The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Hybrid Attentive Answer Selection in CQA with Deep Users Modelling

Jiahui Wen,[†] Jingwei Ma,[†] Yiliu Feng,[‡] Mingyang Zhong[†]
[†]School of Information Technology and Electrical Engineering, The University of Queensland, Australia
[‡]National University of Defence Technology, China
wenjh.nudt@gmail.com, fengyiliu11@nudt.edu.an, {jingwei.ma, m.zhong1}@uq.edu.au

Abstract

In this paper, we propose solutions to advance answer selection in Community Question Answering (CQA). Unlike previous works, we propose a hybrid attention mechanism to model question-answer pairs. Specifically, for each word, we calculate the intra-sentence attention indicating its local importance and the inter-sentence attention implying its importance to the counterpart sentence. The inter-sentence attention is based on the interactions between question-answer pairs, and the combination of these two attention mechanisms enables us to align the most informative parts in question-answer pairs for sentence matching. Additionally, we exploit user information for answer selection due to the fact that users are more likely to provide correct answers in their areas of expertise. We model users from their written answers to alleviate data sparsity problem, and then learn user representations according to the informative parts in sentences that are useful for question-answer matching task. This mean of modelling users can bridge the semantic gap between different users, as similar users may have the same way of wording their answers. The representations of users, questions and answers are learnt in an end-to-end neural network in a mean that best explains the interrelation between question-answer pairs. We validate the proposed model on a public dataset, and demonstrate its advantages over the baselines with thorough experiments.

Introduction

One of the important task in CQA is automatically selecting the correct answer for a specific question, given the fact that the CQA forums have accumulated a large quantity of questions and the corresponding answers (Zhang et al. 2017; van Dijk, Tsagkias, and de Rijke 2015). Selecting the correct answer can also maximize user engagement with the site and minimize the time for satisfying users seeking for correct answers to their questions (Elkahky, Song, and He 2015).

Many latest research leverage deep learning methods to model the latent representations of question answering (QA) text pairs for texts matching. The deep learning methods can avoid feature engineering and bridge the lexical gap between text pairs. However, there are some shortcomings in the those neural networks based methods. To begin with, some solutions (Severyn and Moschitti 2015; Hu et al. 2014; Wang and Nyberg 2015) propose to model the latent representations of question and its answer independently and defer their interactions later in the softmax or classification layer (Yang et al. 2016). The later interactions between QA text pairs hinder the model to learn the distributed representations of words, hence the latent representations of QA text pairs are not semantic enough to mitigate the lexical gap. To overcome this limitation, some other methods (Yang et al. 2016; Shen et al. 2017) calculate the matching matrix that represents the interactions of sentence word pairs, and employ neural networks on top of it to derive the semantic similarity between QA pairs. However, those methods do not learn the representations of QA pairs, and the interaction matrix is static through the learning process, so the performance of the models is limited by the way that the interaction matrix is calculated.

To deal with the shortcomings of the aforementioned solutions, in this paper, we propose a hybrid attention mechanism for semantic QA pairs matching. The basic idea underlying this method is that, useful parts in sentences for sentence matching need to be locally important (e.g. keywords) and mutually important at the same time. We first calculate individual attentions for each question and answer independently, and then for each word, we calculate its attentions over the words in the counterpart sentence. The former attention (**individual attention**) indicates local importance of a word while the entropy of the later (**interactive attention**) implies the importance of a word with respect to the counterpart sentence. By combining these two attention mechanisms together, we can align the most informative parts in each sentence for the matching task. The interactive attentions are calculated based on the interactions between representations of QA pairs, and the interactions are expressed by the way how the words attend to each other. This coupled interaction and attention provides a deep insight of soft-alignment between QA text pairs. Although attentive neural networks have been applied in answer selection in previous works (Zhang et al. 2017; Yin et al. 2015), they simply compute attention weights for each sentence separately. Specifically, they use attention mechanism to model local importance of each word and summary a sentence based on the weighted word representations.

