A Long Short-Term Memory Model for Answer Sentence Selection in Question Answering

Di Wang and **Eric Nyberg**
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213, USA
{diwang,ehn}@cs.cmu.edu

Abstract

In this paper, we present an approach that address the answer sentence selection problem for question answering. The proposed method uses a stacked bidirectional Long-Short Term Memory (BLSTM) network to sequentially read words from question and answer sentences, and then outputs their relevance scores. Unlike prior work, this approach does not require any syntactic parsing or external knowledge resources such as WordNet which may not be available in some domains or languages. The full system is based on a combination of the stacked BLSTM relevance model and keywords matching. The results of our experiments on a public benchmark dataset from TREC show that our system outperforms previous work which requires syntactic features and external knowledge resources.

1 Introduction

A typical architecture of open-domain question answering (QA) systems is composed of three high level major steps: a) question analysis and retrieval of candidate passages; b) ranking and selecting of passages which contain the answer; and optionally c) extracting and verifying the answer (Prager, 2006; Ferrucci, 2012). In this paper, we focus on the answer sentence selection. Being considered as a key subtask of QA, the selection is to identify the answer-bearing sentences from all candidate sentences. The selected sentences should be relevant to and answer the input questions.

The nature of this task is to match not only the words but also the meaning between question and answer sentences. For instance, although both of the following sentences contain keywords "Capriati" and "play", only the first sentence answers the question: "What sport does Jennifer Capriati play?"

Positive Sentence: "Capriati, 19, who has not played competitive *tennis* since November 1994, has been given a wild card to take part in the Paris tournament which starts on February 13."

Negative Sentence: "Capriati also was playing in the U.S. Open semifinals in '91, one year before Davenport won the junior title on those same courts."

Besides its application in the automated factoid QA system, another benefit of the answer sentence selection is that it can be potentially used to predict answer quality in community QA sites. The techniques developed from this task might also be beneficial to the emerging real-time user-oriented QA tasks such as TREC LiveQA. However, user-generated content can be noisy and hard to parse with off-the-shelf NLP tools. Therefore, methods that requires less syntactic features are desirable.

Recently, neural network-based distributed sentence modeling has been found successful in many natural language processing tasks such as word sense disambiguation (McCarthy et al., 2004), discourse parsing (Li et al., 2014), machine translation (Sutskever et al., 2014; Cho et al., 2014), and paraphrase detection (Socher et al., 2011).

In this paper, we present an approach that leverages the power of deep neural network to address the answer sentence selection problem for question answering. Our method employs stacked bidirectional Long Short-Term Memory (BLSTM) to sequentially read the words from question and answer sentences, and then output their relevance scores. The full system, when combined with keywords matching, outperforms previous approaches without using any syntactic parsing or external knowledge resources.

Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Short Papers), pages 707–712,
Beijing, China, July 26-31, 2015.

2 Related Work

Prior to this work there were other approaches to address the sentence selection task. The majority of previous approaches focused on syntactic matching between questions and answers. Punyakanok et al. (2004) and Cui et al. (2005) were among the earliest to propose the general tree matching methods based on tree-edit distance. Subsequent to these two papers, the approach in (Wang et al., 2007) use quasi-synchronous grammar to match each pair of question and sentence by their dependency trees. Later, tree kernel function together with a logistic regression model (Heilman and Smith, 2010) or Conditional Random Fields models (Wang and Manning, 2010; Yao et al., 2013) with extracted feature were adopted to learn the associations between question and answer. Recently, discriminative tree-edit features extraction and engineering over parsing trees are automated in (Severyn and Moschitti, 2013).

Besides syntactic approaches, lexical semantic model (Yih et al., 2013) is also used to select answer sentences. This model is to pair semantically related words based on word relations including synonymy/antonymy, hypernymy/hyponymy and general semantic word similarity.

There were also prior efforts in deep learning neural networks to question answering. Yih et al. (2014) focused on answering single-relation factual questions by a semantic similarity model using convolutional neural networks. Bordes et al. (2014) jointly embedded words and knowledge base constituents into same vector space to measure the relevance of question and answer sentences in that space. Iyyer et al. (2014) worked on the quiz bowl task, which is an application of recursive neural networks for factoid question answering over paragraphs. The correct answers are identified from a relatively small fixed set of candidate answers which are in the form of entities instead of sentences.

3 Approach

The goal of this system is to reduce as much as possible the dependency on syntactic features and external resources by leveraging the power of deep recurrent neural network architecture. The proposed network architecture is trained directly on the word sequences of question and answer passages, and is actually not limited to sentences.

