similarity Measurement

Hua He[1] and **Jimmy Lin**[2]

[1] Department of Computer Science, University of Maryland, College Park
[2] David R. Cheriton School of Computer Science, University of Waterloo

huah@umd.edu, jimmylin@uwaterloo.ca

Abstract

Textual similarity measurement is a challenging problem, as it requires understanding the semantics of input sentences. Most previous neural network models use coarse-grained sentence modeling, which has difficulty capturing fine-grained word-level information for semantic comparisons. As an alternative, we propose to explicitly model pairwise word interactions and present a novel similarity focus mechanism to identify important correspondences for better similarity measurement. Our ideas are implemented in a novel neural network architecture that demonstrates state-of-the-art accuracy on three SemEval tasks and two answer selection tasks.

1 Introduction

Given two pieces of text, measuring their semantic textual similarity (STS) remains a fundamental problem in language research and lies at the core of many language processing tasks, including question answering (Lin, 2007), query ranking (Burges et al., 2005), and paraphrase generation (Xu, 2014).

Traditional NLP approaches, e.g., developing hand-crafted features, suffer from sparsity because of language ambiguity and the limited amount of annotated data available. Neural networks and distributed representations can alleviate such sparsity, thus neural network-based models are widely used by recent systems for the STS problem (He et al., 2015; Tai et al., 2015; Yin and Schütze, 2015).

However, most previous neural network approaches are based on *sentence modeling*, which first maps each input sentence into a fixed-length vector and then performs comparisons on these representations. Despite its conceptual simplicity, researchers have raised concerns about this approach (Mooney, 2014): Will fine-grained word-level information, which is crucial for similarity measurement, get lost in the coarse-grained sentence representations? Is it really effective to "cram" whole sentence meanings into fixed-length vectors?

In contrast, we focus on capturing fine-grained word-level information directly. Our contribution is twofold: First, instead of using sentence modeling, we propose *pairwise word interaction modeling* that encourages explicit word context interactions across sentences. This is inspired by our own intuitions of how people recognize textual similarity: given two sentences $sent_1$ and $sent_2$, a careful reader might look for corresponding semantic units, which we operationalize in our pairwise word interaction modeling technique (Sec. 5). Second, based on the pairwise word interactions, we describe a novel *similarity focus layer* which helps the model selectively identify important word interactions depending on their importance for similarity measurement. Since not all words are created equal, important words that can make more contributions deserve extra "focus" from the model (Sec. 6).

We conducted thorough evaluations on ten test sets from three SemEval STS competitions (Agirre et al., 2012; Marelli et al., 2014; Agirre et al., 2014) and two answer selection tasks (Yang et al., 2015; Wang et al., 2007). We outperform the recent multi-perspective convolutional neural networks of He et al. (2015) and demonstrate state-of-the-art accuracy on all five tasks. In addition, we conducted ablation studies and visualized our models to show the clear benefits of modeling pairwise word interactions for similarity measurement.

2 Related Work

Feature engineering was the dominant approach in most previous work; different types of sparse features were explored and found useful. For example, n-gram overlap features at the word and character levels (Madnani et al., 2012; Wan et al., 2006), syntax features (Das and Smith, 2009; Xu et al., 2014), knowledge-based features using WordNet (Fellbaum, 1998; Fern and Stevenson, 2008) and word-alignment features (Sultan et al., 2014).

The recent shift from sparse feature engineering to neural network model engineering has significantly improved accuracy on STS datasets. Most previous work use sentence modeling with a "Siamese" structure (Bromley et al., 1993). For example, Hu et al. (2014) used convolutional neural networks that combine hierarchical structures with layer-by-layer composition and pooling. Tai et al. (2015) and Zhu et al. (2015) concurrently proposed tree-structured long short-term memory networks, which recursively construct sentence representations following their syntactic trees. There are many other examples of neural network-based sentence modeling approaches for the STS problem (Yin and Schütze, 2015; Huang et al., 2013; Andrew et al., 2013; Weston et al., 2011; Socher et al., 2011; Zarrella et al., 2015).

Sentence modeling is coarse-grained by nature. Most recently, despite still using a sentence modeling approach, He et al. (2015) moved toward fine-grained representations by exploiting multiple perspectives of input sentences with different types of convolution filters and pooling, generating a "matrix" representation where rows and columns capture different aspects of the sentence; comparisons over local regions of the representation are then performed. He et al. (2015) achieves highly competitive accuracy, suggesting the usefulness of fine-grained information. However, these multiple perspectives are obtained at the cost of increased model complexity, resulting in slow model training. In this work, we take a different approach by focusing directly on pairwise word interaction modeling.

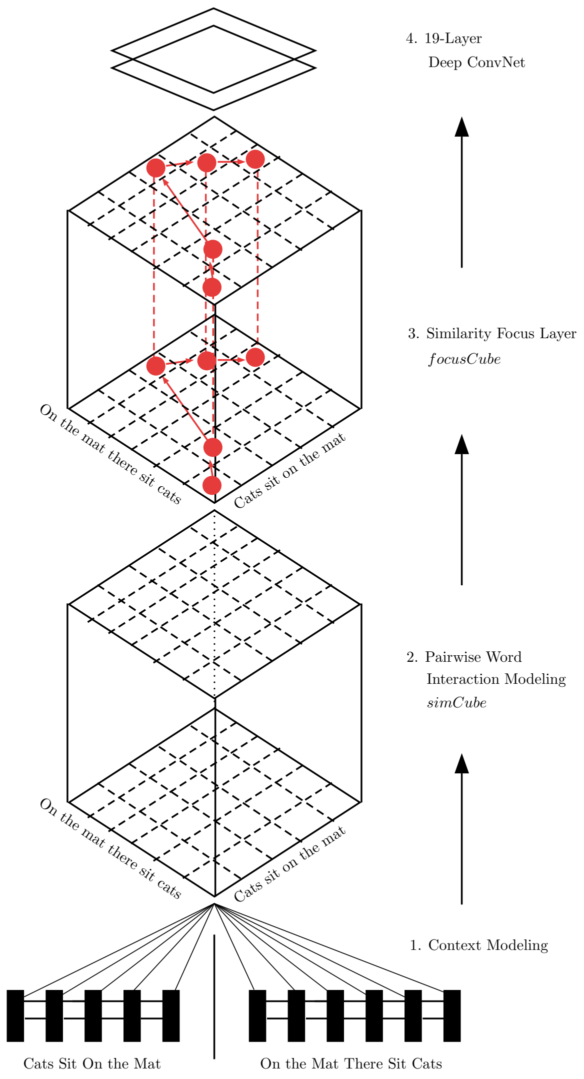

Figure 1: Our end-to-end neural network model, consisting of four major components.

