Improved Representation Learning for Question Answer Matching

Ming Tan, Cicero dos Santos, Bing Xiang & Bowen Zhou
IBM Watson Core Technologies
Yorktown Heights, NY, USA
{mingtan,cicerons,bingxia,zhou}@us.ibm.com

Abstract

Passage-level question answer matching is a challenging task since it requires effective representations that capture the complex semantic relations between questions and answers. In this work, we propose a series of deep learning models to address passage answer selection. To match passage answers to questions accommodating their complex semantic relations, unlike most previous work that utilizes a single deep learning structure, we develop hybrid models that process the text using both convolutional and recurrent neural networks, combining the merits on extracting linguistic information from both structures. Additionally, we also develop a simple but effective attention mechanism for the purpose of constructing better answer representations according to the input question, which is imperative for better modeling long answer sequences. The results on two public benchmark datasets, InsuranceQA and TREC-QA, show that our proposed models outperform a variety of strong baselines.

1 Introduction

Passage-level answer selection is one of the essential components in typical question answering (QA) systems. It can be defined as follows: Given a question and a pool of candidate passages, select the passages that contain the correct answer. The performance of the passage selection task is not only crucial to non-factoid QA systems, where a question is expected to be answered with a sequence of descriptive text (e.g. the question in Table 1), but also very important to factoid QA systems, where the answer passage selection step is also known as passage scoring. In factoid QA, if the sentences selected by the passage scorer module do not contain the answer, it will definitely lead to an incorrect response from the QA system.

Question: Does Medicare cover my spouse?
Ground-truth answer: If your spouse has worked and paid Medicare taxes for the entire required 40 quarters, or is eligible for Medicare by virtue of being disabled or some other reason, your spouse can receive his/her own medicare benefits. If your spouse has not met those qualifications, if you have met them, and if your spouse is age 65, he/she can receive Medicare based on your eligibility.
Another candidate answer: If you were married to a Medicare eligible spouse for at least 10 years, you may qualify for Medicare. If you are widowed, and have not remarried, and you were married to your spouse at least 9 months before your spouse's death, you may be eligible for Medicare benefits under a spouse provision.

Table 1: An example of a question with the ground-truth answer and a negative answer extracted from the InsuranceQA dataset.

One central challenge of this task lies in the complex and versatile semantic relations observed between questions and passage answers. For example, while the task of supporting passage selection for factoid QA may be largely cast as a textual entailment problem, what makes an answer better than another in the real world for non-factoid QA often depends on many factors.

Specifically, different from many other pair-matching NLP tasks, the linguistic similarities between questions and answers may or may not be indicative for our task. This is because, depending on what the question is looking for, a good answer may come in different forms: sometimes a correct answer completes the question precisely with the missing information, and in other scenarios, good answers need to elaborate part of the question to rationalize it, and so on. For instance, the question in Table 1 only contains five words, while the best answer uses 60 words for elaboration. On the other hand, the best answers from a pool can also be noisy and include extraneous information irrelevant to the question. Additionally, while a good answer must relate to the question, they often do not share common lexical units. For instance, in the example question, "cover" is not directly mentioned in the answer. This issue may confuse simple word-matching systems.

These challenges consequently make hand-crafting features much less desirable compared to deep learning based methods. Furthermore, they also require our systems to learn how to distinguish useful pieces from irrelevant ones, and further, to focus more on the former.

Finally, the system should be capable of capturing the nuances between the best answer and an acceptable one. For example, the second answer in Table 1 is suitable for a questioner, whose spouse is Medicare eligible, asking about his/her own coverage, while the example question is more likely asked by a person, who is Medicare eligible, asking about his/her spouse' coverage. Clearly, the first answer is more appropriate for the question, although the second one implicitly answers it. A good system should reflect this preference.

While this task is usually approached as a pairwise-ranking problem, the best strategy to capture the association between the questions and answers is still an open problem. Established approaches normally suffer from two weaknesses at this point. First, prior work, such as (Feng et al., 2015; Wang and Nyberg, 2015), resort to either convolutional neural network (CNN) or recurrent neural network (RNN) respectively. However, each structure describes only one semantic perspective of the text. CNN emphasizes the local interaction within n-gram, while RNN is designed to capture long range information and forget unimportant local information. How to combine the merits from both has not been sufficiently explored. Secondly, previous approaches are usually based on independently generated question and answer embeddings; the quality of such representations, however, usually degrades as the answer sequences grow longer.

In this work, we propose a series of deep learning models in order to address such weaknesses. We start with the basic discriminative framework for answer selection. We first propose two independent models, Convolutional-pooling LSTM and Convolution-based LSTM, which are designed to benefit from both of the two popular deep learning structures to distinguish better between useful and irrelevant pieces presented in questions and answers. Next, by breaking the independence assumption of the question and answer embedding, we introduce an effective attention mechanism to generate answer representations according to the question, such that the embeddings do not overlook informative parts of the answers. We report experimental results for two answer selection datasets: (1) InsuranceQA (Feng et al., 2015) [1], a recently released large-scale non-factoid QA dataset from the insurance domain, and (2) TREC-QA [2], which was created by Wang et al. (2007) based on Text REtrieval Conference (TREC) QA track data.