3.1 Network Architecture

Recurrent Neural Network RNN is an extension of conventional feed-forward neural network, used to deal with variable-length sequence input. It uses a recurrent hidden state whose activation is dependent on that of the one immediate before. More formally, given an input sequence $x = (x_1, x_2, \ldots, x_T)$, a conventional RNN updates the hidden vector sequence $h = (h_1, h_2, \ldots, h_T)$ and output vector sequence $y = (y_1, y_2, \ldots, y_T)$ from $t = 1$ to T as follows:

$$h_t = \mathcal{H}(W_{xh}x_t + W_{hh}h_{t-1} + b_h) \quad (1)$$

$$y_t = W_{hy}h_t + b_y \quad (2)$$

where the W denotes weight matrices, the b denotes bias vectors and $\mathcal{H}(\cdot)$ is the recurrent hidden layer function.

Long Short-Term Memory (LSTM) Due to the gradient vanishing problem, conventional RNNs is found difficult to be trained to exploit long-range dependencies. In order to mitigate this weak point in conventional RNNs, specially designed activation functions have been introduced. LSTM is one of the earliest attempts and still a popular option to tackle this problem. LSTM cell was originally proposed by Hochreiter and Schmidhuber (1997). Several minor modifications have been made to the original LSTM cell since then. In our approach, we adopted a slightly modified implementation of LSTM in (Graves, 2013).

In the LSTM architecture, there are three gates (input i, forget f and output o), and a cell memory activation vector c. The vector formulas for recurrent hidden layer function $\mathcal{H}$ in this version of LSTM network are implemented as following:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \quad (3)$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \quad (4)$$

$$c_t = f_t c_{t-1} + i_t \tau(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \quad (5)$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \quad (6)$$

$$h_t = o_t \theta(c_t) \quad (7)$$

where, τ and θ are the cell input and cell output non-linear activation functions which are stated as $tanh$ in this paper.

LSTM uses input and output gates to control the flow of information through the cell. The input gate should be kept sufficiently active to allow the signals in. Same rule applies to the output gate. The forget gate is used to reset the cell's own state.

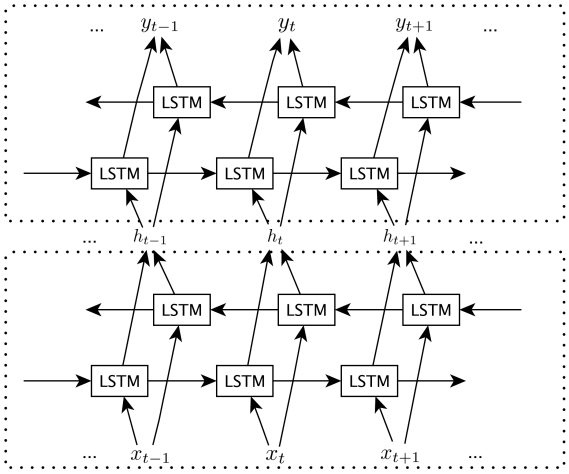

Figure 1: An illustration of a stacked bidirectional LSTM network

In (Graves, 2013), peephole connections are usually used to connect gates to the cell in tasks requiring precise timing and counting of the internal states. In our approach, we don't use peephole connections because the precise timing does not seem to be required.

Bidirectional RNNs Another weak point of conventional RNNs is their utilization of only previous context with no exploitation of future context. Unlike conventional RNNs, bidirectional RNNs utilize both the previous and future context, by processing the data from two directions with two separate hidden layers. One layer processes the input sequence in the forward direction, while the other processes the input in the reverse direction. The output of current time step is then generated by combining both layers' hidden vector $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$ by: $y_t = W_{\overrightarrow{h}y}\overrightarrow{h_t} + W_{\overleftarrow{h}y}\overleftarrow{h_t} + b_y$.

Stacked RNNs In a stacked RNN, the output h_t from the lower layer becomes the input of the upper layer. Through the multi-layer stacked network, it is possible to achieve different levels of abstraction from multiple network layers. There are theoretical supports indicating that a deep, hierarchical model can be more efficient in representing some functions than a shallow one (Bengio, 2009). Empirical performance improvement is also observed in LSTM network compared with the shallow network (Graves et al., 2013).

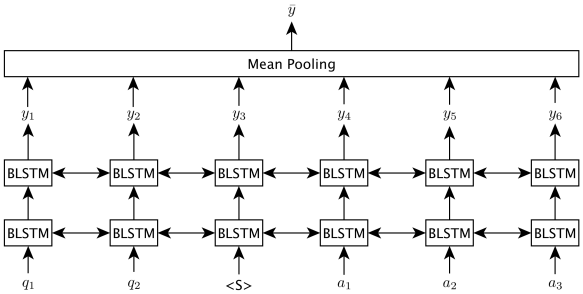

Figure 2: An illustration of our QA sentence relevance model based on stacked BLSTM

3.2 Answer Sentence Selection with Stacked BLSTM

As per analysis in section 3.1, we adopt multilayer stacked bidirectional LSTM RNNs (rather than conventional RNNs) to model the answer sentence selection problem as illustrated in Figure 2. The words of input sentences are first converted to vector representations learned from word2vec tool (Mikolov et al., 2013). In order to differentiate question q and answer a sentences, we insert a special symbol, <S>, after the question sequence. Then, the question and answer sentences word vectors are sequentially read by BLSTM from both directions. In this way, the contextual information across words in both question and answer sentences is modeled by employing temporal recurrence in BLSTM.

