Constrained Deep Answer Sentence Selection

Ahmad Aghaebrahimian(✉)

Faculty of Mathematics and Physics, Institute of Formal and Applied Linguistics, Charles University in Prague, Malostranske nam. 25, 11800 Praha 1, Czech Republic
ebrahimian@ufal.mff.cuni.cz

Abstract. In this paper, we propose Constrained Deep Neural Network (CDNN) a simple deep neural model for answer sentence selection. CDNN makes its predictions based on neural reasoning compound with some symbolic constraints. It integrates pattern matching technique into sentence vector learning. When trained using enough samples, CDNN outperforms regular models. We show how using other sources of training data as a mean of transfer learning can enhance the performance of the network. In a well-studied dataset for answer sentence selection, our network improves the state of the art in answer sentence selection significantly.

Keywords: Deep neural network · Sentence selection · Transfer learning

1 Introduction

A typical Question Answering (QA) system consists of three basic components; passage retrieval, sentence selection, and answer extraction [21]. Given a question, different possible Information Retrieval (IR) methods can be used to extract a relevant passage that hopefully contains the answer.

Given a question and all sentences in its passage, the job of sentence selection component is to choose a subset of sentences as the answer of the question. If a finer answer is expected, the third component (i.e. answer extraction) extracts a word or a span of words from the selected sentences.

In the literature, the last two components are referred to as sentence-level [23] and word-level [18] QA systems. The sentence selection component helps the answer extraction component by eliminating non-relevant sentences and reducing the search space. Moreover, it provides a sentence which can be used as an evidence for the final answer. This formulation of QA has also other applications such as paraphrase detection [27] or Recognizing Textual Entailment (RTE) [16].

Deep Neural Networks (DNN) have been shown to outperform traditional machine learning algorithms in many Natural Language Processing (NLP) tasks. A natural choice for sentence selection using DNNs is the hinge approximation approach in which the weights associated to positive question-answer couples are increased and those associated to negative couples are decreased [19] through training. We extend this idea to integrate the number of shared patterns of question-answer couples into the model.

K. Ekštein and V. Matoušek (Eds.): TSD 2017, LNAI 10415, pp. 57–65, 2017.
DOI: 10.1007/978-3-319-64206-2_7

Such a large DNN has millions of parameters that should be trained properly. This requires a large number of samples which are not available in regular datasets. In order to provide the network with enough training samples, we used SQuAD [18] to train our model and we show that this out-sourced trained model performs remarkably better than previous best models.

The main contributions of this paper are a DNN model for sentence selection and integrating learning transfer into DNNs for this and other similar tasks.

2 Related Work

The emergence of DNNs in recent years has remarkably helped to improve the performance of NLP applications including different kinds of QA systems. From QA systems based on semantic parsing [4,13] to IR-based systems [24], from cloze-type [9,10] to free-text systems [18] and from factoid [1–3] to reasoning-type systems [11] all has been benefited from DNNs.

Before the introduction of DNNs, feature engineering based on knowledge base data, n-grams or syntactic rules [6,14,22] was a common yet cumbersome practice in QA systems. DNNs have helped QA systems in at least two ways; first, to get rid of many time-consuming intermediate feature engineering processes, and second, to reduce the need for domain-specific knowledge and to make transfer learning possible and easier.

In DNN-based QA systems, instead of manual and hard-coded features, DNNs learn an internal representation of both questions and sentences. One common approach in learning the internal representation is to define the problem as a point-wise ranking problem [7,8,26,28]. In point-wise ranking approach, a DNN is trained on tuples of questions and their correct sentences. In this way, it learns a probability distribution over all sentences given each question.

A more intuitive approach to QA systems is to train a DNN on positive and negative sentences at the same time [19]. Our model is similar to this one with two differences. First, we defined the loss function to enforce the number of shared patterns between questions and sentences as a hard constraint. Second, instead of working with triplets (question, positive sample, negative sample), we used tuples (question, positive sample) and (question, negative sample). Our approach is more manageable especially when working with big datasets. Moreover, instead of a convolution, we used an attention mechanism which is much faster and more flexible with the length of sentences.