The contribution of this paper is hence three-fold: 1) We propose hybrid neural networks, which learn better representations for both questions and answers by combining merits of both RNN and CNN. 2) We prove the effectiveness of attention on the answer selection task, which has not been sufficiently explored in prior work. 3) We achieve the state-of-the-art results on both TREC-QA and InsuranceQA datasets.

The rest of the paper is organized as follows: Section 2 describes the related work for answer selection; Section 3 provides the details of the proposed models; Experimental settings and results are discussed in Section 4 and 5; Finally, we draw conclusions in Section 6.

2 Related work

Previous work on answer selection normally used feature engineering, linguistic tools, or external resources. For example, semantic features were constructed based on WordNet in (Yih et al., 2013). This model pairs semantically related words based on word semantic relations. In (Wang and Manning, 2010; Wang et al., 2007), the answer selection problem was transformed to a syntacti-

[1] git clone https://github.com/shuzi/insuranceQA.git (We use the V1 version of this dataset).

[2] The data is obtained from (Yao et al., 2013) http://cs.jhu.edu/~xuchen/packages/jacana-qa-naacl2013-data-results.tar.bz2

cal matching between the question/answer parse trees. Some work tried to fulfill the matching using minimal edit sequences between dependency parse trees (Heilman and Smith, 2010; Yao et al., 2013). Discriminative tree-edit feature extraction and engineering over parsing trees were automated in (Severyn and Moschitti, 2013). Such methods might suffer from the availability of additional resources, the effort of feature engineering and the systematic complexity introduced by the linguistic tools, such as parse trees and dependency trees.

Some recent work has used deep learning methods for the passage-level answer selection task. The approaches normally pursue the solution on the following directions. First, a joint feature vector is constructed based on both the question and the answer, and then the task can be converted into a classification or ranking problem (Wang and Nyberg, 2015; Hu et al., 2014). Second, recently proposed models for text generation can intrinsically be used for answer selection and generation (Bahdanau et al., 2015; Vinyals and Le, 2015). Finally, the question and answer representations can be learned and then matched by certain similarity metrics (Feng et al., 2015; Yu et al., 2014; dos Santos et al., 2015; Qiu and Huang, 2015). Fundamentally, our proposed models belong to the last category.

Meanwhile, attention-based systems have shown very promising results on a variety of NLP tasks, such as machine translation (Bahdanau et al., 2015; Sutskever et al., 2014), machine reading comprehension (Hermann et al., 2015), text summarization (Rush et al., 2015) and text entailment (Rocktäschel et al., 2016). Such models learn to focus their attention to specific parts of their input and most of them are based on a one-way attention, in which the attention is basically performed merely over one type of input based on another (e.g. over target languages based on the source languages for machine translation, or over documents according to queries for reading comprehension). Most recently, several two-way attention mechanisms are proposed, where the information from the two input items can influence the computation of each others representations. Rocktäschel et al. (2016) develop a two-way attention mechanism including another one-way attention over the premise conditioned on the hypothesis, in addition to the one over hypothesis conditioned on premise. dos Santos et al. (2016) and Yin et al. (2015) generate interactive attention weights on both inputs by assignment matrices. Yin et al. (2015) use a simple Euclidean distance to compute the interdependence between the two input texts, while dos Santos et al. (2016) resort to attentive parameter matrices.

3 Approaches

In this section, we first present our basic discriminative framework for answer selection based on long short-term memory (LSTM), which we call QA-LSTM. Next, we detail the proposed hybrid and attentive neural networks that are built on top of the QA-LSTM framework.

3.1 LSTM for Answer Selection

Our LSTM implementation is similar to the one in (Graves et al., 2013) with minor modifications. Given an input sequence $\mathbf{X} = \{\mathbf{x}(1), \mathbf{x}(2), \cdots, \mathbf{x}(n)\}$, where $\mathbf{x}(t)$ is an E-dimension word vector in this paper, the hidden vector $\mathbf{h}(t)$ (with size H) at the time step t is updated as follows.

$$\begin{aligned}
i_t &= \sigma(\mathbf{W}_i\mathbf{x}(t) + \mathbf{U}_i\mathbf{h}(t-1) + \mathbf{b}_i) && (1)\\
f_t &= \sigma(\mathbf{W}_f\mathbf{x}(t) + \mathbf{U}_f\mathbf{h}(t-1) + \mathbf{b}_f) && (2)\\
o_t &= \sigma(\mathbf{W}_o\mathbf{x}(t) + \mathbf{U}_o\mathbf{h}(t-1) + \mathbf{b}_o) && (3)\\
\tilde{C}_t &= \tanh(\mathbf{W}_c\mathbf{x}(t) + \mathbf{U}_c\mathbf{h}(t-1) + \mathbf{b}_c) && (4)\\
C_t &= i_t * \tilde{C}_t + f_t * C_{t-1} && (5)\\
\mathbf{h}_t &= o_t * \tanh(C_t) && (6)
\end{aligned}$$

There are three gates (input i, forget f and output o), and a cell memory vector C_t. σ is the $sigmoid$ function. $\mathbf{W} \in R^{H\times E}$, $\mathbf{U} \in R^{H\times H}$ and $\mathbf{b} \in R^{H\times 1}$ are the network parameters.