3 Model Overview

Figure 1 shows our end-to-end model with four major components:

1. Bidirectional Long Short-Term Memory Networks (Bi-LSTMs) (Graves et al., 2005; Graves et al., 2006) are used for context modeling of input sentences, which serves as the basis for all following components (Sec. 4).
2. A novel pairwise word interaction modeling technique encourages direct comparisons between word contexts across sentences (Sec. 5).
3. A novel similarity focus layer helps the model identify important pairwise word interactions across sentences (Sec. 6).

4. A 19-layer deep convolutional neural network (ConvNet) converts the similarity measurement problem into a pattern recognition problem for final classification (Sec. 7).

To our best knowledge this is the first neural network model, a novel hybrid architecture combining Bi-LSTMs and a deep ConvNet, that uses a similarity focus mechanism with selective attention to important pairwise word interactions for the STS problem. Our approach only uses pretrained word embeddings, and unlike several previous neural network models (Yin and Schütze, 2015; Tai et al., 2015), we do not use sparse features, unsupervised model pretraining, syntactic parsers, or external resources like WordNet. We describe details of each component in the following sections.

4 Context Modeling

Different words occurring in similar semantic contexts of respective sentences have a higher chance to contribute to the similarity measurement. We therefore need word context modeling, which serves as a basis for all following components of this work.

LSTM (Hochreiter and Schmidhuber, 1997) is a special variant of Recurrent Neural Networks (Williams and Zipser, 1989). It can capture long-range dependencies and nonlinear dynamics between words, and has been successfully applied to many NLP tasks (Sutskever et al., 2014; Filippova et al., 2015). LSTM has a memory cell that can store information over a long history, as well as three gates that control the flow of information into and out of the memory cell. At time step t, given an input x_t, previous output h_{t-1}, input gate i_t, output gate o_t and forget gate f_t, $LSTM(x_t, h_{t-1})$ outputs the hidden state h_t based on the equations below:

$$i_t = \sigma(W^i x_t + U^i h_{t-1} + b^i) \quad (1)$$
$$f_t = \sigma(W^f x_t + U^f h_{t-1} + b^f) \quad (2)$$
$$o_t = \sigma(W^o x_t + U^o h_{t-1} + b^o) \quad (3)$$
$$u_t = tanh(W^u x_t + U^u h_{t-1} + b^u) \quad (4)$$
$$c_t = i_t \cdot u_t + f_t \cdot c_{t-1} \quad (5)$$
$$h_t = o_t \cdot tanh(c_t) \quad (6)$$
$$LSTM(x_t, h_{t-1}) = h_t \quad (7)$$
$$BiLSTMs(x_t, h_{t-1}) = \{LSTM^f, LSTM^b\} \quad (8)$$

where σ is the logistic sigmoid activation, W^*, U^* and b^* are learned weight matrices and biases. LSTMs are better than RNNs for context modeling, in that their memory cells and gating mechanisms handle the vanishing gradients problem in training.

We use bidirectional LSTMs (Bi-LSTMs) for context modeling in this work. Bi-LSTMs consist of two LSTMs that run in parallel in opposite directions: one (forward $LSTM^f$) on the input sequence and the other (backward $LSTM^b$) on the reverse of the sequence. At time step t, the Bi-LSTMs hidden state h_t^{bi} is a concatenation of the hidden state h_t^{for} of $LSTM^f$ and the hidden state h_t^{back} of $LSTM^b$, representing the neighbor contexts of input x_t in the sequence. We define the *unpack* operation below:

$$h_t^{for}, h_t^{back} = unpack(h_t^{bi}) \quad (9)$$

Context modeling with Bi-LSTMs allows all the following components to be built over word contexts, rather than over individual words.

5 Pairwise Word Interaction Modeling

From our own intuitions, given two sentences in a STS task, a careful human reader might compare words and phrases across the sentences to establish semantic correspondences and from these infer similarity. Our pairwise word interaction model is inspired by such behavior: whenever the next word of a sentence is read, the model would compare it and its context against all words and their contexts in the other sentence. Figure 2 illustrates this model.

We first define a comparison unit for comparing two hidden states $\overrightarrow{h_1}, \overrightarrow{h_2}$ of Bi-LSTMs.

$$coU(\overrightarrow{h_1}, \overrightarrow{h_2}) = \{\cos(\overrightarrow{h_1}, \overrightarrow{h_2}), L_2Euclid(\overrightarrow{h_1}, \overrightarrow{h_2}), DotProduct(\overrightarrow{h_1}, \overrightarrow{h_2})\} \quad (10)$$

Cosine distance (cos) measures the distance of two vectors by the angle between them, while L_2 Euclidean distance ($L_2Euclid$) and dot-product distance ($DotProduct$) measure magnitude differences. We use three similarity functions for richer measurement.