Since the LSTM in each direction carries a cell memory while reading the input sequence, it is capable of aggregating the context information and storing it into cell memory vector. For each time step in the BLSTM layer, the hidden vector or the output vector is generated by combining the cell memory vectors from two LSTM of both sides. In other words, all the contextual information across the entire sequence (both question and answer sentences) has been taken into consideration. The final output of each time step is the label indicating whether the candidate answer sentence should be selected as the correct answer sentence for the input question. This objective encourages the BLSTMs to learn a weight matrix that outputs a positive label if there is overlapping context information between two LSTM cell memories. Mean pooling is applied to all time step outputs during the training. During the test phase, we collect mean, sum and max poolings as features.

3.3 Incorporating Keywords Matching

In order to identify the correct candidate answer sentences, it is crucial to match the cardinal numbers and proper nouns with those occurred in the question. However, many cardinal numbers and proper nouns are out of the vocabulary (OOV) of our word embeddings. In addition, some proper nouns' embeddings may bring noise to the matching process. For example, "Japan" and "China" are two words very close in the embedding space. It is critical to discriminate these two proper nouns when matching question and answer sentences. In order to mitigate this weak point of the distributed representations, our full system combined the stacked BLSTM relevance model and exact keywords overlapping baseline by gradient boosted regression tree (GBDT) method (Friedman, 2001).

4 Experiments

Dataset The answer sentence selection dataset used in this paper was created by Wang et al. (2007) based on Text REtrieval Conference (TREC) QA track (8-13) data.[1] Candidate answer sentences were automatically retrieved for each question which is on average associated with 33 candidate sentences. There are two sets of data provided for training. One is the full training set containing 1229 questions that are automatically labeled by matching answer keys' regular expressions.[2] However, the generated labels are noisy and sometimes erroneously mark unrelated sentences as the correct answers solely because those sentences contain answer keys. Wang et al. (2007) also provided one small training set contains 94 questions, which were manually corrected for errors. In our experiments, we use the full training set because it provides significantly more question and answer sentences for learning, even though some of its labels are noisy.

The development and test data sets have 82 and 100 questions, respectively. Following (Wang et al., 2007), candidate answer sentences with over 40 words and questions with only positive or negative candidate answer sentences are removed from evaluation.[3]

Evaluation Metric Following previous works on this task, we also use Mean Average Precision (MAP) and Mean Reciprocal Rank (MRR) as evaluation metrics, which are calculated using the official $trec\_eval$ evaluation scripts.

Keywords Matching Baseline (BM25) As noted by Yih et al. (2013), counting overlapped keywords, especially when re-weighted by idf value of the question word, is a fairly competitive baseline. Following (Yih et al., 2013), our keywords matching baseline also counts the words that occurred in both questions and answer sentences, after excluding stop words and lowering the case. But, instead of the $tf \cdot idf$ formula used in (Yih et al., 2013), word counts are re-weighted by its idf value using the Okapi BM25 (Robertson and Walker, 1997) formula (with constants values $K_1 = 1.2$ and $B = 0.75$).

Network Setup The network weights are randomly initialized using a Gaussian distribution ($\mu = 0$ and $\sigma = 0.1$), and the network is trained with the stochastic gradient descent (SGD) with momentum 0.9. We experimented single-layer unidirectional LSTM, single-layer BLSTM, and three-layer stacked BLSTM. Each layer of LSTM and BLSTM has a memory size of 500. We use 300-dimensional vectors that were trained and provided by word2vec tool (Mikolov et al., 2013) using a part of the Google News dataset[4] (around 100 billion tokens) .

5 Results

Table 1 surveys prior results on this task, and places our models in the context of the current state-of-the-art results. Table 2 summarizes the results of our model on the answer selection task. According to Table 1 and 2, our combined system outperforms prior works on MAP and MRR metrics.

As indicated in Table 2, the three-layer stacked BLSTM alone shows better experiment results than single-layer BLSTM and unidirectional

[1] http://nlp.stanford.edu/mengqiu/data/qg-emnlp07-data.tgz

[2] Because the original full training dataset is no longer available from the website of the lead author of (Wang et al., 2007), we obtained this data re-released from Yao et al. (2013): http://cs.jhu.edu/~xuchen/packages/jacana-qa-naacl2013-data-results.tar.bz2

[3] As mentioned in the footnote 7 of (Yih et al., 2013): *"Among the 72 questions in the test set, 4 of them would always be treated answered incorrectly by the evaluation script used by previous work. This makes the upper bound of both MAP and MRR become 0.9444 instead of 1."* In order to make experiment results comparable with previous works, we also use this experiment setting.