Single-direction LSTMs suffer from the weakness of not making use of the contextual information from the future tokens. Bidirectional LSTMs (biLSTMs) use both the previous and future context by processing the sequence in two directions, and generate two sequences of output vectors. The output for each token is the concatenation of the two vectors from both directions, i.e. $h_t = \overrightarrow{h_t} \parallel \overleftarrow{h_t}$.

QA-LSTM: Our basic answer selection framework is shown in Figure 1. Given an input pair (q,a), where q is a question and a is a candidate answer, first we retrieve the word embeddings (WEs) of both q and a. Then, we separately apply a biLSTM over the two sequences of WEs. Next, we generate a fixed-sized distributed vector representations using one of the following three approaches: (1) the concatenation of the last vectors on both directions of the biLSTM; (2) average pooling over all the output vectors of the biLSTM; (3) max pooling over all the output vectors. Finally, we use cosine similarity $sim(q, a)$ to score the input (q, a) pair. It is important to note that the same biLSTM is applied to both q and a.

Similar to (Feng et al., 2015; Weston et al., 2014; Hu et al., 2014), we define the training objective as a hinge loss.

$$\mathcal{L} = \max\{0, M - sim(q, a_+) + sim(q, a_-)\} \quad (7)$$

where a_+ is a ground truth answer, a_- is an incorrect answer randomly chosen from the entire answer space, and M is a margin. We treat any question with more than one ground truth as multiple training examples. During training, for each question we randomly sample K negative answers, but only use the one with the highest $\mathcal{L}$ to update the model. Finally, dropout operation is performed on the representations before cosine similarity matching.

The same scoring function, loss function and negative sampling procedure is also used in the NN architectures presented in what follows.

3.2 Convolutional LSTMs

The pooling strategies used in QA-LSTM suffer from the incapability of filtering important local information, especially when dealing with long answer sequences.

Also, it is well known that LSTM models successfully keep the useful information from long-range dependency. But the strength has a trade-off effect of ignoring the local n-gram coherence. This can be partially alleviated with bidirectional architectures.

Meanwhile, the convolutional structures have been widely used in the question answering tasks, such as (Yu et al., 2014; Feng et al., 2015; Hu et al., 2014). Classical convolutional layers usually emphasize the local lexical connections of the n-gram. However, the local pieces are associated with each other only at the pooling step. No long-range dependencies are taken into account during the formulation of convolution vectors.

Fundamentally, recurrent and convolutional neural networks have their own pros and cons, due to their different topologies. How to keep both merits motivates our studies of the following two hybrid models.

3.2.1 Convolutional-pooling LSTMs

In Figure 2 we detail the convolutional-pooling LSTM architecture. In this NN architecture, we replace the simple pooling layers (average/max-pooling) by a convolutional layer, which allows to capture richer local information by applying a convolution over sequences of LSTM output vectors. The number of output vectors k (context window size) considered by the convolution is a hyper-parameter of the model.

The convolution structure adopted in this work is as follows: $Z \in \mathcal{R}^{k|h| \times L}$ is a matrix where the m-th column is the concatenation of k hidden vectors generated from biLSTM centralized in the m-th word of the sequence, L is the length of the sequence after wide convolution (Kalchbrenner et al., 2014). The output of the convolution with c filters is,

$$C = \tanh(\mathbf{W}_{cp} Z) \quad (8)$$

where W_{cp} are network parameters, and $C \in \mathcal{R}^{c \times L}$. The j-th element of the representation vectors ($\mathbf{o}_q$ and $\mathbf{o}_a$) is computed as follows,