Algorithm 1 provides details of the modeling process. Given the input $x_t^a \in sent_a$ at time step t where $a \in \{1, 2\}$, its Bi-LSTMs hidden state h_{at}^{bi} is the concatenation of the forward state h_{at}^{for} and the

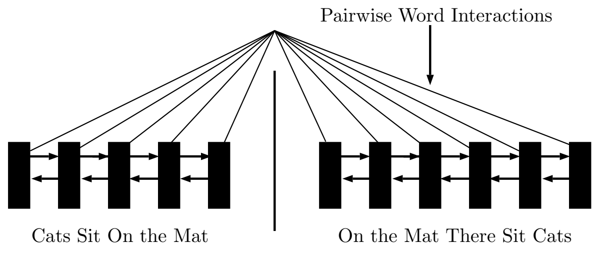

Figure 2: Pairwise word interaction modeling. Sentences are encoded by weight-shared Bi-LSTMs. We construct pairwise word interactions for context comparisons across sentences.

Algorithm 1 Pairwise Word Interaction Modeling

```
1:  Initialize: $simCube \in R^{13 \cdot |sent_1| \cdot |sent_2|}$ to all 1
2:  for each time step $t = 1...|sent_1|$ do
3:      for each time step $s = 1...|sent_2|$ do
4:          $h_{1t}^{for}, h_{1t}^{back} = unpack(h_{1t}^{bi})$
5:          $h_{2s}^{for}, h_{2s}^{back} = unpack(h_{2s}^{bi})$
6:          $h_{1t}^{add} = h_{1t}^{for} + h_{1t}^{back}$
7:          $h_{2s}^{add} = h_{2s}^{for} + h_{2s}^{back}$
8:          $simCube[1:3][t][s] = coU(h_{1t}^{bi}, h_{2s}^{bi})$
9:          $simCube[4:6][t][s] = coU(h_{1t}^{for}, h_{2s}^{for})$
10:         $simCube[7:9][t][s] = coU(h_{1t}^{back}, h_{2s}^{back})$
11:         $simCube[10:12][t][s] = coU(h_{1t}^{add}, h_{2s}^{add})$
12:     end for
13: end for
14: return $simCube$
```

backward state h_{at}^{back}. Algorithm 1 proceeds as follows: it enumerates all word pairs (s, t) across both sentences, then perform comparisons using the coU unit four times over: 1) Bi-LSTMs hidden states h_{1t}^{bi} and h_{2s}^{bi}; 2) forward hidden states h_{1t}^{for} and h_{2s}^{for}; 3) backward hidden states h_{1t}^{back} and h_{2s}^{back}; and 4) the addition of forward and backward hidden states h_{1t}^{add} and h_{2s}^{add}. The output of Algorithm 1 is a similarity cube $simCube$ with size $R^{13 \cdot |sent_1| \cdot |sent_2|}$, where $|sent_*|$ is the number of words in the sentence $sent_*$. The 13 values collected from each word pair (s, t) are: the 12 similarity distances, plus one extra dimension for the padding indicator. Note that all word interactions are modeled over word contexts in Algorithm 1, rather than individual words.

Our pairwise word interaction model shares similarities with recent popular neural attention models (Bahdanau et al., 2014; Rush et al., 2015). However, there are important differences: For example, we do not use attention weight vectors or weighted representations, which are the core of attention models. The other difference is that attention weights are typically interpreted as soft degrees with which the model attends to particular words; in contrast, our word interaction model directly utilizes multiple similarity metrics, and thus is more explicit.

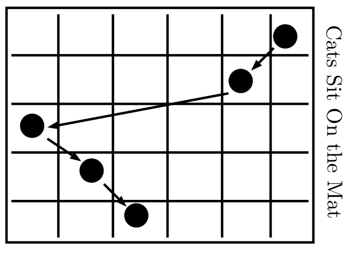

Figure 3: The similarity focus layer helps identify important pairwise word interactions (in black dots) depending on their importance for similarity measurement.

6 Similarity Focus Layer

Since not all words are created equal, important pairwise word interactions between the sentences (Sec. 5) that can better contribute to the similarity measurement deserve more model focus. We therefore develop a similarity focus layer which can identify important word interactions and increase their model weights correspondingly. This similarity focus layer is directly incorporated into our end-to-end model and is placed on top of the pairwise word interaction model, as in Figure 1.

Figure 3 shows one example where each cell of the matrix represents a pairwise word interaction. The similarity focus layer introduces re-weightings to word interactions depending on their importance for similarity measurement. The ones tagged with black dots are considered important, and are given higher weights than those without.

Algorithm 2 shows the forward pass of the similarity focus layer. Its input is the similarity cube $simCube$ (Section 5). Algorithm 2 is designed to incorporate two different aspects of similarity based on $cosine$ (angular) and $L2$ (magnitude) similarity, thus it has two symmetric components: the first one is based on $cosine$ similarity (Line 5 to Line 13); and the second one is based on $L2$ similarity (Line 15 to Line 23). We also aim for the goal that similarity values of all found important word in-

Algorithm 2 Forward Pass: Similarity Focus Layer

```
1: Input: $simCube \in R^{13 \cdot |sent_1| \cdot |sent_2|}$
2: Initialize: $mask \in R^{13 \cdot |sent_1| \cdot |sent_2|}$ to all 0.1
3: Initialize: $s1tag \in R^{|sent_1|}$ to all zeros
4: Initialize: $s2tag \in R^{|sent_2|}$ to all zeros
5: $sortIndex_1 = sort(simCube[10])$
6: for each $id = 1...|sent_1| + |sent_2|$ do
7:     $pos_{s1}, pos_{s2} = calcPos(id, sortIndex_1)$
8:     if $s1tag[pos_{s1}] + s2tag[pos_{s2}] == 0$ then
9:         $s1tag[pos_{s1}] = 1$
10:        $s2tag[pos_{s2}] = 1$
11:        $mask[:][pos_{s1}][pos_{s2}] = 1$
12:    end if
13: end for
14: Re-Initialize: $s1tag, s2tag$ to all zeros
15: $sortIndex_2 = sort(simCube[11])$
16: for each $id = 1...|sent_1| + |sent_2|$ do
17:     $pos_{s1}, pos_{s2} = calcPos(id, sortIndex_2)$
18:     if $s1tag[pos_{s1}] + s2tag[pos_{s2}] == 0$ then
19:         $s1tag[pos_{s1}] = 1$
20:         $s2tag[pos_{s2}] = 1$
21:         $mask[:][pos_{s1}][pos_{s2}] = 1$
22:     end if
23: end for
24: $mask[13][:][:] = 1$
25: $focusCube = mask \cdot simCube$
26: return $focusCube \in R^{13 \cdot |sent_1| \cdot |sent_2|}$
```

