UTA_DLNLP at SemEval-2016 Task 1: Semantic Textual Similarity: A Unified Framework for Semantic Processing and Evaluation

Peng Li
Computer Science and Engineering
University of Texas at Arlington
jerryli1981@gmail.com

Heng Huang*
Computer Science and Engineering
University of Texas at Arlington
heng@uta.edu

Abstract

In this paper, we propose a deep neural network based natural language processing system for semantic textual similarity prediction. We leverage multi-layer bidirectional LSTM to learn sentence representation. After that, we construct matching features followed by Highway Multilayer Perceptron to make predictions. Experimental results demonstrate that this approach can't get better results on standard evaluation datasets.

1 Introduction

Traditional approaches (Lai and Hockenmaier, 2014; Zhao et al., 2014; Jimenez et al., 2014) for semantic textual similarity prediction usually build the supervised model using a variety of hand crafted features. Hundreds of features generated at different linguistic levels are exploited to boost classification. With the success of deep learning in many machine learning related applications, there has been much interest in applying deep neural network based techniques to further improve the prediction tasks in natural language processing (NLP) (Socher et al., 2011b; Iyyer et al., 2014; Tai et al., 2015).

A key component of deep neural network is word embeddings which serves as a lookup table to get word representations. From low-level NLP tasks such as language modeling, POS tagging, name entity recognition, and semantic role labeling, to high-level tasks such as machine translation, information retrieval and semantic analysis (Kalchbrenner and Blunsom, 2013; Socher et al., 2011a; Tai et al., 2015). Deep word representation learning has demonstrated its importance for these tasks. All the tasks get performance improvement via further learning either word level representations or sentence level representations.

In this work, we focus on deep neural network based semantic textual similarity prediction. We use multi-layer bidirectional LSTM (Long Short Term Memory) (Graves et al., 2013) to learn sentence representations. After that, we construct matching features followed by Highway Multilayer Perceptron to learn high-level hidden matching feature representations. Below, we will briefly introduce Multi-Layer Bidirectional LSTM.

2 Multi-Layer Bidirectional LSTM

2.1 RNN vs LSTM

Recurrent neural networks (RNNs) are capable of modeling sequences of varying lengths via the recursive application of a transition function on a hidden state. For example, at each time step t, an RNN takes the input vector $\mathbf{x}_t \in \mathbb{R}^n$ and the hidden state vector $\mathbf{h}_{t-1} \in \mathbb{R}^m$, then applies affine transformation followed by an element-wise nonlinearity such as hyperbolic tangent function to produce the next hidden state vector $\mathbf{h}_t$:

$$\mathbf{h}_t = \tanh(\mathbf{W}\mathbf{x}_t + \mathbf{U}\mathbf{h}_{t-1} + \mathbf{b})\,. \tag{1}$$

A major issue of RNNs using these transition functions is that it is difficult to learn long-range de-

*To whom all correspondence should be addressed. This work was partially supported by NSF-IIS 1117965, NSF-IIS 1302675, NSF-IIS 1344152, NSF-DBI 1356628, NIH R01 AG049371.

Proceedings of SemEval-2016, pages 596–599,
San Diego, California, June 16-17, 2016.

pendencies during training step because the components of the gradient vector can grow or decay exponentially (Bengio et al., 1994).

The LSTM architecture (Hochreiter and Schmidhuber, 1998) addresses the problem of learning long range dependencies by introducing a memory cell that is able to preserve state over long periods of time. Concretely, at each time step t, the LSTM *unit* can be defined as a collection of vectors in $\mathbb{R}^d$: an *input gate* $\mathbf{i}_t$, a *forget gate* $\mathbf{f}_t$, an *output gate* $\mathbf{o}_t$, a *memory cell* $\mathbf{c}_t$ and a hidden state $\mathbf{h}_t$. We refer to d as the *memory dimensionality* of the LSTM. One step of an LSTM takes as input $\mathbf{x}_t$, $\mathbf{h}_{t-1}$, $\mathbf{c}_{t-1}$ and produces $\mathbf{h}_t$, $\mathbf{c}_t$ via the following transition equations:

$$\begin{aligned}
\mathbf{i}_t &= \sigma(\mathbf{W}^{(\mathbf{i})}\mathbf{x}_t + \mathbf{U}^{(\mathbf{i})}\mathbf{h}_{t-1} + \mathbf{b}^{(\mathbf{i})}) \,, \\
\mathbf{f}_t &= \sigma(\mathbf{W}^{(\mathbf{f})}\mathbf{x}_t + \mathbf{U}^{(\mathbf{f})}\mathbf{h}_{t-1} + \mathbf{b}^{(\mathbf{f})}) \,, \\
\mathbf{o}_t &= \sigma(\mathbf{W}^{(\mathbf{o})}\mathbf{x}_t + \mathbf{U}^{(\mathbf{o})}\mathbf{h}_{t-1} + \mathbf{b}^{(\mathbf{o})}) \,, \\
\mathbf{u}_t &= \tanh(\mathbf{W}^{(\mathbf{u})}\mathbf{x}_t + \mathbf{U}^{(\mathbf{u})}\mathbf{h}_{t-1} + \mathbf{b}^{(\mathbf{u})}) \,, \\
\mathbf{c}_t &= \mathbf{i}_t \odot \mathbf{u}_t + \mathbf{f}_t \odot \mathbf{c}_{t-1} \,, \\
\mathbf{h}_t &= \mathbf{o}_t \odot \tanh(\mathbf{c}_t) \,,
\end{aligned} \tag{2}$$

where $\sigma(\cdot)$ and $\tanh(\cdot)$ are the element-wise sigmoid and hyperbolic tangent functions, $\odot$ is the element-wise multiplication operator.

2.2 Model Description

One shortcoming of conventional RNNs is that they are only able to make use of previous context. In semantic text similarity prediction task, the decision is made after the whole sentence pair is digested. Therefore, exploring future context would be better for sequence meaning representation. Bidirectional RNNs architecture (Graves et al., 2013) proposed a solution of making prediction based on future words. At each time step t, the model maintains two hidden states, one for the left-to-right propagation $\overrightarrow{\mathbf{h}_t}$ and the other for the right-to-left propagation $\overleftarrow{\mathbf{h}}_t$. The hidden state of the Bidirectional LSTM is the concatenation of the forward and backward hidden states. The following equations illustrate the main ideas:

$$\begin{aligned}
\overrightarrow{\mathbf{h}_t} &= \tanh(\overrightarrow{\mathbf{W}}\mathbf{x}_t + \overrightarrow{\mathbf{U}}\overrightarrow{\mathbf{h}}_{t-1} + \overrightarrow{\mathbf{b}}) \\
\overleftarrow{\mathbf{h}_t} &= \tanh(\overleftarrow{\mathbf{W}}\mathbf{x}_t + \overleftarrow{\mathbf{U}}\overleftarrow{\mathbf{h}}_{t+1} + \overleftarrow{\mathbf{b}}) \,,
\end{aligned} \tag{3}$$

Deep RNNs can be created by stacking multiple RNN hidden layer on top of each other, with the output sequence of one layer forming the input sequence for the next. Assuming the same hidden layer function is used for all N layers in the stack, the hidden vectors $\mathbf{h}^n$ are iteratively computed from $n = 1$ to N and $t = 1$ to T:

$$\mathbf{h}_t^n = \tanh(\mathbf{W}\mathbf{h}_t^{n-1} + \mathbf{U}\mathbf{h}_{t-1}^n + \mathbf{b}) \,. \tag{4}$$

Multilayer bidirectional RNNs can be implemented by replacing each hidden vector $\mathbf{h}^n$ with the forward and backward vectors $\overrightarrow{\mathbf{h}^n}$ and $\overleftarrow{\mathbf{h}}^n$, and ensuring that every hidden layer receives input from both the forward and backward layers at the level below. Furthermore, we can apply LSTM memory cell to hidden layers to construct multilayer bidirectional LSTM.