$$[o_j] = \max_{1<l<L} [C_{j,l}] \quad (9)$$

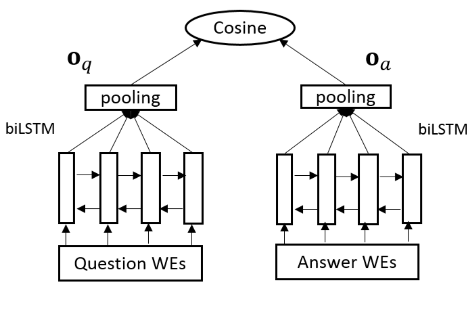

Figure 1: Basic Model: QA-LSTM

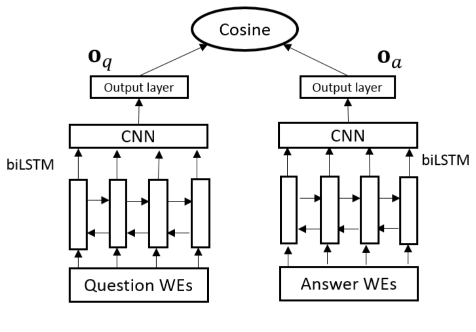

Figure 2: Convolutional-pooling LSTM

3.2.2 Convolution-based LSTMs

In Figure 3, we detail our second hybrid NN architecture. The aim of this approach is to capture the local n-gram interaction at the lower level using a convolution. At the higher level, we build bidirectional LSTMs, which extract the long range dependency based on convoluted n-gram. Combining convolutional and recurrent structures have been investigated in prior work other than question answering (Donahue et al., 2015; Zuo et al., 2015; Sainath et al., 2015).

As shown in Figure 3, the model first retrieves word vectors for each token in the sequence. Next, we compose the matrix $D \in \mathcal{R}^{kE \times L}$, where each column l in D consists of the concatenation of k word vectors of size E centered at the l-th word. The matrix $\mathbf{X} \in \mathcal{R}^{c \times L}$, which is the output of the convolution with c filters is computed as follows:

$$\mathbf{X} = \tanh(\mathbf{W}_{cb}D) \tag{10}$$

The matrix $\mathbf{X}$ is the input to the biLSTM structure in Eqs. 1-6. After the biLSTM step, we use max-pooling over the biLSTM output vectors to obtain the representations of both q and a.

3.3 Attentive LSTMs

In the previous subsections, the two most popular deep learning architectures are integrated to generate semantic representations for questions and answers from both the long-range sequential and local n-gram perspectives.

QA-LSTM and the two proposed hybrid models are basically siamese networks (Chopra et al., 2005). These structures overlook another potential issue. The answers might be extremely long and contain lots of words that are not related to the question at hand. No matter what advanced neural networks are exploited at the answer side, the resulting representation might still be distracted by non-useful information. A typical example is the second candidate answer in Table 1. If the construction of the answer representation is not aware of the input question, the representation might be strongly influenced by n-grams such as "are widowed" and "your spouse's death", which are informative if we only look at the candidate answer, but are not so important for the input question. We address this problem by developing a simple attention model for the answer vector generation, in order to alleviate this weakness by dynamically aligning the more informative parts of answers to the questions.

Inspired by the work in (Hermann et al., 2015), we develop a very simple but efficient word-level attention on the basic model. In Figure 4, we detail our Attentive LSTM architecture. Prior to the average or mean pooling, each biLSTM output vector is multiplied by a softmax weight, which is determined by the question representation from biLSTM. Specifically, given the output vector of biLSTM on the answer side at time step t, $\mathbf{h}_a(t)$, and the question representation, $\mathbf{o}_q$, the updated vector $\widetilde{\mathbf{h}}_a(t)$ for each answer token are formulated below.

$$\begin{aligned}
\mathbf{m}_{a,q}(t) &= \mathbf{W}_{am}\mathbf{h}_a(t) + \mathbf{W}_{qm}\mathbf{o}_q & (11)\\
s_{a,q}(t) &\propto \exp(\mathbf{w}_{ms}^T \tanh(\mathbf{m}_{a,q}(t))) & (12)\\
\widetilde{\mathbf{h}}_a(t) &= \mathbf{h}_a(t)s_{a,q}(t) & (13)
\end{aligned}$$

where $\mathbf{W}_{am}$, $\mathbf{W}_{qm}$ and $\mathbf{w}_{ms}$ are attention parameters. Conceptually, the attention mechanism gives more weight to certain words of the candidate answer, where the weights are computed by taking into consideration information from the question. The expectation is that words in the candidate answer that are more important with regard to the input question should receive larger weights.

The attention mechanism in this paper is conceptually analogous to the one used in one-layer

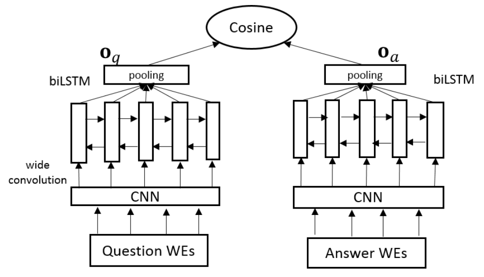

Figure 3: Convolution-based LSTM

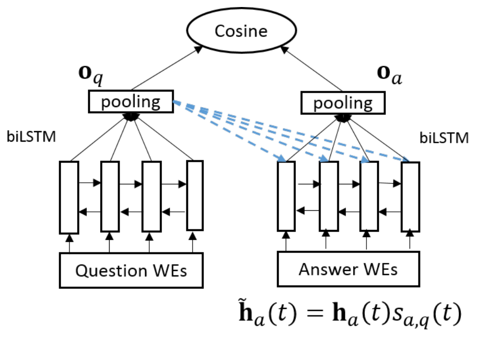

Figure 4: Attentive LSTM

	Train	Validation	Test1	Test2
# of Qs	12887	1000	1800	1800
# of As	18540	1454	2616	2593

Table 2: Numbers of Qs and As in InsuranceQA.

memory network (Sukhbaatar et al., 2015). The fundamental difference is that the transformed question vector and answer unit vectors are combined in an inner-product pattern in order to generate attentive weights in memory network, whereas this work adopts a summation operation (Eq. 11).