teractions should be maximized. To achieve this, we sort the similarity values in descending order (Line 5 for $cosine$, Line 15 for $L2$). Note channels 10 and 11 of the $simCube$ contain $cosine$ and $L2$ values, respectively; the padding indicator is in Line 24.

We start with the $cosine$ part first, then $L2$. For each, we check word interaction candidates moving down the sorted list. Function $calcPos$ is used to calculate relative sentence positions pos_{s*} in the $simCube$ given one interaction pair. We follow the constraint that no word in both sentences should be tagged to be important more than once. We increase weights of important word interactions to 1 (in Line 11 based on $cosine$ and Line 21 based on $L2$), while unimportant word interactions receive weights of 0.1 (in Line 2).

We use a mask matrix, $mask$, to hold the weights of each. The final output of the similarity focus layer is a focus-weighted similarity cube $focusCube$, which is the element-wise multiplication (Line 25) of the matrix $mask$ and the input $simCube$.

The similarity focus layer is based on the following intuition: given each word in one sentence, we look for its semantically similar twin in the other sentence; if found then this word is considered important, otherwise it contributes to a semantic difference. Though technically different, this process shares conceptual similarity with finding translation equivalences in statistical machine translation (Al-onaizan et al., 1999).

The backward pass of the similarity focus layer is straightforward: we reuse the $mask$ matrix as generated in the forward pass and apply the element-wise multiplication of $mask$ and inflow gradients, then propagate the resulting gradients backward.

7 Similarity Classification with Deep Convolutional Neural Networks

The $focusCube$ contains focus-weighted fine-grained similarity information. In the final model component we use the $focusCube$ to compute the final similarity score. If we treat the $focusCube$ as an "image" with 13 channels, then semantic similarity measurement can be converted into a pattern recognition (image processing) problem, where we are looking for patterns of strong pairwise word interactions in the "image". The stronger the overall pairwise word interactions are, the higher similarity the sentence pair will have.

Recent advances from successful systems at ImageNet competitions (Simonyan and Zisserman, 2014; Szegedy et al., 2015) show that the depth of a neural network is a critical component for achieving competitive performance. We therefore use a deep homogeneous architecture which has repetitive convolution and pooling layers.

Our network architecture (Table 1) is composed of spatial max pooling layers, spatial convolutional layers (Conv) with a small filter size of 3×3 plus stride 1 and padding 1. We adopt this filter size because it is the smallest one to capture the space of left/right, up/down, and center; the padding and stride is used to preserve the spatial input resolution. We then use fully-connected layers followed by the final softmax layer for the output. After each spatial convolutional layer, a rectified linear units (ReLU) non-linearity layer (Krizhevsky et al., 2012) is used.

The input to this deep ConvNet is the $focusCube$, which does not always have the same size because

Deep ConvNet Configurations	
Input Size: 32 **by** 32	**Input Size:** 48 **by** 48
Spatial Conv 128: size 3 × 3, stride 1, pad 1	
ReLU	
Max Pooling: size 2 × 2, stride 2	
Spatial Conv 164: size 3 × 3, stride 1, pad 1	
ReLU	
Max Pooling: size 2 × 2, stride 2	
Spatial Conv 192: size 3 × 3, stride 1, pad 1	
ReLU	
Max Pooling: size 2 × 2, stride 2	
Spatial Conv 192: size 3 × 3, stride 1, pad 1	
ReLU	
Max Pooling: size 2 × 2, stride 2	
Spatial Conv 128: size 3 × 3, stride 1, pad 1	
ReLU	
Max Pooling: 2 × 2, s2	Max Pooling: 3 × 3, s1
Fully-Connected Layer	
ReLU	
Fully-Connected Layer	
LogSoftMax	

Table 1: Deep ConvNet architecture given two padding size configurations for final classification.

the lengths of input sentences vary. To address this, we use zero padding. For computational reasons we provide two configurations in Table 1, for length padding up to either 32×32 or 48×48. The only difference between the two configurations is the last pooling layer. If sentences are longer than the padding length limit we only use the number of words up to the limit. In our experiments we found the 48×48 padding limit to be acceptable since most sentences in our datasets are only $1-30$ words long.

8 Experimental Setup

Datasets. We conducted five separate experiments on ten different datasets: three recent SemEval competitions and two answer selection tasks. Note that the answer selection task, which is to rank candidate answer sentences based on their similarity to the questions, is essentially the similarity measurement problem. The five experiments are as follows:

1. Sentences Involving Compositional Knowledge (SICK) is from Task 1 of the 2014 SemEval competition (Marelli et al., 2014) and consists of 9,927 annotated sentence pairs, with 4,500 for training, 500 as a development set, and 4,927 for testing. Each pair has a relatedness score $\in [1, 5]$ which increases with similarity.
2. Microsoft Video Paraphrase Corpus (MSRVID) is from Task 6 of the 2012 SemEval competition (Agirre et al., 2012) and consists of 1,500 annotated pairs of video descriptions, with half for training and the other half for testing. Each sentence pair has a relatedness score $\in [0, 5]$ which increases with similarity.
3. Task 10 of the 2014 SemEval competition on Semantic Textual Similarity (STS2014) (Agirre et al., 2014) provided six different test sets from different domains. Each pair has a similarity score $\in [0, 5]$ which increases with similarity. Following the competition rules, our training data is only drawn from previous STS competitions in 2012 and 2013. We excluded training sentences with lengths longer than the 48 word padding limit, resulting in 7,382 training pairs out of a total of 7,592. Table 2 provides a brief description of the test sets.
4. The open domain question-answering WikiQA data is from Bing query logs by Yang et al. (2015). We followed the same pre-processing steps as Yang et al. (2015), where questions with no correct candidate answer sentences are excluded and answer sentences are truncated to 40 tokens. The resulting dataset consists of 873 questions with 8,672 question-answer pairs in the training set, 126 questions with 1,130 pairs in the development set, and 243 questions with 2,351 pairs in the test set.
5. The TrecQA dataset (Wang et al., 2007) from the Text Retrieval Conferences has been widely used for the answer selection task during the past decade. To enable direct comparison with previous work, we used the same training, development, and test sets as released by Yao et al. (2013).

STS2014	Domain	Pairs
deft-forum	discussion forums	450
deft-news	news articles	300
headlines	news headlines	750
images	image descriptions	750
OnWN	word sense definitions	750
tweet-news	social media	750
Total		**3,750**

Table 2: Description of STS2014 test sets.

The TrecQA data consists of 1,229 questions with 53,417 question-answer pairs in the *TRAIN-ALL* training set, 82 questions with 1,148 pairs in the development set, and 100 questions with 1,517 pairs in the test set.

Training. For experiments on SICK, MSRVID, and STS2014, the training objective is to minimize the KL-divergence loss:

$$loss(\theta) = \frac{1}{n}\sum_{k=1}^{n} KL\left(f^k \mid\mid \widehat{f_\theta^k}\right) \qquad (11)$$

where f is the ground truth, $\widehat{f_\theta}$ is the predicted distribution with model weights θ, and n is the number of training examples.

We used a hinge loss for the answer selection task on WikiQA and TrecQA data. The training objective is to minimize the following loss, summed over examples $\langle x, y_{gold}\rangle$:

$$loss(\theta, x, y_{gold}) = \sum_{y' \neq y_{gold}} \max(0, 1 + f_\theta(x, y') - f_\theta(x, y_{gold})) \qquad (12)$$

where y_{gold} is the ground truth label, input x is the pair of sentences $x = \{S_1, S_2\}$, θ is the model weight vector, and the function $f_\theta(x, y')$ is the output of our model.

In all cases, we performed optimization using RMSProp (Tieleman and Hinton, 2012) with back-propagation (Bottou, 1998), with a learning rate fixed to 10^{-4}.

Settings. For the SICK and MSRVID experiments, we used 300-dimension GloVe word embeddings (Pennington et al., 2014). For the STS2014, WikiQA, and TrecQA experiments, we used 300-dimension PARAGRAM-SL999 embeddings from Wieting et al. (2015) and the PARAGRAM-PHRASE embeddings from Wieting et al. (2016), trained on word pairs from the Paraphrase Database (PPDB) (Ganitkevitch et al., 2013). We did not update word embeddings in all experiments.

We used the SICK development set for tuning and then applied exactly the *same* hyperparameters to *all* ten test sets. For the answer selection task (WikiQA and TrecQA), we used the official trec_eval scorer to compute the metrics Mean Average Precision (MAP) and Mean Reciprocal Rank (MRR) and selected the best development model based on MRR for final testing. Our timing experiments were conducted on an Intel Xeon E5-2680 CPU.

Model	r	ρ	**MSE**
Socher et al. (2014) DTRNN	0.7863	0.7305	0.3983
Socher et al. (2014) SDTRNN	0.7886	0.7280	0.3859
Lai and Hockenmaier (2014)	0.7993	0.7538	0.3692
Jimenez et al. (2014)	0.8070	0.7489	0.3550
Bjerva et al. (2014)	0.8268	0.7721	0.3224
Zhao et al. (2014)	0.8414	-	-
LSTM	0.8477	0.7921	0.2949
Bi-LSTM	0.8522	0.7952	0.2850
2-layer LSTM	0.8411	0.7849	0.2980
2-layer Bi-LSTM	0.8488	0.7926	0.2893
Tai et al. (2015) Const. LSTM	0.8491	0.7873	0.2852
Tai et al. (2015) Dep. LSTM	0.8676	0.8083	0.2532
He et al. (2015)	0.8686	0.8047	0.2606
This work	**0.8784**	**0.8199**	**0.2329**

Table 3: Test results on SICK grouped as: (1) RNN variants; (2) SemEval 2014 systems; (3) Sequential LSTM variants; (4) Dependency and constituency tree LSTMs. Evaluation metrics are Pearson's r, Spearman's ρ, and mean squared error (MSE). Rows in grey are neural network models.

Due to sentence length variations, for the SICK and MSRVID data we padded the sentences to 32 words; for the STS2014, WikiQA, and TrecQA data, we padded the sentences to 48 words.

9 Results

SICK Results (Table 3). Our model outperforms previous neural network models, most of which are based on sentence modeling. The ConvNet work (He et al., 2015) and TreeLSTM work (Tai et al., 2015) achieve comparable accuracy; for example, their difference in Pearson's r is only 0.1%. In comparison, our model outperforms both by 1% in Pearson's r, over 1.1% in Spearman's ρ, and 2-3% in MSE. Note that we used the same word embeddings, sparse distribution targets, and loss function as in He et al. (2015) and Tai et al. (2015), thereby representing comparable experimental conditions.

MSRVID Results (Table 4). Our model outperforms the work of He et al. (2015), which already reports a Pearson's r score of over 0.9,

STS2014 Results (Table 5). Systems in the competition are ranked by the weighted mean (the of-

Model	Pearson's r
Beltagy et al. (2014)	0.8300
Bär et al. (2012)	0.8730
Šarić et al. (2012)	0.8803
He et al. (2015)	0.9090
This work	**0.9112**

Table 4: Test results on MSRVID data.