Furthermore, most of the existing works only model the

semantic similarity between QA pairs for answer selection, while we also exploit other latent factors underlying the correct answer. The motivation is that, answerers are more likely to provide correct answers for questions on their interested topics or in their areas of expertise. Therefore, by modelling users, we can bridge the gap between different users may have the same way of expressing their answers for a same question. Unlike the previous work (Zhao et al. 2017) that also exploits user information, we model users from user-generated answers to alleviate the data sparsity problem, and learn user representations in a way that they can attend to informative parts in questions for answer selection. This joint modelling of users and questions is capable of finding composite and conclusive user representations for a specific answer selection task.

This paper makes the following contributions:

- We propose a hybrid attention mechanism for answer selection in CQA, which takes into consideration the local and mutual importance of the words in QA pairs, and can align the most informative parts for sentence matching.
- We jointly model users and questions in our neural network. The user representations are learnt by explicitly attending to the informative question parts for answer selection task. Therefore, user representations are learnt in a way that best explains user expertise on question topics.
- We demonstrate the effectiveness of the proposed method with thorough experiments on publicly available dataset, and valid the advantage of the proposed method by comparing with the state-of-the-art methods.

Model

In this paper, we formulate answer selection as classification problem. That is given a question $q = \{w_t^q\}_{t=1}^{L^q}$, a candidate answer $a = \{w_t^a\}_{t=1}^{L^q}$ and a text $u = \{w_t^u\}_{t=1}^{L^u}$ of the user who provides the answer, our model is to learn a scoring function $f(q, a, u) \in [0, 1]$ that produces 1 if an answer a is a correct answer to a question q and 0 otherwise, where w_t^q, w_t^a and w_t^u are the $t-$th word in the question, the answer and the user text respectively, and L^q, L^a and L^u are their lengths. In this paper, a user text is all the answers provided by the user. Fig. 1 shows the overview architecture of the proposed model. Generally, the input data goes through the embedding layer, latent representation layer, attention and interaction layer, hidden layer and finally output layer. The detail of each layer is described in the following subsections.

Embedding

We first convert words in the sentences of questions and answers into their corresponding word-level embeddings with Glove (Pennington, Socher, and Manning 2014). The row-rank embedding vectors are able to capture the distributional syntactic and semantic information via the word co-occurrence statistics (Bengio et al. 2003), and the words with similar context are positioned in close proximity to each other in the embedding space.

In this paper, the embedding vectors of words in the training set are initialized with Glove, and remain unchanged during the learning process. To conclude, through the embedding layer, a question and an answer can be represented as a set of embedding vectors:

$$\begin{aligned} \mathbf{E}^q &= \{\mathbf{x}_t^q\}_{t=1}^{L^q} \\ \mathbf{E}^a &= \{\mathbf{x}_t^a\}_{t=1}^{L^a} \end{aligned} \tag{1}$$

where $\mathbf{x}_t^a, \mathbf{x}_t^q \in \mathbb{R}^k$, and k is the embedding size.

Latent Representation

Recurrent Neural Networks (RNNs) are widely applied in natural language processing (NLP) tasks, such as machine translation (Sutskever, Vinyals, and Le 2014), textual entailment recognition (Bowman et al. 2015), and language modelling (Zaremba, Sutskever, and Vinyals 2014). RNNs are mainly proposed to model sequence, and non-linearly transform input vectors into corresponding hidden vectors in a way that optimizes a customized objective function. The long-term history is preserved in RNNs as the hidden vector at current time step depends on the hidden vector at previous time step. Long Short Term Memory (LSTM) is one popular variation of RNN that is first proposed in (Hochreiter and Schmidhuber 1997) to alleviate the gradient vanishing problem of RNN. LSTM introduces three gates and a memory cell to control the data flow within the unit. Given an input sequence $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_1, \cdots, \mathbf{x}_T\}$, the hidden vector at time step t, $\mathbf{h}_t$ is calculate as follows.