4 InsuranceQA Experiments

The first dataset we use to evaluate the proposed approaches is the InsuranceQA, which has been recently proposed by Feng et al. (2015). We use the first version of this dataset. This dataset contains question and answer pairs from the insurance domain and is already divided into a training set, a validation set, and two test sets. We do not see any obvious categorical differentiation between two tests' questions. We list the numbers of questions and answers of the dataset in Table 2. We refer the reader to (Feng et al., 2015), for more details regarding the InsuranceQA data. In this dataset, a question may have multiple correct answers, and normally the questions are much shorter than answers. The average length of questions in tokens is 7, while the average length of answers is 94. Such difference posts additional challenges for the answer selection task. This corpus contains 24981 unique answers in total. For the development and test sets, the InsuranceQA also includes an answer pool of 500 candidate answers for each question. These answer pools were constructed by including the correct answer(s) and randomly selected candidates from the complete set of unique answers. The top-1 accuracy of the answer selection is reported.

4.1 Setup

The proposed models are implemented with Theano (Bastien et al., 2012) and all experiments are conducted in a GPU cluster. We use the accuracy on validation set to select the best epoch and best hyper-parameter settings for testing.

The word embeddings are pre-trained, using word2vec (Mikolov et al., 2013) [3]. The training data for the word embeddings is a Wikipedia corpus of 164 million tokens combined with the questions and answers in the InsuranceQA training set. The word vector size is set to 100. Word embeddings are also part of the parameters and are optimized during the training. Stochastic Gradient Descent (SGD) is the optimization strategy. The learning rate λ is 1.1. We get the best performances when the negative answer count $K = 50$. We also tried different margins in the hing loss function, and finally fixed the margin as M=0.2. We train our models in mini-batches (with batch size as 20), and the maximum length L of questions and answers is 200. Any tokens out of this range are discarded. In order to get more obvious comparison between the proposed models and the basic framework, with respect to the ground-truth answer length in Fig. 5, we also provide the results of $K = 1$. In this case, we set $M = 0.1$, $\lambda = 0.1$ and mini-batches as 100 to get the best performance on the validation set. Also, the dimension of LSTM output vectors is 141x2 for bidirectional LSTM in QA-LSTM, Attentive LSTM and Convolutional-pooling LSTM, such that biLSTM has a comparable number of parameters with a single-direction LSTM with 200 dimensions. For Convolution-based LSTM, since LSTM structure is built on the top of CNN, we fixed the CNN output as 282 dimensions and tune the biLSTM hidden vector size in the experiments.

Because the sequences within a mini-batch have different lengths, we use a mask matrix to indicate the real length of each sequence.

4.2 Baselines

For comparison, we report the performances of four baselines in the top group in Table 3: two state-of-the-art non-DL approaches and two variations of a strong DL approach based on CNN.

Bag-of-word: The idf-weighted sum of word vectors is used as a feature vector. The candidates are ranked by the cosine similarity to the question.

Metzler-Bendersky IR model: A state-of-the-art weighted dependency model (Bendersky et al., 2010; Bendersky et al., 2011), which employs a weighted combination of term-based and term proximity-based features to score each candidate.

Architecture-II in (Feng et al., 2015): A CNN model is employed to learn distributed representations of questions and answers. Cosine similarity is used to rank answers.

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

	Model	Validation	Test1	Test2
	Bag-of-word	31.9	32.1	32.2
	Metzler-Bendersky IR model	52.7	55.1	50.8
	CNN (Feng et al., 2015)	61.8	62.8	59.2
	CNN with GESD (Feng et al., 2015)	**65.4**	**65.3**	**61.0**
A	QA-LSTM (head/tail)	54.8	53.6	51.0
B	QA-LSTM (avg pooling,K=50)	55.0	55.7	52.4
C	QA-LSTM (max pooling,K=1)	64.3	63.1	58.0
D	QA-LSTM (max pooling,K=50)	66.6	66.6	63.7
E	Conv-pooling LSTM (c=4000,K=1)	66.2	64.6	62.2
F	Conv-pooling LSTM (c=200,K=50)	66.4	67.4	63.5
G	Conv-pooling LSTM (c=400,K=50)	67.8	67.5	64.4
H	Conv-based LSTM ($\|h\|$=200,K=50)	66.0	66.1	63.0
I	Conv-based LSTM ($\|h\|$=400,K=50)	67.1	67.6	64.4
J	QA-CNN (max-pooling, $k = 3$)	61.6	62.2	57.9
K	Attentive CNN (max-pooling, $k = 3$)	62.3	63.3	60.2
L	Attentive LSTM (avg-pooling K=1)	68.4	68.1	62.2
M	Attentive LSTM (avg-pooling K=50)	68.4	67.8	63.2
N	Attentive LSTM (max-pooling K=50)	**68.9**	**69.0**	**64.8**

Table 3: The experimental results of InsuranceQA.