STS2014	3rd	2nd	1st	This work
deft-forum	0.5305	0.4711	0.4828	**0.5684**
deft-news	**0.7813**	0.7628	0.7657	0.7079
headlines	**0.7837**	0.7597	0.7646	0.7551
image	**0.8343**	0.8013	0.8214	0.8221
OnWN	0.8502	0.8745	0.8589	**0.8847**
tweetnews	0.6755	**0.7793**	0.7639	0.7469
Wt. Mean	0.7549	0.7605	0.7610	**0.7666**

Table 5: Test results on all six test sets in STS2014. We show results of the top three participating systems at the competition in Pearson's r scores.

ficial measure) of Pearson's r scores calculated based on the number of sentence pairs in each test set. We show the 1st ranked (Sultan et al., 2014), 2nd (Kashyap et al., 2014), 3rd (Lynum et al., 2014) systems in the STS2014 competition, all of which are based on heavy feature engineering. Our model does not use any sparse features, WordNet, or parse trees, but still performs favorably compared to the STS2014 winning system (Sultan et al., 2014).

WikiQA Results (Table 6). We compared our model to competitive baselines prepared by Yang et al. (2015) and also evaluated He et al. (2015)'s multi-perspective ConvNet on the same data. The neural network models in the table, paragraph vector (PV) (Le and Mikolov, 2014), CNN (Yu et al., 2014), and PV-Cnt/CNN-Cnt with word matching features (Yang et al., 2015), are mostly based on sentence modeling. Our model outperforms them all.

TrecQA Results (Table 7). This is the largest dataset in our experiments, with over 55,000 question-answer pairs. Only recently have neural network approaches (Yu et al., 2014) started to show promising results on this decade-old dataset. Previous approaches with probabilistic tree-edit techniques or tree kernels (Wang and Manning, 2010; Heilman and Smith, 2010; Yao et al., 2013) have been successful since tree structure information per-

Model	MAP	MRR
Word Cnt (Yang et al., 2015)	0.4891	0.4924
Wgt Word Cnt (Yang et al., 2015)	0.5099	0.5132
PV (Le and Mikolov, 2014)	0.5110	0.5160
PV-Cnt (Yang et al., 2015)	0.5976	0.6058
LCLR (Yih et al., 2013)	0.5993	0.6086
CNN (Yu et al., 2014)	0.6190	0.6281
CNN-Cnt (Yang et al., 2015)	0.6520	0.6652
He et al. (2015)	0.6930	0.7090
This work	**0.7090**	**0.7234**

Table 6: Test results on WikiQA data.

Model	MAP	MRR
Cui et al. (2005)	0.4271	0.5259
Wang et al. (2007)	0.6029	0.6852
Heilman and Smith (2010)	0.6091	0.6917
Wang and Manning (2010)	0.5951	0.6951
Yao et al. (2013)	0.6307	0.7477
Severyn and Moschitti (2013)	0.6781	0.7358
Yih et al. (2013)	0.7092	0.7700
Wang and Nyberg (2015)	0.7134	0.7913
Severyn and Moschitti (2015)	0.7459	0.8078
This work	**0.7588**	**0.8219**

Table 7: Test results on TrecQA data.

mits a fine-grained focus on important words for similarity comparison purposes. Our approach essentially follows this intuition, but in a neural network setting with the use of our similarity focus layer. Our model outperforms previous work.

10 Analysis

Ablation Studies. Table 8 shows the results of ablation studies on SICK and WikiQA data. We removed or replaced one component at a time from the full system and performed re-training and re-testing. We found large drops when removing the context modeling component, indicating that the context information provided by the Bi-LSTMs is crucial for the following components (e.g., interaction modeling). The use of our similarity focus layer is also beneficial, especially on the WikiQA data. When we replaced the entire similarity focus layer with a random dropout layer ($p = 0.3$), the dropout layer hurts accuracy; this shows the importance of directing the model to focus on important pairwise word interactions, to better capture similarity.

Model Efficiency and Storage. He et al. (2015)'s

Ablation on SICK Data	**Pearson**
Full Model	0.8784
- Remove context modeling (Sec. 4)	-0.1225
- Remove entire focus layer (Sec. 6)	-0.0083
- Replace entire focus layer with dropout	-0.0314
Ablation on WikiQA Data	**MRR**
Full Model	0.7234
- Remove context modeling (Sec. 4)	-0.0990
- Remove entire focus layer (Sec. 6)	-0.0327
- Replace entire focus layer with dropout	-0.0403

Table 8: Ablation studies on SICK and WikiQA data, removing each component separately.

Model	**# of Parameters**	**Timing (s)**
He et al. (2015)	10.0 million	2265
This work	**1.7 million**	**664**

Table 9: Comparison of training efficiency and number of tunable model parameters on SICK data. Timing is the average epoch time in seconds for training on a single CPU thread.

ConvNet model uses multiple types of convolution and pooling for sentence modeling. This results in a wide architecture with around 10 million tunable parameters. Our approach only models pairwise word interactions and does not require such a complicated architecture. Compared to that previous work, Table 9 shows that our model is 3.4× faster in training and has 83% fewer tunable parameters.

Visualization. Table 10 visualizes the cosine value channel of the $focusCube$ for pairwise word interactions given two sentence pairs in the SICK test set. Note for easier visualization, the values are multiplied by 10. Darker red areas indicate stronger pairwise word interactions. From these visualizations, we see that our model is able to identify important word pairs (in dark red) and tag them with proper similarity values, which are significantly higher than the ones of their neighboring unimportant pairs. This shows that our model is able to recognize important fine-grained word-level information for better similarity measurement, suggesting the reason why our model performs well.