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

where $\mathbf{W}_i, \mathbf{W}_f, \mathbf{W}_o, \mathbf{W}_c \in \mathbb{R}^{k\times k}$ are the trainable transformation matrices and $\mathbf{b}_i, \mathbf{b}_f, \mathbf{b}_o, \mathbf{b}_c \in \mathbb{R}^k$ are the corresponding biases that parameterize the gates, σ is the element-wise sigmoid function and $\odot$ is the element-wise multiplication of two vectors. The three gates in a LSTM unit are input gate ($\mathbf{i}_t$), output gate ($\mathbf{o}_t$) and forget gate ($\mathbf{f}_t$), where the input gate denotes the impact of input vectors on the memory cell ($\mathbf{c}_t$), the output gate allows the memory gate to have impact on the output and the forget get determines how the memory cell depends on its previous state.

With the LSTM, the latent representations of question and answer sentence are as follows.

$$\begin{aligned} \mathbf{H}^q &= \{\mathbf{h}_t^q\}_{t=1}^{L^q} = LSTM(\{\mathbf{x}_t^q\}_{t=1}^{L^q}) \\ \mathbf{H}^a &= \{\mathbf{h}_t^a\}_{t=1}^{L^a} = LSTM(\{\mathbf{x}_t^a\}_{t=1}^{L^a}) \end{aligned} \tag{3}$$

Hybrid Attention

Attention mechanism has been successfully applied into a wide range of NLP related tasks, including sentence summarization (Rush, Chopra, and Weston 2015), machine translation (Bahdanau, Cho, and Bengio 2014; Sutskever, Vinyals, and Le 2014). The basic idea underlying attentive mechanism is that it is not necessary to capture semantics of a

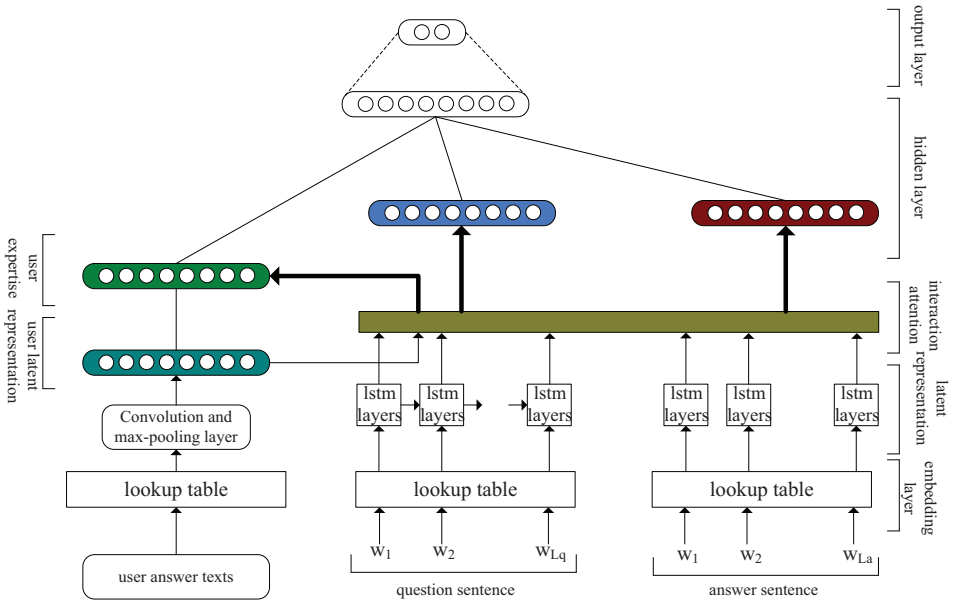

Figure 1: The overview of the proposed neural network

whole sentence (Rocktäschel et al. 2015). On the contrary, one should put more emphasis on sentence parts that are informative and useful for texts matching, while ignore noisy and meaningless parts. In addition, interaction between QA pairs plays an important role in QA matching, as it requires the alignment of semantically similar parts between a sentence and its counterpart.