Architecture-II with Geometricmean of Euclidean and Sigmoid Dot product (GESD): Cosine similarity is replaced by GESD, which got the best performance in (Feng et al., 2015).

4.3 Results and discussions

In this section, we provide detailed analysis on the experimental results. Table 3 summarizes the results of our models on InsuranceQA. From Row (A) to (D), we list QA-LSTM without either CNN structure or attention. They vary on the pooling method used. We can see that by concatenating the last vectors from both directions, (A) performs the worst. We see that using max-pooling (C) is much better than average pooling (B). The potential reason may be that the max-pooling extracts more local values for each dimension. Compared to (C), (D) is better, showing the need of multiple negative answers in training.

Row (E) to (I) show the results of Convolutional-pooling LSTMs and Convolution-based LSTMs with different filter sizes c, biLSTM hidden sizes $|h|$ and negative answer pool size K. Increasing the negative answer pool size, we are allowed to use less filter counts (F vs E). Larger filter counts help on the test accuracies (G vs F) for Convolutional-pooling LSTMs. We have the same observation with larger biLSTM hidden vector size for Convolution-based LSTMs. Both convolutional models outperform the plain QA-LSTM (D) by about 1.0% on test1, and 0.7% on test2.

Rows (L-N) correspond to QA-LSTM with the attention model, with either max-pooling or average pooling. We observe that max-pooling is better than avg-pooling, which is consistent with QA-LSTMs. In comparison to Model (D), Model (N) shows over 2% improvement on both validation and Test1 sets. And (N) gets improvements over the best baseline in Table 3 by 3.5%, 3.7% and 3.8% on the validation, Test1 and Test2 sets, respectively. Compared to Architecture II in (Feng et al., 2015), which involved a large number of CNN filters, (N) model also has fewer parameters.

We also test the proposed attention mechanism on convolutional networks. (J) replaces the LSTM in QA-LSTM with a convolutional layer. We set the filter size $c = 400$ and window size $k = 3$ according to the validation accuracy. (K) performs the similar attention on the convolutional output of the answers. Similar to biLSTM, the attention on the convolutional layer gives over 2% accuracy improvement on both test sets, which proves the attention's efficiency on both CNN and RNN structures.

Finally, we investigate the proposed models on how they perform with respect to long answers. To better illustrate the performance difference, we

Models	MAP	MRR
(Yao et al., 2013)	0.631	0.748
(Severyn and Moschitti, 2013)	0.678	0.736
(Yih et al., 2013)-BDT	0.694	0.789
(Yih et al., 2013)-LCLR	0.709	0.770
(Wang and Nyberg, 2015)	0.713	0.791
Architecture-II (Feng et al., 2015)	0.711	0.800
(Severyn and Moschitti, 2015) w/o additional features	0.671	0.728
(Severyn and Moschitti, 2015) with additional features	**0.746**	**0.808**
A. QA-CNN	0.714	0.807
B. QA-LSTM (max-pooling)	0.733	0.819
C. Conv-pooling LSTM	0.742	0.819
D. Conv-based LSTM	0.737	0.827
E. Attentive LSTM	**0.753**	**0.830**

Table 4: The test set results on TREC-QA

compare the models with $K = 1$ (i.e. the models C, E, L). We divide the questions of Test1 and Test2 sets into eleven buckets, according to the average length of their ground truth answers. As shown in Figure 5, QA-LSTM gets better or similar performance compared to the proposed models on buckets with shorter answers ($L \leq 50$, $50 < L \leq 55$, $55 < L \leq 60$). As the answer lengths increase, the gap between QA-LSTM and other models becomes more obvious. It suggests the effectiveness of Convolutional-pooling LSTM and Attentive LSTM for long-answer questions.

In (Feng et al., 2015), GESD outperforms cosine similarity in their models. However, the proposed models with GESD as similarity scores do not provide any improvement on the accuracy.

5 TREC-QA Experiments

In this section we detail our experimental setup and results using the TREC-QA dataset.

5.1 Data, metrics and baselines

We test the models on TREC-QA dataset, created based on Text REtrieval Conference (TREC) QA track (8-13) data. More detail of the generation steps for this data can be found in (Wang et al., 2007). We follow the exact approach of train/dev/test questions selection in (Wang and Nyberg, 2015), in which all questions with only positive or negative answers are removed. Finally, we have 1162 training, 65 development and 68 test questions. Similar to previous work, we use Mean Average Precision (MAP) and Mean Reciprocal Rank (MRR) as evaluation metrics, which are evaluated using the official scripts.