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

Reference	**MAP**	**MRR**
Yih et al. (2013) – Random	0.3965	0.4929
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) – BDT	0.6940	0.7894
Yih et al. (2013) – LCLR	0.7092	0.7700

Table 1: Overview of prior results on the answer sentence selection task

Features	MAP	MRR
BM25	0.6370	0.7076
Single-Layer LSTM	0.5302	0.5956
Single-Layer BLSTM	0.5636	0.6304
Three-Layer BLSTM	0.5928	0.6721
Three-Layer BLSTM + BM25	**0.7134**	**0.7913**

Table 2: Overview of our results on the answer sentence selection task. Features are keywords matching baseline score (BM25), and pooling values of single-layer unidirectional LSTM (Single-Layer LSTM), single-Layer bidirectional LSTM (Single-Layer BLSTM) and three-Layer stacked BLSTM's (Three-Layer BLSTM) outputs. Gradient boosted regression tree (GBDT) method is used to combine features.

LSTM, and performs comparably to previous systems. In order to mitigate the weak point of the distributed representations previously discussed in section 3.3, we combine the stacked BLSTM outputs with a keywords matching baseline (BM25). Our combined system's results are statistically significantly better than the keywords matching baseline (using the Student's t-test with $p < 0.05$) and outperforms previous state-of-art results.

6 Conclusion

In this paper, we presented an approach to address the answer sentence selection problem for question answering, by a combination of the stacked bidirectional LSTM model and keywords matching. The experiments provide strong evidence that distributed and symbolic representations encode complementary types of knowledge, which are all helpful in identifying answer sentences. Based on the experiment results, we found that our model not only performs better than previous work but most importantly does not require any syntactic features or external resources. In the future, we would like to further evaluate the models presented in this paper for different tasks, such as answer quality prediction in Community QA, recognizing textual entailment, and machine comprehension of text.

References

Yoshua Bengio. 2009. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1):1–127. Also published as a book. Now Publishers, 2009.

Antoine Bordes, Sumit Chopra, and Jason Weston. 2014. Question answering with subgraph embeddings. *CoRR*, abs/1406.3676.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, October. Association for Computational Linguistics.

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*, SIGIR '05, pages 400–407, New York, NY, USA. ACM.

David A. Ferrucci. 2012. Introduction to "This is Watson". *IBM Journal of Research and Development*, 56(3):1.

Jerome H. Friedman. 2001. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29:1189–1232.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey E. Hinton. 2013. Speech recognition with deep recurrent neural networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, BC, Canada, May 26-31, 2013*, pages 6645–6649.

Alex Graves. 2013. Generating sequences with recurrent neural networks. *CoRR*, abs/1308.0850.

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*, HLT '10, pages 1011–1019, Stroudsburg, PA, USA. Association for Computational Linguistics.

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

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*.

Jiwei Li, Rumeng Li, and Eduard Hovy. 2014. Recursive deep models for discourse parsing. In *Proceedings of Empirical Methods in Natural Language Processing*.

Diana McCarthy, Rob Koeling, Julie Weeds, and John Carroll. 2004. Finding predominant word senses in untagged text. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, ACL '04, Stroudsburg, PA, USA. Association for Computational Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.

John M. Prager. 2006. Open-domain question-answering. *Foundations and Trends in Information Retrieval*, 1(2):91–231.

V. Punyakanok, D. Roth, and W. Yih. 2004. Mapping dependencies trees: An application to question answering. 1.

Stephen E. Robertson and Steve Walker. 1997. On relevance weights with little relevance information. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '97, pages 16–24, New York, NY, USA. ACM.

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, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 458–467.

Richard Socher, Eric H. Huang, Jeffrey Pennin, Christopher D Manning, and Andrew Y. Ng. 2011. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In J. Shawe-Taylor, R.S. Zemel, P.L. Bartlett, F. Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 801–809. Curran Associates, Inc.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proc. NIPS*, Montreal, CA.

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*, COLING '10, pages 1164–1172, Stroudsburg, PA, USA. Association for Computational Linguistics.

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 (EMNLP-CoNLL)*, pages 22–32, Prague, Czech Republic, June. Association for Computational Linguistics.

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. Association for Computational Linguistics.

Wen-tau 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. Association for Computational Linguistics.

Wen-tau Yih, Xiaodong He, and Christopher Meek. 2014. Semantic parsing for single-relation question answering. In *Proceedings of ACL*. Association for Computational Linguistics.