Jiangnan at SemEval-2018 Task 11: Deep Neural Network with Attention Method for Machine Comprehension Task

Jiangnan Xia
Alibaba Group
Hangzhou, China
jiangnan_xjn@alibaba-inc.com

Abstract

This paper describes our submission for the International Workshop on Semantic Evaluation (SemEval-2018) shared task 11– Machine Comprehension using Commonsense Knowledge (Ostermann et al., 2018b). We use a deep neural network model to choose the correct answer from the candidate answers pair when the document and question are given. The interactions between document, question and answers are modeled by attention mechanism and a variety of manual features are used to improve model performance. We also use CoVe (McCann et al., 2017) as an external source of knowledge which is not mentioned in the document. As a result, our system achieves 80.91% accuracy on the test data, which is on the third place of the leaderboard.

1 Introduction

In recent years, machine reading comprehension (MRC) which attempts to enable machines to answer questions when given a set of documents, has attracted great attentions. Several MRC datasets have been released such as the Stanford Question Answering Dataset (SQuAD) (Rajpurkar et al., 2016) and the Microsoft MAchine Reading COmprehension Dataset (MS-MARCO) (Nguyen et al., 2016). These datasets provide large scale of manually created data, greatly inspired the research in this field. And a series of neural network model, such as BiDAF (Seo et al., 2016), R-Net (Wang et al., 2017), have achieved promising results on these evaluation tasks. However, machine reading comprehension is still a difficult task because without knowledge, machines cannot really understand the question and make a correct answer.

As an effort to discover how machine reading comprehension systems would be benefited from commonsense knowledge, (Ostermann et al., 2018b) developed the Machine Comprehension using Commonsense Knowledge task. In this task, commonsense knowledge is given as the form of script knowledge. Script knowledge is defined as the knowledge about everyday activities which is mentioned in narrative documents. For each document, a series questions are asked and each question is associated with a pair of candidate answers. Machines have to choose which is the correct answer. To let machines make correct decisions, explicit information which can be found in the document and external commonsense knowledge are both required. Table 1 shows an example of the dataset in this task.

In this paper, we make a description about our submission system for the task. The system is based on a deep neural network model. The input of the model is a $(document, question, answer)$ triple and the output is the probability that the answer is the correct one for the given document and question. We also combine the neural network model with a variety of manual features, including word exact match features and token features such as part-of-speech (POS) ,named entity recognition (NER) and term frequency (TF). These manual features are helpful in solving the problem that the correct answer can be easily found in the given document.

Furthermore, for more complicated problem that the answer is not explicitly mentioned in the document, we try to model the interactions between document, question and answer by computing the attention score of question to document and question to answer respectively, which is described in (Lee et al., 2016). These features add soft alignments between similar but non-identical words (Chen et al., 2017). We evaluate our system on the shared task and obtain 80.91% accuracy on the test set, which is on the third place of the leaderboard.

The rest of this paper is organized as follows.

Proceedings of the 12th International Workshop on Semantic Evaluation (SemEval-2018), pages 1063–1067
New Orleans, Louisiana, June 5–6, 2018. ©2018 Association for Computational Linguistics

Document:
I went into my bedroom and flipped the light switch. Oh, I see that the ceiling lamp is not turning on. It must be that the light bulb needs replacement. I go through my closet and find a new light bulb that will fit this lamp and place it in my pocket. I also get my stepladder and place it under the lamp. I make sure the light switch is in the off position. I climb up the ladder and unscrew the old light bulb. I place the old bulb in my pocket and take out the new one. I then screw in the new bulb. I climb down the stepladder and place it back into the closet. I then throw out the old bulb into the recycling bin. I go back to my bedroom and turn on the light switch. I am happy to see that there is again light in my room.
Question1: Which room did the light go out in?
0. Kitchen. (Wrong)
1. Bedroom. (Correct)
Question2: Was the light bulb still hot?
0. yes. (Wrong)
1. No. (Correct)

Table 1: An example from the machine comprehension using commonsense knowledge task (Ostermann et al., 2018b). The first line shows the document and the following lines show question and answer pair respectively. The answer of $question1$ can be easily found in the text while answering $question2$ requires external knowledge which is not mentioned in the text.