In the top part of Table 4, we list the performance of recent prior work on this dataset. We implemented the Architecture II in (Feng et al., 2015) from scratch. The CNN structure in (Severyn and Moschitti, 2015) combined with additional human-designed features achieved the best MAP and MRR.

5.2 Setup

We keep the configurations same as those in InsuranceQA in section 4.1, except the following differences: 1) Following Wang and Nyberg (2015), we use 300-dimensional vectors that were trained and provided by word2vec (Mikolov et al., 2013) using a part of the Google News dataset [4]. 2) Since the word vectors of TREC-QA have a greater dimension than InsuranceQA, we accordingly have larger biLSTM hidden vectors and CNN filters, in order not to lose information from word vectors. Here we set both of them as 600. 3) We use the models from the epoch with the best MAP on the validation set. 4) We also observe that because of the smaller data size, we need a decayed learning rate λ in order to stablize the models' training. Specifically, we set the initial $\lambda_0 = 1.1$, and decrease it for each epoch $T > 1$ as $\lambda_T = \lambda_0 / T$. 5) We fix the negative answer size $K = 50$ during training.

5.3 Results

The bottom part of Table 4 shows the performance of the proposed models. For the comparison purpose, we replace biLSTM with a convolution in Model (A), and also use max-pooling to get question and answer embeddings, and call this model QA-CNN. QA-LSTM (B) improves MAP and MRR in more than 1% when compared to QA-CNN (A). Compared to (B), convolutional-pooling (C) performs better on MAP by 0.9%, and convolution-based models on MAP by 0.4% and MRR by 0.8%. Attentive LSTM is the best proposed model, and outperforms the best baseline (Severyn and Moschitti, 2015) by 0.7% on MAP and 2.2% on MRR. Note that the best result in (Severyn and Moschitti, 2015) was obtained by combining CNN-based features with additional human-defined features. In contrast, our attentive LSTM model achieves higher performance without using any human-defined features.

6 Conclusion

In this paper, we address the following problem for the answer passage selection: how can we construct the embeddings for questions and candidate

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

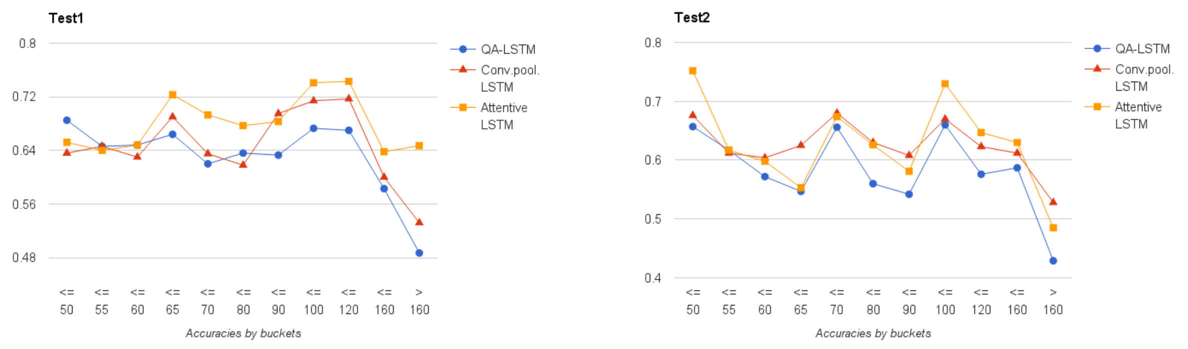

Figure 5: The accuracy of Test1 and Test2 of InsuranceQA sets for three models, i.e. maxpooling QA-LSTM (C), Convolutional-pooling LSTM (E) and Attentive LSTM (L) in Table 3, on different levels of ground truth answer lengths on each test set. The figures show the accuracy of each bucket.

answers, in order to better distinguish the correct answers from other candidates? We propose three independent models in two directions. First, we develop two hybrid models which combine the strength of both recurrent and convolutional neural networks. Second, we introduce a simple one-way attention mechanism, in order to generate answer embeddings influenced by the question context. Such attention fixes the issue of independent generation of the question and answer embeddings in previous work. All proposed models are departed from a basic architecture, built on bidirectional LSTMs. We conduct experiments on InsuranceQA and TREC-QA datasets, and the experimental results demonstrate that the proposed models outperform a variety of strong baselines. Potential future work include: 1) Evaluating the proposed approaches for different tasks, such as community QA and textual entailment; 2) Including the sentential attention mechanism; 3) Integrating the hybrid and the attentive mechanisms into a single framework.