DNNs are very good at domain adaptation and transfer learning. There are numerous successful cases for transfer learning such as object detection [29], leaning word vectors (embeddings) [15] and most recently Question Answering [16]. Our work is similar to the latter experiment. However, we used exact match and overlap measures to map SQuAD [18] questions to their answer sentences. Besides, the use of the attention mechanism on sentences and our pattern constraint on the hinge approximation helped our model to outperform their models.

3 The Datasets

We tried our model on two widely used datasets namely TrecQA [25] and SQuAD [18].

TrecQA is compiled using the data in TREC 8–13 QA tracks. There is a modified version of TrecQA available in which unanswered questions and questions with only one positive and negative sentences are removed from the development set and the test set divisions. The training questions in both the original and the modified versions are the same.

SQuAD is a dataset for QA in the context of machine comprehension. It includes 107,785 question-answer pairs posed by crowd workers on 536 Wikipedia articles. The answers in SQuAD can be any span of consecutive words in a paragraph which comes with each question. SQuAD is not designed for sentence selection models. However, it is a good source of data for transfer learning experiments [16].

Although SQuAD is not designed originally for answer sentence selection, it can be used for these experiments with some trivial modifications. For this purpose, we first extracted sentences that contain the answer given each question using exact matching.

Each question in SQuAD is mapped to a paragraph. Some paragraphs contain tens of different sentences, and hence by exact matching a question can be mapped to more than one sentence. We assumed that each question has just one positive answer sentence. In order to make sure that each question is mapped to the best possible sentence, in the next step, among sentences which contain the answer, we selected the sentence that has the maximal n-gram overlap with the question. These n-grams were kept limited to uni, bi and trigrams.

To obtain negative sentences, we used two different settings, which seem to cause no real difference in the final results. In the first setting, given a question, we detected the positive sentence as explained above and used the other sentences in the same paragraph as negative samples. In the other setting, we sampled negative sentences for a given question from all the paragraphs disregarding witch paragraph the question belongs to. At the end, we had one positive and up to five negative samples for each question.

The original test set of SQuAD is unseen and is not available online. Hence, for our experiments, we used the original training set for both training and development purposes. For testing purpose, we used the original development set. Table 1 shows the number of the questions in each dataset. Since this is the first time that SQuAD is being tested in a sentence selection experiment, we established a simple baseline for sentence selection in our test set by counting the number of shared uni, bi and trigrams in each question and sentence and assigning the sentence with the highest number of shared patterns to the question.

Table 1. Datasets statistics

	Train set	Dev. set	Test set
Original TrecQA	1229	82	100
Modified TrecQA	1229	65	68
SQuAD	68983	17245	10778

4 The Approach

The task in a sentence selection experiment is to estimate a probability distribution over all sentences given each question and to get the sentence with the highest probability.

$$s_{best} = argmax_s \quad p(\mathbf{s}|\mathbf{q}) \tag{1}$$

For learning the probability associated with each question and sentence or, in other words, for learning the association between questions and their correct sentences, training on negative sentences are informative as training on positive ones. In order to feed both negative and positive sentences into the network at the same time, we defined a loss function that not only considers the similarity between questions and their sentences but also enforces their pattern similarity as a hard constraint.

$$loss = \max\{0, \mathbf{m} - \alpha * \text{Sim}(\mathbf{q}, \mathbf{s}^+) + \beta * \text{Sim}(\mathbf{q}, \mathbf{s}^-)\} \tag{2}$$

This loss decreases with the similarity between a question and its positive samples and increases with the similarity between a question and its negative samples.

Parameter $\mathbf{m}$ is the margin between positive and negative sentences. There is a trade off between the margin and the number of mistakes in positive and negative classification in the model. A margin between 0.01 and 0.1 usually gives good results in this model. $\mathbf{s}^+$ are correct sentences and $\mathbf{s}^-$ are wrong ones. α and β are the numbers of shared patterns between a question and its positive and negative sentences, respectively. Finally, Sim is the similarity function that computes the similarity between questions and sentences.