11 Conclusion

In summary, we developed a novel neural network model based on a hybrid of ConvNet and Bi-

	A	man	is	playing	the	drum
A	8.99	0.69	0.43	0.32	0.38	0.22
man	0.70	9.93	0.62	0.45	0.46	0.38
is	0.64	0.80	8.50	0.62	0.58	0.36
practicing	0.46	0.67	0.66	6.51	0.62	0.48
the	0.35	0.56	0.66	0.64	7.85	0.52
drum	0.27	0.47	0.46	0.55	0.64	8.82
	A	man	is	carrying	a	tree
A	0.53	0.33	0.32	0.33	5.53	0.49
tree	0.32	0.30	0.19	0.20	0.38	8.73
is	0.35	0.31	0.21	0.06	0.03	0.40
being	0.28	0.37	2.60	0.18	0.13	0.38
picked	0.15	0.18	0.10	1.60	0.07	0.27
up	0.26	0.27	0.06	0.13	0.05	0.21
by	0.43	0.36	0.08	1.33	0.15	0.29
a	6.50	0.45	0.03	0.08	0.16	0.23
man	0.50	8.60	0.45	0.34	0.34	0.34

Table 10: Visualization of cosine values (multiplied by 10) in the $focusCube$ given two sentence pairs in the SICK test set.

LSTMs for the semantic textual similarity measurement problem. Our pairwise word interaction model and the similarity focus layer can better capture fine-grained semantic information, compared to previous sentence modeling approaches that attempt to "cram" all sentence information into a fixed-length vector. We demonstrated the state-of-the-art accuracy of our approach on data from three SemEval competitions and two answer selection tasks.

Acknowledgments

This work was supported by the U.S. National Science Foundation under awards IIS-1218043 and CNS-1405688. Any opinions, findings, conclusions, or recommendations expressed are those of the authors and do not necessarily reflect the views of the sponsor. We would like to thank Kevin Gimpel, John Wieting and TS for all the support.

References

Eneko Agirre, Mona Diab, Daniel Cer, and Aitor Gonzalez-Agirre. 2012. SemEval-2012 task 6: A pilot on semantic textual similarity. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics*, pages 385–393.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe.

2014. SemEval-2014 task 10: Multilingual semantic textual similarity. In *Proceedings of the 8th International Workshop on Semantic Evaluation*, pages 81–91.

Yaser Al-onaizan, Jan Curin, Michael Jahr, Kevin Knight, John Lafferty, Dan Melamed, Franz-Josef Och, David Purdy, Noah A. Smith, and David Yarowsky. 1999. Statistical machine translation. Final report, JHU Summer Workshop on Language Engineering.

Galen Andrew, Raman Arora, Jeff Bilmes, and Karen Livescu. 2013. Deep canonical correlation analysis. In *Proceedings of the 30th International Conference on Machine Learning*, pages 1247–1255.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

Daniel Bär, Chris Biemann, Iryna Gurevych, and Torsten Zesch. 2012. UKP: Computing semantic textual similarity by combining multiple content similarity measures. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics*, pages 435–440.

Islam Beltagy, Katrin Erk, and Raymond Mooney. 2014. Probabilistic soft logic for semantic textual similarity. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics*, pages 1210–1219.

Johannes Bjerva, Johan Bos, Rob van der Goot, and Malvina Nissim. 2014. The meaning factory: Formal semantics for recognizing textual entailment and determining semantic similarity. In *Proceedings of the 8th International Workshop on Semantic Evaluation*, pages 642–646.

Léon Bottou. 1998. Online learning and stochastic approximations. In David Saad, editor, *Online Learning and Neural Networks*. Cambridge University Press.

Jane Bromley, James W Bentz, Léon Bottou, Isabelle Guyon, Yann LeCun, Cliff Moore, Eduard Säckinger, and Roopak Shah. 1993. Signature verification using a "Siamese" time delay neural network. *International Journal of Pattern Recognition and Artificial Intelligence*, 7(4):669–688.

Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. 2005. Learning to rank using gradient descent. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 89–96.

Hang Cui, Renxu Sun, Keya Li, Min-Yen Kan, and Tat-Seng Chua. 2005. Question answering passage retrieval using dependency relations. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 400–407.

Dipanjan Das and Noah A. Smith. 2009. Paraphrase identification as probabilistic quasi-synchronous recognition. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics*, pages 468–476.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

Samuel Fern and Mark Stevenson. 2008. A semantic similarity approach to paraphrase detection. In *Proceedings of the 11th Annual Research Colloquium of the UK Special-Interest Group for Computational Lingusitics*, pages 45–52.

Katja Filippova, Enrique Alfonseca, Carlos A. Colmenares, Lukasz Kaiser, and Oriol Vinyals. 2015. Sentence compression by deletion with LSTMs. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 360–368.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The Paraphrase Database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 758–764.

Alex Graves, Santiago Fernández, and Jürgen Schmidhuber. 2005. Bidirectional LSTM networks for improved phoneme classification and recognition. In *Proceedings of the 15th International Conference on Artificial Neural Networks: Formal Models and Their Applications - Part II*, pages 799–804.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 369–376.

Hua He, Kevin Gimpel, and Jimmy Lin. 2015. Multi-perspective sentence similarity modeling with convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1576–1586.

Michael Heilman and Noah A. Smith. 2010. Tree edit models for recognizing textual entailments, paraphrases, and answers to questions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 1011–1019.

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

Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. In *Advances* *in Neural Information Processing Systems 27*, pages 2042–2050.

Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management*, pages 2333–2338.

Sergio Jimenez, George Duenas, Julia Baquero, Alexander Gelbukh, Av Juan Dios Bátiz, and Av Mendizábal. 2014. UNAL-NLP: Combining soft cardinality features for semantic textual similarity, relatedness and entailment. In *Proceedings of the 8th International Workshop on Semantic Evaluation*, pages 732–742.