References

Dzmitry Bahdanau, KyungHyun Cho, and Yoshua. Bengio. 2015. Neural machine translation by jointly learning to align and translate. *Proceedings of International conference of learning representations.*

Frederic Bastien, Pascal Lamblin, Razvan Pascanu, James Bergstra, Ian J. Goodfellow, Arnaud Bergeron, Nicolas Bouchard, and Yoshua Bengio. 2012. Theano: new features and speed improvements. Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop.

Michael Bendersky, Donald Metzler, and W. Bruce Croft. 2010. Learning concept importance using a weighted dependence model. In *in Proceedings of the Third ACM International Conference on Web Search and Data Mining (WSDM).*

Michael Bendersky, Donald Metzler, and W. Bruce Croft. 2011. Parameterized concept weighting in verbose queries. In *in Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).*

Sumit Chopra, Raia Hadsell, and Yann LeCun. 2005. Learning a similarity metric discriminatively, with application to face verification. *Computer Vision and Pattern Recognition (CVPR).*

Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. 2015. Long-term recurrent convolutional networks for visual recognition and description. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June.

Cícero dos Santos, Luciano Barbosa, Dasha Bogdanova, and Bianca Zadrozny. 2015. Learning hybrid representations to retrieve semantically equivalent questions. In *Proceedings of ACL*, pages 694–699, Beijing, China, July.

Cícero dos Santos, Ming Tan, Bing Xiang, and Bowen Zhou. 2016. Attentive pooling networks. *CoRR*, abs/1602.03609.

Minwei Feng, Bing Xiang, Michael Glass, Lidan Wang, and Bowen Zhou. 2015. Applying deep learning to answer selection: A study and an open task. *IEEE Automatic Speech Recognition and Understanding Workshop (ASRU).*

Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. 2013. Speech recognition with deep recurrent neural networks. *In IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).*

Michael Heilman and Noah A. Smith. 2010. Tree edit models for recognizing textual entailments, paraphrases, and answers to questions. *Annual Conference of the North American Chapter of the Association for Computational Linguistics. Association for Computational Linguistics (NAACL).*

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. *In Advances in Neural Information Processing Systems (NIPS).*

Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. *Advances in Neural Information Processing Systems (NIPS).*

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. In *ACL*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems (NIPS).*

Xipeng Qiu and Xuanjing Huang. 2015. Convolutional neural tensor network architecture for community-based question answering. *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI).*

Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomás Kociský, and Phil Blunsom. 2016. Reasoning about entailment with neural attention. *International Conference on Learning Representations (ICLR).*

Alexander Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for sentence summarization. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP).*

Tara N. Sainath, Andrew Senior Oriol Vinyals, and Hasim Sak. 2015. Convolutional, long short-term memory, fully connected deep neural networks. In *Acoustics, Speech and Signal Processing (ICASSP), IEEE International Conference*.

Aliaksei Severyn and Alessandro Moschitti. 2013. Automatic feature engineering for answer selection and extraction. *In Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP).*

Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to rank short text pairs with convolutional deep neural networks. In *SIGIR*.

Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. 2015. End-to-end memory networks. *In Advances in Neural Information Processing Systems (NIPS).*

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. *In Advances in Neural Information Processing Systems (NIPS).*

Oriol Vinyals and Quoc V. Le. 2015. A neural conversational model. *Proceedings of the 31st International Conference on Machine Learning*.

Mengqiu Wang and Christopher Manning. 2010. Probabilistic tree-edit models with structured latent variables for textual entailment and question answering. *The Proceedings of the 23rd International Conference on Computational Linguistics (COLING).*

Di Wang and Eric Nyberg. 2015. A long short-term memory model for answer sentence selection in question answering. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*.

Mengqiu Wang, Noah Smith, and Mitamura Teruko. 2007. What is the jeopardy model? a quasi-synchronous grammar for qa. *The Proceedings of EMNLP-CoNLL.*

Jason Weston, Sumit Chopra, and Keith Adams. 2014. #tagspace: Semantic embeddings from hashtags. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP).*

Xuchen Yao, Benjamin Durme, and Peter Clark. 2013. Answer extraction as sequence tagging with tree edit distance. *Proceedings of NAACL-HLT*.

Wen-tau Yih, Ming-Wei Chang, Christopher Meek, and Andrzej Pastusiak. 2013. Question answering using enhanced lexical semantic models. *Proceedings of the 51st Annual Meeting of the Association for Computational Linguist (ACL).*

Wenpeng Yin, Hinrich Schutze, Bing Xiang, and Bowen Zhou. 2015. Abcnn: attention-based convolutional neural network for modeling sentence pairs. *CoRR*, abs/1512.05193.

Lei Yu, Karl M. Hermann, Phil Blunsom, and Stephen Pulman. 2014. Deep learning for answer sentence selection. *NIPS Deep Learning Workshop*.

Zhen Zuo, Bing Shuai, Gang Wang, Xiao Liu, Xingxing Wang, Bing Wang, and Yushi Chen. 2015. Convolutional recurrent neural networks: Learning spatial dependencies for image representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*.