Among various techniques for measuring the similarity between two sentences, we adopted Geometric mean of Euclidean and Sigmoid Dot product (GESD) [5] which seems to outperform other similarity measures in this model. GESD linearly combines two other similarity measures called L2-norm and inner product. L2-norm is the forward-line semantic distance between two sentences and inner product measures the angle between two sentences vectors.

$$GESD(\mathbf{q}, \mathbf{s}) = \frac{1}{1 + \exp\left(-(\mathbf{q} \cdot \mathbf{s})\right)} * \frac{1}{1 + \|\mathbf{q}, \mathbf{s}\|} \tag{3}$$

In order to provide the loss function with its inputs, we should train three vectors; question, positive sample and negative sample vectors. Similar to [20],

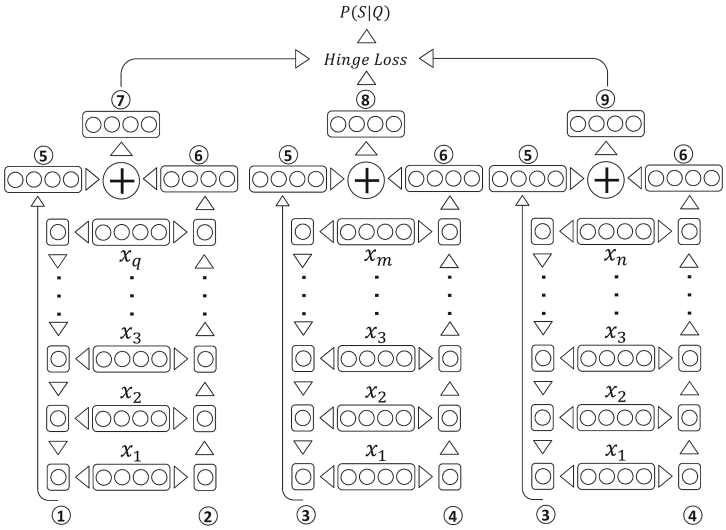

Fig. 1. Abstract model architecture. Numbered components in the figure are: 1-Forward LSTM, 2-Backward LSTM, 3-Forward Attentive LSTM, 4-Backward Attentive LSTM, 5-Backward max-pooling, 6-Forward max-pooling, 7-Question vector, 8-Positive sample vector, 9-Negative sample vector

we defined an embedding, an LSTM with attention layer, a max pooling and a loss function in four separate layers as illustrated in Fig. 1.

Question vectors are generated by passing through these four layers. Embedding layer in our model is a static layer. Word vectors in this layer are initialized with pre-trained 100-dimensional Glove vectors [17]. During training, they were held fixed as we observed no improvements in the performance of the model on the validation set. We also observed no noticeable effect on the performance by increasing the dimensionality of the embeddings to 200 or 300-dimensional vectors.

In the next layer, two LSTM cells are allocated to each token in questions to form a forward and a backward LSTM. Given a question, in forward LSTM, each LSTM cell receives the output of its previous cell as the input layer. It happens from left to right to cover all the tokens. The first LSTM receives the embedding layer output as its input and the output of the last LSTM is fed into the max-pooling layer.

In the backward LSTM, the same question is fed into the same LSTM, but in reverse order (i.e. from right to left). The outputs of forward and backward LSTMs then go through a column-wise max-pooling layer. Similar to [20], the max-pooling layer is a column-wise max operator that estimates the importance of each token given the surrounding context. Max-pooling is applied on both forward and backward LSTMs and the resulting vectors are concatenated to form the final vectors.

For generating positive and negative vectors, positive and negative samples are passed through the same network, where instead of regular LSTM, attention LSTM cells over corresponding questions are used. For this part we used the attentive LSTM model proposed by [20].