Finally, we can concatenate sequence hidden matrix $\overrightarrow{\mathbf{M}} \in \mathbb{R}^{n \times d}$ and reversed sequence hidden matrix $\overleftarrow{\mathbf{M}} \in \mathbb{R}^{n \times d}$ to form the sentence representation. Here n is the number of layers, d is the *memory dimensionality* of the LSTM. In the next section, we will use the two matrices to generate matching feature planes via linear algebra operations.

3 Learning from Matching Features

Inspired by (Tai et al., 2015), we apply element-wise merge to first sentence matrix $M_1 \in \mathbb{R}^{n \times 2d}$ and second sentence matrix $M_2 \in \mathbb{R}^{n \times 2d}$. Similar to previous method, we can define two simple matching feature planes (*FPs*) with below equations:

$$\begin{aligned}
FP_1 &= M_1 \odot M_2 \,, \\
FP_2 &= |M_1 - M_2| \,,
\end{aligned} \tag{5}$$

where $\odot$ is the element-wise multiplication. The FP_1 measure can be interpreted as an element-wise comparison of the signs of the input representations. The FP_2 measure can be interpreted as the distance between the input representations.

3.1 Highway MLP

Inspired by (Kim et al., 2016), we build Highway Multilayer Perceptron (HMLP) layer to further enhance representation learning. Conventional MLP applies an affine transformation followed by a nonlinearity to obtain a new set of features:

$$\mathbf{z} = g(\mathbf{W}\mathbf{y} + \mathbf{b}) \,. \tag{6}$$

One layer of a highway network does the following:

$$\mathbf{z} = \mathbf{t} \odot g(\mathbf{W}_H \mathbf{y} + \mathbf{b}_H) + (1 - \mathbf{t}) \odot \mathbf{y}\,, \quad (7)$$

where g is a nonlinearity, $\mathbf{t} = \sigma(\mathbf{W}_T \mathbf{y} + \mathbf{b}_T)$ is called as the transform gate, and $(1 - \mathbf{t})$ is called as the carry gate. Similar to the memory cells in LSTM networks, highway layers allow adaptively carrying some dimensions of the input directly to the input for training deep networks.

4 Experiments

We use all previous dataset to train our LSTM classifier. The total number of training examples is 12912, and the number of dev examples is 680. Note that we didn't use cross validation to find the best model. Table 1 shows the Pearson correlation results of STS task.

4.1 Hyperparameters and Training Details

We first initialize our word representations using publicly available 300-dimensional Glove word vectors [1]. LSTM memory dimension is 100, the number of layers is 2. Training is done through stochastic gradient descent over shuffled mini-batches with the AdaGrad update rule (Duchi et al., 2011). The learning rate is set to 0.05. The mini-batch size is 25. The model parameters were regularized with a perminibatch L2 regularization strength of 10^{-4}. Note that word embeddings were fixed during training.

4.2 Objective Functions

The task of semantic similarity prediction tries to measure the degree of semantic similarity of a sentence pair by assigning a similarity score ranging from 1 (completely unrelated) to 5 (semantically equivalent). Inspired by (Tai et al., 2015), given a sentence pair, we wish to predict a real-valued similarity score in a range of $[1, K]$, where $K > 1$ is an integer. The sequence $1, 2, ..., K$ is the ordinal scale of similarity, where higher scores indicate greater degrees of similarity. We can predict the similarity score $\hat{y}$ by predicting the probability that the learned hidden representation x_h belongs to the ordinal scale. This is done by projecting an input representation onto a set of hyperplanes, each of which corresponds to a class. The distance from the input to a hyperplane reflects the probability that the input will located in corresponding scale.

Mathematically, the similarity score $\hat{y}$ can be written as:

$$\begin{aligned} \hat{y} &= r^T \cdot \hat{p}_\theta(y|x_h) \\ &= r^T \cdot softmax(W \cdot x_h + b) \\ &= r^T \cdot \frac{e^{W_i x_h + b_i}}{\sum_j e^{W_j x_h + b_j}} \end{aligned} \quad (8)$$

where $r^T = [1\ 2 \ldots K]$ and the weight matrix W and b are parameters.