Section 2 describes the submission system. Section 3 presents and discusses the experiment results. Section 4 makes a conclusion about our work.

2 Model

In this task, a document (D), a question (Q), and a pair of answers (A_0, A_1) are given and a machine comprehension system should choose the correct answer from the answers pair. We attempt to solve this problem by leveraging a deep neural network model which can generate the probability $p_\theta(A_i|D, Q), i = 0\,or\,1$ that the input answer is correct for the given document and question. The system predicts the probability for each answer in (A_0, A_1) respectively and decides which is the correct answer by comparing their probability scores. We represent the set of all trainable parameters of the neural network model as θ. The model basically consists 3 parts: an encode layer, an interaction layer and a final inference layer, which is depicted in figure 1. Below we will discuss the model in more detail.

2.1 Encode layer

We first represent all tokens of document $\{d_1, ..., d_m\}$, question $\{q_1, ..., q_n\}$ and answer $\{a_1, ..., a_l\}$ as sequences of word embeddings $\{E_1^d, ..., E_m^d\}$, $\{E_1^q, ..., E_n^q\}$ and $\{E_1^a, ..., E_l^a\}$, where m, n and l are sequence lengths of document, question and answer respectively. In this task, we use the 300-dimensional 840B Glove word embeddings (Pennington et al., 2014). We then pass each sequence through a multi-layer bidirectional long short term memory network (BiLSTM) to get the word level semantic representations of each sequence:

$$\boldsymbol{h}_j^d = BiLSTM_j(\{E_i^d\}_{i=1}^m) \quad (1)$$

$$\boldsymbol{h}_j^q = BiLSTM_j(\{E_i^q\}_{i=1}^n) \quad (2)$$

$$\boldsymbol{h}_j^a = BiLSTM_j(\{E_i^a\}_{i=1}^l) \quad (3)$$

The index j represents the jth BiLSTM layer. We concat all the output units of each BiLSTM layer and get the final word level representations: $\boldsymbol{h}^d$, $\boldsymbol{h}^q$ and $\boldsymbol{h}^a$. The BiLSTM layers used to encode document, question and answer sequence share same parameters in order to reduce the number of trainable parameters and make the model uneasily overfitting.

2.2 Interaction layer

This layer models the interactions between document, question and answer. We first align each word representation vectors in the question sequence to document and answer by leveraging attention mechanism and get question-aware representation $\boldsymbol{Att}^d$, $\boldsymbol{Att}^a$ for document and answer respectively:

$$Att_i^d = \Sigma_j s_{i,j}^d h_j^q \quad (4)$$

$$Att_i^a = \Sigma_j s_{i,j}^a h_j^q \quad (5)$$

The attention score $s_{i,j}^d$ captures the similarity between the word representation vector $\boldsymbol{d_i}$ and $\boldsymbol{q_j}$ in document sequence and question sequence respectively. And $s_{i,j}^a$ captures the similarity between answer vector $\boldsymbol{a_i}$ and question vector $\boldsymbol{q_j}$.