In this paper, we propose a hybrid attention mechanism to capture both local and mutual importance of a word in QA pairs. We first calculate attention weights for the words in question and answer sentences separately, which indicate the importance of words in a sentence. In parallel, we calculate the attention over the words in one sentence for each word in the counterpart sentence. The information entropy of this attention weights indicates the importance of each word to the other sentence. By combining the two attention mechanisms, we can align the informative parts in inter-sentence and intra-sentence for sentence matching. The individual attentions are obtained by Eq.(4).

$$\boldsymbol{\alpha}^q = softmax(\mathbf{w}_1^T \mathbf{H}^q) \tag{4}$$

where $\mathbf{w}_1 \in \mathbb{R}^k$ is a trainable transformation vector, and $\boldsymbol{\alpha}^q$ is the attention vector for a question sentence with α_i^q indicating the importance of i-th word in the question sentence (Zhai et al. 2016), and the attention of a word can be vary in different sentences, so it is referred to as local importance of a word.

Inspired by (Rocktäschel et al. 2015), we utilize the word-level attention mechanism to calculate interactive attentions. Instead of using the attention to generate words, we exploit the alignment of important parts for encoding the sentence pairs. In addition, the attention is based on the interactions between word pairs rather than their linearly combination. Specifically, let $\mathbf{h}_i^q$ be the hidden vector for the i-th word in a question sentence, and $\mathbf{h}_j^a$ be the hidden vector for the j-th word in the corresponding answer sentence, then the question word's attention over the answer words is obtained as follow.

$$\begin{aligned} \mathbf{m}_{ij} &= tanh(\mathbf{W}^q \mathbf{h}_i^q + \mathbf{W}^a \mathbf{h}_j^a + \mathbf{W}^{qa}(\mathbf{h}_i^q \odot \mathbf{h}_j^a)) \\ \boldsymbol{\gamma}_i^q &= softmax(\mathbf{w}_2^T \mathbf{m}_{i,:}) \\ \beta_i^q &= \mathbf{H}(\boldsymbol{\gamma}_i^q) \end{aligned} \tag{5}$$

where $\mathbf{W}^q, \mathbf{W}^a, \mathbf{W}^{qa} \in \mathbb{R}^{k \times k}$ are trainable transformation matrices, $\mathbf{w}_2$ is a trainable vector. $\mathbf{h}_i^q \odot \mathbf{h}_j^a$ explicitly models the interaction between a question-answer pair, and $\mathbf{m}_{ij} \in \mathbb{R}^k$ is a non-linear combination of $\mathbf{h}_i^q, \mathbf{h}_j^a$, and the interaction between them, representing the intermediate attention of i-th word in the question over j-th word in the answer. $\mathbf{m}_{i,:} \in \mathbb{R}^{k \times L^a}$ is a matrix where the j-th column is $\mathbf{m}_{ij}$. Finally, $\boldsymbol{\gamma}_i^q \in \mathbb{R}^{L^a}$ is a vector containing attentions over the words in the answer for i-th word in the question, and β_i^q is the information entropy of the attention vector. $\boldsymbol{\gamma}_i^q$ depicts the attention distribution of i-th word over the words in the answer. The lower the entropy is, the more likely the word matches some parts of the answer. Therefore, the entropy β_i^q implies the mutual importance of i-word for question-answer matching task.

Finally, the representation of a question can be summarized as in Eq.(6). The significance underlying Eq.(6) is two-fold. On one hand, if a word is locally important but does not align well with the words in the counterpart sentence, it needs to be endowed with less importance as it is useless for semantic matching. On the other hand, if a word is highly related to the counterpart sentence but is not a keyword (e.g. stop words), it should be neglected as it can mislead sentence

matching.

$$\eta_i = \frac{exp(\alpha_i^q / \beta_i^q)}{\sum_j exp(\alpha_j^q / \beta_j^q)}$$
$$\tilde{\mathbf{h}}^q = \sum_{i=1}^{L^q} \eta_i \mathbf{h}_i^q \tag{6}$$

The representation of an answer sentence $\tilde{\mathbf{h}}^a$ can be obtained in a similar way.