Abhay Kashyap, Lushan Han, Roberto Yus, Jennifer Sleeman, Taneeya Satyapanich, Sunil Gandhi, and Tim Finin. 2014. Meerkat mafia: Multilingual and cross-level semantic textual similarity systems. In *Proceedings of the 8th International Workshop on Semantic Evaluation*, pages 416–423.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pages 1097–1105.

Alice Lai and Julia Hockenmaier. 2014. Illinois-LH: A denotational and distributional approach to semantics. In *Proceedings of the 8th International Workshop on Semantic Evaluation*, pages 329–334.

Quoc V. Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the 31th International Conference on Machine Learning*, pages 1188–1196.

Jimmy Lin. 2007. An exploration of the principles underlying redundancy-based factoid question answering. *ACM Transactions on Information Systems*, 25(2):1–55.

André Lynum, Partha Pakray, Björn Gambäck, and Sergio Jimenez. 2014. NTNU: Measuring semantic similarity with sublexical feature representations and soft cardinality. In *Proceedings of the 8th International Workshop on Semantic Evaluation*, pages 448–453.

Nitin Madnani, Joel Tetreault, and Martin Chodorow. 2012. Re-examining machine translation metrics for paraphrase identification. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 182–190.

Marco Marelli, Luisa Bentivogli, Marco Baroni, Raffaella Bernardi, Stefano Menini, and Roberto Zamparelli. 2014. 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*, pages 1–8.

Raymond J. Mooney. 2014. Semantic parsing: Past, present, and future. In *ACL Workshop on Semantic Parsing. Presentation slides.*

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. *CoRR*, abs/1509.00685.

Frane Šarić, Goran Glavaš, Mladen Karan, Jan Šnajder, and Bojana Dalbelo Bašić. 2012. TakeLab: systems for measuring semantic text similarity. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics*, pages 441–448.

Aliaksei Severyn and Alessandro Moschitti. 2013. Automatic feature engineering for answer selection and extraction. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 458–467.

Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to rank short text pairs with convolutional deep neural networks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 373–382.

Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556.

Richard Socher, Eric H. Huang, Jeffrey Pennington, Andrew Y. Ng, and Christopher D. Manning. 2011. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Advances in Neural Information Processing Systems 24*, pages 801–809.

Richard Socher, Andrej Karpathy, Quoc V. Le, Christopher D. Manning, and Andrew Y. Ng. 2014. Grounded compositional semantics for finding and describing images with sentences. *Transactions of the Association for Computational Linguistics*, 2:207–218.

Md Arafat Sultan, Steven Bethard, and Tamara Sumner. 2014. Dls@cu: Sentence similarity from word alignment. In *Proceedings of the 8th International Workshop on Semantic Evaluation*, pages 241–246.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27*, pages 3104–3112.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich.

2015. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9.

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

Tijmen Tieleman and Geoffrey E. Hinton. 2012. Lecture 6.5—RMSProp: Divide the gradient by a running average of its recent magnitude. Coursera: Neural Networks for Machine Learning.

Stephen Wan, Mark Dras, Robert Dale, and Cecile Paris. 2006. Using Dependency-based Features to Take the "Para-farce" out of Paraphrase. In *Australasian Language Technology Workshop*, pages 131–138.

Mengqiu Wang and Christopher D. Manning. 2010. Probabilistic tree-edit models with structured latent variables for textual entailment and question answering. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 1164–1172.

Di Wang and Eric Nyberg. 2015. A long short-term memory model for answer sentence selection in question answering. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*, pages 707–712.

Mengqiu Wang, Noah A. Smith, and Teruko Mitamura. 2007. What is the Jeopardy model? A quasi-synchronous grammar for QA. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 22–32.

Jason Weston, Samy Bengio, and Nicolas Usunier. 2011. Wsabie: Scaling up to large vocabulary image annotation. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, pages 2764–2770.

John Wieting, Mohit Bansal, Kevin Gimpel, Karen Livescu, and Dan Roth. 2015. From paraphrase database to compositional paraphrase model and back. *Transactions of the Association for Computational Linguistics*, 3:345–358.

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

Ronald J. Williams and David Zipser. 1989. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1(2):270–280.

Wei Xu, Alan Ritter, Chris Callison-Burch, William B. Dolan, and Yangfeng Ji. 2014. Extracting lexically divergent paraphrases from Twitter. *Transactions of the Association for Computational Linguistics*, 2:435–448.

Wei Xu. 2014. *Data-Drive Approaches for Paraphrasing Across Language Variations*. Ph.D. thesis, Department of Computer Science, New York University.

Yi Yang, Wen-tau Yih, and Christopher Meek. 2015. WikiQA: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2013–2018.

Xuchen Yao, Benjamin Van Durme, Chris Callison-burch, and Peter Clark. 2013. Answer extraction as sequence tagging with tree edit distance. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 858–867.

Wentau Yih, Ming-Wei Chang, Christopher Meek, and Andrzej Pastusiak. 2013. Question answering using enhanced lexical semantic models. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 1744–1753.

Wenpeng Yin and Hinrich Schütze. 2015. Convolutional neural network for paraphrase identification. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 901–911.

Lei Yu, Karl Moritz Hermann, Phil Blunsom, and Stephen Pulman. 2014. Deep Learning for Answer Sentence Selection. In *NIPS Deep Learning Workshop*.

Guido Zarrella, John Henderson, Elizabeth M. Merkhofer, and Laura Strickhart. 2015. MITRE: Seven systems for semantic similarity in tweets. In *Proceedings of the 9th International Workshop on Semantic Evaluation*, pages 12–17.

Jiang Zhao, Tian Tian Zhu, and Man Lan. 2014. ECNU: One stone two birds: Ensemble of heterogenous measures for semantic relatedness and textual entailment. In *Proceedings of the 8th International Workshop on Semantic Evaluation*, pages 271–277.

Xiaodan Zhu, Parinaz Sobhani, and Hongyu Guo. 2015. Long short-term memory over recursive structures. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 1604–1612.