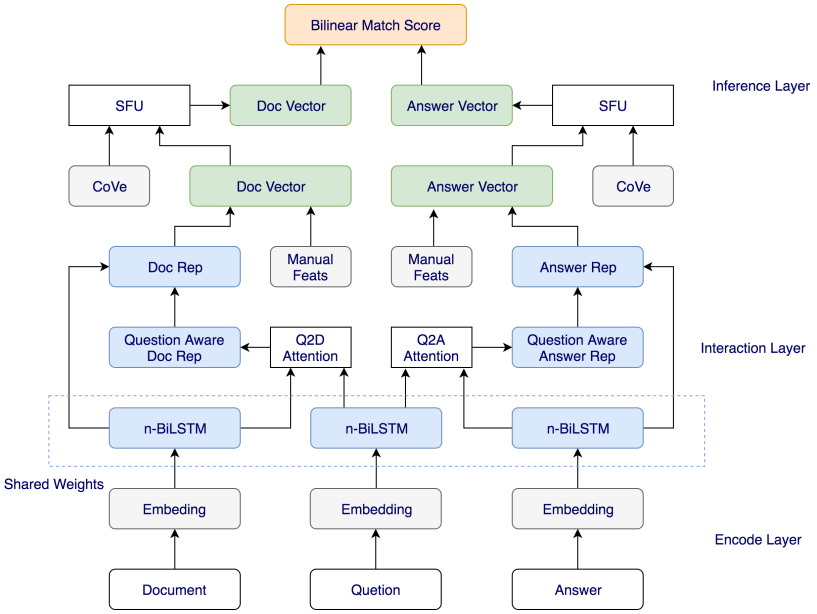

Figure 1: Neural network model architecture for the machine comprehension task

We get $s^d_{i,j}$ and $s^a_{i,j}$ by computing the dot products between the nonlinear mappings of two word representation vectors:

$$s^d_{i,j} = \frac{exp(\alpha(d_i) \cdot \alpha(q_j))}{\Sigma_{j'} exp(\alpha(d_i) \cdot \alpha(q_{j'}))} \quad (6)$$

$$s^a_{i,j} = \frac{exp(\alpha(a_i) \cdot \alpha(q_j))}{\Sigma_{j'} exp(\alpha(a_i) \cdot \alpha(q_{j'}))} \quad (7)$$

$\alpha(\cdot)$ is a single dense layer with ReLU nonlinearity. We concat Att^d_i and Att^a_i behind each h^d_i and h^a_i and get new word representation vectors $\boldsymbol{r}^d$ and $\boldsymbol{r}^a$ for document and answer.

Following (Chen et al., 2017), we combine the model with a variety of manual features, including word exact match features and token features. For exact match features, we use three binary features indicating whether a token in $\boldsymbol{d}$ and $\boldsymbol{a}$ can be exactly matched by one token in $\boldsymbol{q}$, either in its original, lowercase or lemma form. For token features, we use part-of-speech (POS), named entity recognition (NER) and term frequency (TF). For document and answer, we combine the manual features as vectors f^d_i, f^a_i and concat to r^d_i, r^a_i and get new word level representation vectors r'^d_i and r'^a_i:

$$\boldsymbol{r'}^d = \{r'^d_i\}^m_{i=1} = \{[r^d_i; f^d_i]\}^m_{i=1} \quad (8)$$

$$\boldsymbol{r'}^a = \{r'^a_i\}^l_{i=1} = \{[r^a_i; f^a_i]\}^l_{i=1} \quad (9)$$

2.3 Inference layer

In inference layer, we first convert the document and answer sequence $\boldsymbol{r'}^d$, $\boldsymbol{r'}^a$ into fixed length vectors with weighted pooling method and get sequence level representation vectors R_d and R_a:

$$R_d = \Sigma^m_{i=1} u^d_i r'^d_i \quad (10)$$

$$R_a = \Sigma^l_{i=1} u^a_i r'^a_i \quad (11)$$

$$u^d_i = \frac{exp(w^d \cdot r'^d_i)}{\Sigma^m_{j'=1} exp(w^d \cdot r'^d_{j'})} \quad (12)$$

$$u^a_i = \frac{exp(w^a \cdot r'^a_i)}{\Sigma^l_{j'=1} exp(w^a \cdot r'^a_{j'})} \quad (13)$$

The weight vector w^d and w^a are learnable parameters of the model.

As we haven't use any external source of knowledge, we attempt to use other pre-trained language model as external knowledge, in order to get more implicit information which is not mentioned in the document. Here we use CoVe (McCann et al., 2017) in document and answer sequences. The Glove embedding of each token will pass through a pre-trained BiLSTM layer. The BiLSTM layer outputs a sequence of CoVe vectors of document and answer $\boldsymbol{c}^d = \{c^d_i\}^m_{i=1}$, $\boldsymbol{c}^a = \{c^a_i\}^l_{i=1}$. We then convert the sequences into fixed length vectors C_d and C_a by using the weighted pooling method which is mentioned above.

We fuse the pooled CoVe vectors with the sequence level representation vectors with semantic fusion unit (SFU) (Hu et al., 2017) and get the final sequence level representation vectors R'_d and R'_a:

$$R'_d = SFU_d(R_d, C_d) \quad (14)$$

$$R'_a = SFU_a(R_a, C_a) \quad (15)$$

Finally, we represent the probability that the answer is correct by computing the bilinear match score of document and answer vectors:

$$P = \sigma(R'_d W R'_a) \quad (16)$$

W is a trainable matrix and $\sigma(\cdot)$ is the sigmoid function. In this task, we use this model to predict the probability for each answer in (A_0, A_1) and decide which is the correct one by selecting the answer with higher probability score.