Having a couple of questions and sentences, we first compute their vectors and then compute their similarity, which is parameterized by the parameters of the network. Now, by putting the similarity of positive and negative couples into the loss function which was mentioned above, we can train the network to maximize the similarity between questions and the corresponding correct sentences and to minimize the similarity between questions and wrong sentences.

At the end, we rearrange the scores generated by the model by a coefficient of question-sentence shared patterns similar to what we used in the loss function.

5 Experimental Results

For training, we used Adam [12] to optimize the parameters using SGD. We used LSTM with 128 layer size and set the margin of the loss to 0.05. Since the number of the samples in our training set is very large, a couple of iterations over all the samples is enough for training the model.

The questions in TrecQA has more than one positive and negative samples and the output of the model is an ordered list of sentences. Therefore, for evaluation purposes the most suitable measures are Mean Average Precision (MAP) and Mean Reciprocal Rank (MRR).

However, the questions in SQuAD are limited only to one positive answer and a couple of negative ones. For this reason, we used Accuracy to evaluate the performance of the model for SQuAD dataset. It should be mentioned that Accuracy measure is the floor measure for the performance of this system[1]. Table 2 shows the experimental results.

This research demonstrated that transferring learned models which are trained on large datasets to be test on small ones can be beneficial. However,

Table 2. Experimental results. TrecQA-org is the original TrecQA, TrecQA-mod is the modified TrecQA and state-of-the-art refers to [19]

	Trained on	Test set	MRR	MAP	ACC
State-of-the-art	TrecQA	TrecQA-mod	87.7	80.1	-
State-of-the-art	TrecQA	TrecQA-org	83.4	78.8	-
Current paper	SQuAD + TrecQA	TrecQA-mod	86.2	73.2	-
Current paper	SQuAD + TrecQA	TrecQA-org	**89.5**	**79.5**	-
Baseline	SQuAD	SQuAD	-	-	69
Current paper	SQuAD	SQuAD	93.4	90.1	**86**

[1] Compared to MRR and MAP, Accuracy is a pessimistic measure for this experiment, because it just considers the first selected answer and disregards the others.

the opposite does not seem to be valid. It seems that adding training data to a large dataset distorts its true distribution. In our experiment, we get significantly better results when we included SQuAD in TrecQA for training, but including TrecQA in SQuAD for training decreased the model performance on SQuAD.

6 Conclusion

We proposed CDNN as a network which is able to enforce hard constraints on the parameters of a deep neural network. We also showed that applying transfer learning technique on a model with enough learning capacity can obtain state-of-the-art results without dependence on any external resource or doing any time-consuming feature designing procedure.

Acknowledgments. This research was partially funded by the Ministry of Education, Youth and Sports of the Czech Republic under SVV project number 260 453, core research funding, and GAUK 207-10/250098 of Charles University in Prague.

References

1. Aghaebrahimian, A., Jurčíček, F.: Constraint-based open-domain question answering using knowledge graph search. In: Sojka, P., Horák, A., Kopeček, I., Pala, K. (eds.) TSD 2016. LNCS (LNAI), vol. 9924, pp. 28–36. Springer, Cham (2016). doi:10.1007/978-3-319-45510-5_4
2. Aghaebrahimian, A., Jurčíček, F.: Open-domain factoid question answering via knowledge graph search. In: Proceedings of the Workshop on Human-Computer Question Answering, The North American Chapter of the Association for Computational Linguistics (NAACL) (2016)
3. Bordes, A., Usunier, N., Chopra, S., Weston, J.: Large-scale simple question answering with memory networks. arXiv preprint arXiv:1506.02075 (2015)
4. Clarke, J., Goldwasser, D., Chang, M.W., Roth, D.: Driving semantic parsing from the worlds response. In: Proceedings of the Conference on Computational Natural Language Learning (2010)
5. Feng, M., Xiang, B., Glass, M.R., Wang, L., Zhou, B.: Applying deep learning to answer selection: a study and an open task. In: Proceedings of IEEE ASRU Workshop (2015)
6. Fern, S., Stevenson, M.: A semantic similarity approach to paraphrase detection. In: Proceedings of the 11th Annual Research Colloquium of the UK Special-Interest Group for Computational Linguistics (2008)
7. He, H., Gimpel, K., Lin, J.: Multi-perspective sentence similarity modeling with convolutional neural networks. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP) (2015)
8. He, H., Lin, J.: Pairwise word interaction modeling with deep neural networks for semantic similarity measurement. In: The North American Chapter of the Association for Computational Linguistics (NAACL) (2016)