User Modelling

The rationale of modelling users is that users are more likely to contribute answers of good quality if the questions reside in their areas of expertise. Therefore, the user's expertise with respect to a question has strong indication of the correctness of his/her answer. Notice that previous work (Zhao et al. 2017) also considers users' information for answer selection. However, modelling users directly from the user-question relations makes their model suffering from cold-start problem, and using inner product to measure the similarity between user and question is inappropriate as they are from different feature spaces. Many recommender systems (Zheng, Noroozi, and Yu 2017; Zhang et al. 2016) model users with the user-generated texts to alleviate the data sparsity problem. Similarly, we model user representations using information contained in their answers, and let them interact with the representations of questions, so that we can model the user expertise and question topics jointly, which can potentially benefit answer selection. In our case, representations of a user and a question indicate user expertise and question latent topics respectively, and the matching between them represents the user's expertise level with respect to the question.

Formally, we concatenate all the answers provided by a user into a single document, $u = \{w_k^u\}_{k=1}^{L^u}$, where L^u is the length of the document. After that, we transform each document into embedding vectors with Glove: $\mathbf{E}^u = \{\mathbf{x}_k^u\}_{k=1}^{L^u}$. The embedding vectors are then fed into a convolution layer and a max pooling layer to obtain a representation for each document. Specifically, for each filter map $K_j \in \mathbb{R}^{(k \times m)}$, we generate a feature map when moving the filter map through the embedding vectors. The feature map is a vector as shown in Eq.(7), where k is the embedding size and m is the filter size.

$$z_t = f(\mathbf{E}^u_{[1:k,t:t+m-1]} * K_j + b_j)$$
$$\mathbf{f}_j = \{z_1, z_2, \cdots, z_{L^u - m + 1}\} \tag{7}$$

where $*$ is convolution operator and f is an activation function (i.e. relu). We then apply max pooling operation over each feature map to extract the most important feature, as presented in Eq.(8). In practice, we apply filter maps of various sizes to the embedding vectors, and set the number of filter map to be equal to the dimension of question representations so that users and questions can interact with each other.

$$o_j = max\{z_1, z_2, \cdots, z_{L^u - m + 1}\}$$
$$\mathbf{h}^u = \{o_1, o_2, \cdots, o_k\} \tag{8}$$

The reason of utilizing attention mechanism to model the interactions between users and questions is that, question sentences usually contain noisy text, and only a small set of them can describe user expertise or interests. Therefore, those keywords need to be attended and endowed with more importance. The attentions are based on both of the question sentences and users due to the fact that users have different areas of expertise or interests.

$$\mathbf{U} = \mathbf{h}^u \oplus \mathbf{e}_{L^q}$$
$$\mathbf{M} = tanh(\mathbf{W}^h \mathbf{H}^q + \mathbf{W}^u \mathbf{U} + \mathbf{W}^{qu}(\mathbf{U} \odot \mathbf{H}^q)$$
$$\boldsymbol{\lambda} = softmax(\mathbf{w}_3^T \mathbf{M})$$
$$\theta_i = \frac{exp(\lambda_i \alpha_i^q / \beta_i^q)}{\sum_j exp(\lambda_j \alpha_j^q / \beta_j^q)} \tag{9}$$
$$\tilde{\mathbf{h}}^u = \sum_{i=1}^{L^q} \theta_i \mathbf{h}_i^q$$

where $\mathbf{h}^u \oplus \mathbf{e}_{L^q}$ is the operation that repeats the vector $\mathbf{h}^u$ for L^q times, and $\mathbf{W}^h, \mathbf{W}^u, \mathbf{W}^{qu} \in \mathbb{R}^{k \times k}$ and $\mathbf{w_3} \in \mathbb{R}^k$ are trainable parameters in our model. The latent vector $\tilde{\mathbf{h}}^u$ is the attended question vector for a specific user considering both of the user representation and the informative parts in the question tailored for answer selection, hence it is more informative and conclusive for a specific answer selection problem than the user representation $\mathbf{h}^u$.