3 Experiments

3.1 Datasets

The statistics of official training, development and test data are shown in Table 2.

	Training	**Dev**	**Test**
Num of examples	9,731	1,411	2,797

Table 2: Statistics of the official datasets

We remove the words occurring less than 2 times and finally get about 12000 words in the vocabulary. We keep most pre-trained word embeddings fixed during training and only fine-tune the 100 most frequent words. For manual features, we get POS and NER features by using Stanford CoreNLP[1] toolkits.

3.2 Experimental Settings

We implement our model by using PyTorch [2]. The model is trained in the given training set and we choose the model which performs best on the development set among training epochs. We train the model with mini batch size 32. We use two layers BiLSTM with 128 hidden units. A dropout rate of 0.4 is applied to word embeddings and all hidden units in BiLSTM layers. We use logistic loss as the loss function optimized by using Adamax optimizer (Kingma and Ba, 2014) with learning rate $\eta = 0.002$.

3.3 Results

The performances of our model are depicted in Table 3. The single model achieves accuracy of 85.05% on the development data and 79.03% on the test data. The ensemble model which we finally submitted to the shared task achieves accuracy of 87.30% on the development data and 80.91% on the test data. From the result we can see that there is a gap between development data and test data for both single model and ensemble model. The model overfits the development data but does not perform well on the test data. Shows that the robustness of our model needs to be improved.

We conduct ablation analysis of different features used in the model on the development data. Table 4 shows the ablation analysis results from which we can see that all the features we used can contribute to model performance. Without manual features, the model accuracy is 83.70%, which is 1.3% less than the full model. and without CoVe, the accuracy drops 1.8%. The accuracy drops 6.6% when neither manual features nor CoVe are used. The results show that the model requires both explicit information which can be found in the document and external source of knowledge to make correct decisions.

Model	**Acc.(Dev)**	**Acc.(Test)**
Single Model	0.8505	0.7903
Ensemble Model	0.8730	0.8091

Table 3: Results of the single and ensemble model on development data and test data.

Features	**Acc.(Dev)**
Full	0.8505
w/o Manual features	0.8370
w/o CoVe	0.8320
w/o Manual features and CoVe	0.7845

Table 4: Ablation analysis of features.

4 Conclusion

In this paper, we make a description of our submitted system to the SemEval-2018 shared task 11. The system is based on a deep neural network model which will choose the correct answer from the answers pair when the document and question are given. We combine the model with a variety of manual features which are helpful in solving the problem that the correct answer can be easily found in the given document. For the problem that the answer is not explicitly mentioned in the document, we model the interactions between

[1] https://stanfordnlp.github.io/CoreNLP/
[2] http://pytorch.org/

document, question and answers by using attention mechanism. We also attempt to use CoVe as an external source of knowledge. We conduct experiment and prove that the features we used are helpful in contributing to the model performance. Our system achieves 80.91% accuracy on the test data, which is on the third place of the leaderboard.

References

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Association for Computational Linguistics (ACL)*.

Minghao Hu, Yuxing Peng, and Xipeng Qiu. 2017. Mnemonic reader for machine comprehension. *arXiv preprint arXiv:1705.02798*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kenton Lee, Shimi Salant, Tom Kwiatkowski, Ankur Parikh, Dipanjan Das, and Jonathan Berant. 2016. Learning recurrent span representations for extractive question answering. *arXiv preprint arXiv:1611.01436*.

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: Contextualized word vectors. In *Advances in Neural Information Processing Systems*, pages 6297–6308.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*.

Simon Ostermann, Michael Roth, Modi Ashutosh, Stefan Thater, and Manfred Pinkal. 2018b. Semeval-2018 task 11: Machine comprehension using commonsense knowledge. *In Proceedings of International Workshop on Semantic Evaluation(SemEval-2018)*.

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

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603*.

Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. 2017. Gated self-matching networks for reading comprehension and question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 189–198.