In order to introduce the task objective function, we define a sparse target distribution p that satisfies $y = r^T p$:

$$p_i = \begin{cases} y - \lfloor y \rfloor, & i = \lfloor y \rfloor + 1 \\ \lfloor y \rfloor - y + 1, & i = \lfloor y \rfloor \\ 0 & otherwise \end{cases} \quad (9)$$

where $1 \leq i \leq K$. The objective function then can be defined as the regularized KL-divergence between p and p_θ:

$$J(\theta) = -\frac{1}{m} \sum_{k=1}^{m} KL(p^{(k)} || p_\theta^k) + \frac{\lambda}{2} ||\theta||_2^2\,, \quad (10)$$

where m is the number of training pairs and the superscript k indicates the k-th sentence pair (Tai et al., 2015).

5 Conclusions and Discussions

In this paper, we propose a deep neural network architecture that leverages pre-trained word embeddings to learn sentence meanings. Our approach first generates word sequence representations as inputs into a multilayer bidirectional LSTM to learn sentence representations. After that, we construct matching features followed by highway MLP to learn high-level hidden matching feature representations. Experimental results on benchmark datasets demonstrate that our model didn't achieved the state-of-the-art performance compared with other approaches. Our approach is above the median scores only on question-question domain. We suspect our model have worse capability of domain adaption. Also the Highway MLP may increase the model complexity and lead to worse performance.

[1] `http://nlp.stanford.edu/projects/glove/`

Method	All	answer-answer	headlines	plagiarism	postediting	question-question
Our Run: 100-1	0.64965	0.46391	0.74499	0.74003	0.71947	0.58083
Our Run: 150-1	0.64500	0.43042	0.72133	0.71620	0.74471	0.62006
Our Run: 150-3	0.63698	0.41871	0.72485	0.70296	0.69652	0.65543
Median	0.68923	0.48018	0.76439	0.78949	0.81241	0.57140
Best	0.77807	0.69235	0.82749	0.84138	0.86690	0.74705

Table 1: The pearson correlation score comparison on STS Task, Here 100 and 150 are LSTM memory dimension. 1 and 3 are the number of LSTM layers

References

Yoshua Bengio, Patrice Simard, and Paolo Fransconi. 1994. Learning long-term dependencies with gradient descent is difficult. In *IEEE Transactions on Neural Networks 5(2)*.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 12:2121–2159.

Alex Graves, Navdeep Jaitly, and Abdel rahman Mohamed. 2013. Hybrid speech recognition with deep bidirectional lstm. In *IEEE Workshop on Au- tomatic Speech Recognition and Understanding (ASRU)*, pages 273–278.

Sepp Hochreiter and Jürgen Schmidhuber. 1998. Long short-term memory. In *Neural Computation 9(8)*.

Mohit Iyyer, Jordan Boyd-Graber, Leonardo Claudino, Richard Socher, and Hal Daumé III. 2014. A neural network for factoid question answering over paragraphs. In *Empirical Methods in Natural Language Processing*.

Sergio Jimenez, George Duenas, Julia Baquero, and Alexander Gelbukh. 2014. UNAL-NLP: Combining soft cardinality features for semantic textual similarity, relatedness and entailment. In *Proceedings of SemEval 2014: International Workshop on Semantic Evaluation.*

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1700–1709, Seattle, Washington, USA. Association for Computational Linguistics.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. 2016. Character-aware neural language models. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Alice Lai and Julia Hockenmaier. 2014. Illinois-lh: A denotational and distributional approach to semantics. In *Proceedings of SemEval 2014: International Workshop on Semantic Evaluation.*

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

Richard Socher, Jeffrey Pennington, Eric H Huang, Andrew Y Ng, and Christopher D Manning. 2011b. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 151–161. Association for Computational Linguistics.

Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075*.

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 SemEval 2014: International Workshop on Semantic Evaluation.*