9. Hermann, K.M., Kocisky, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., Blunsom, P.: Teaching machines to read and comprehend. In: Advances in Neural Information Processing Systems (2015)
10. Kadlec, R., Vodolan, M., Libovicky, J., Macek, J., Kleindienst, J.: Knowledge-based dialog state tracking. In: 2014 IEEE Spoken Language Technology Workshop (SLT) (2014)
11. Khashabi, D., Khot, T., Sabharwal, A., Clark, P., Etzioni, O., Roth, D.: Question answering via integer programming over semi-structured knowledge. In: Proceedings of International Joint Conference on Artificial Intelligence (IJCAI) (2016)
12. Kingma, D., Ba, J.: Adam: a method for stochastic optimization. arXiv:1412.6980 (2014)
13. Kwiatkowski, T., Zettlemoyer, L., Goldwater, S., Steedman., M.: Inducing probabilistic CCG grammars from logical form with higher-order unification. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (2010)
14. Madnani, N., Tetreault, J., Chodorow, M.: 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 (2012)
15. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. In: Proceedings of Workshop at ICLR (2013)
16. Min, S., Seo, M., Hajishirzi, H.: Question answering through transfer learning from large fine-grained supervision data. arXiv:1702.02171 (2017)
17. Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: Proceedings of the Empirical Methods in Natural Language Processing (2014)
18. Rajpurkar, P., Zhang, J., Lopyrev, K., Liang, P.: Squad: 100,000+ questions for machine comprehension of text. arXiv preprint arXiv:1606.05250 (2016)
19. Rao, J., He, H., Lin, J.: Noise-contrastive estimation for answer selection with deep neural networks. In: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, CIKM (2016)
20. Santos, C.D., Tan, M., Xiang, B., Zhou, B.: Attentive pooling networks. arXiv:1602.03609 (2016)
21. Tellex, S., Katz, B., Lin, J., Fernandes, A., Marton, G.: Quantitative evaluation of passage retrieval algorithms for question answering. In: SIGIR (2003)
22. Xu, W., Ritter, A., Callison-Burch, C., Dolan, W.B., Ji, Y.: Extracting lexically divergent paraphrases from twitter. Trans. Assoc. Comput. Linguist. **2**, 435–448 (2014)
23. Yang, Y., Yih, W.T., Meek, C.: WikiQA: a challenge dataset for open-domain question answering. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP) (2015)
24. Yao, X., Durme, B.V.: Information extraction over structured data: question answering with freebase. In: Proceedings of Association for Computational Linguistics (2014)
25. Yao, X., Van Durme, B., Callison-Burch, C., Clark, P.: Answer extraction as sequence tagging with tree edit distance. In: HLT-NAACL (2013)
26. Yih, W.T., Chang, M.W., Meek, C., Pastusiak, A.: Question answering using enhanced lexical semantic models. In: Proceedings of Association for Computational Linguistics (ACL) (2013)
27. Yin, W., Schtze, H., Xiang, B., Zhou, B.: ABCNN: attention-based convolutional neural network for modeling sentence pairs. arXiv:1512.05193 (2015)

28. Yu, L., Moritz Hermann, K., Blunsom, P., Pulman, S.: Deep learning for answer sentence selection. In: NIPS Deep Learning Workshop (2014)
29. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8689, pp. 818–833. Springer, Cham (2014). doi:10.1007/978-3-319-10590-1_53