Hidden layer

This hidden layer takes the representations of a question, a candidate answer and the user who provides the answer as input, and outputs a final hidden representation for the tuple (q, a, u). The representation is a non-linear combination of the respective representations of the question, the answer and the user.

$$\mathbf{h} = tanh(\tilde{\mathbf{W}}^q \tilde{\mathbf{h}}^q + \tilde{\mathbf{W}}^a \tilde{\mathbf{h}}^a + \tilde{\mathbf{W}}^u \tilde{\mathbf{h}}^u) \tag{10}$$

where $\tilde{\mathbf{W}}^q, \tilde{\mathbf{W}}^a, \tilde{\mathbf{W}}^u \in \mathbb{R}^{k \times k}$ are the trainable projection matrices, and $\mathbf{h}$ is the final representation for a (q, a, u) tuple.

Output Layer

The output vector from the hidden layer is then passed through a fully connected softmax layer that outputs a posterior probability for each class given a (q, a, u) tuple.

$$p(y = i|(q, a, u)) = \frac{exp(\mathbf{w}_i \mathbf{h} + b_i)}{\sum_j exp(\mathbf{w}_j \mathbf{h} + b_j)} \tag{11}$$

where $\mathbf{w}_i \in \mathbb{R}^k, b_i \in \mathbb{R}$ are trainable parameters for i-th class in the output layer. We formulate the answer selection problem as a binary classification problem, and $p(y = 1|(q, a, u))$ is the probability that answer a is a correct answer for question q.

	Train	Dev	Test
Num. of ques.	5319	244	327
Num. of QA pairs	39563	2440	3270
Avg. len. of ques.	45	53	55
Avg. len. of ans.	37	36	37

Table 1: Statistics of SemEval-2016 Task 3

Training

In this work, we define an end-to-end neural network for answer selection, so the parameters in the network are tuned collaboratively towards the optimization of the pre-defined objective function as shown in Eq.(12).

$$loss = \sum_{(q,a,u,i)\in D} -logp(y = i|(q, a, u)) \qquad (12)$$

where D is the training set, and $i \in \{0, 1\}$ is the ground truth for a (q, a, u) tuple, in which $i = 1$ means a is a correct answer for q, and $i = 0$ otherwise. We use stochastic gradient descent (SGD) with Adam (Kingma and Ba 2014) optimizer to minimize the object function. In our end-to-end neural network, the input data flow through the aforementioned layers till the output layer, based on which the error is calculated and back propagated to the network to update the trainable parameters. Therefore, all the hidden vectors are learnt collaboratively towards optimizing the object function, and this learning process allows the hidden vectors to best characterize the semantic matching between QA text pairs.

Experiment

Dataset

We validate the proposed method on SemEval-2016 Task 3: Community Question Answering (AlessandroMoschitti et al. 2016), as it's a public dataset containing user ID of each question and answer. Another dataset having user information is the Quora dataset (Zhao et al. 2017), but it is not publicly available. In SemEval-2016 Task 3, a question is followed by several comments, and each comment is labelled with one of three labels: "Good", "Bad" and "PotentiallyUseful". A comment is equal to an answer, and it is considered to be correct if it is labelled with "Good", and incorrect otherwise. Table.1 presents the statistics of the CQA dataset used in this paper. There are 8274 users and each user provides 4.7 answers on average.

Experiment Setup

We use Glove to obtain word embeddings of 100 dimensions, and fix them during the training process, thus we are able to preserve the semantic similarity between those words and unseen similar words in the validation and testing phase. As for the out-of-vocabulary words, their embeddings are set by randomly sampling values uniformly from (-0.1, 0.1). The max length of questions and answers are set to 60 and 40 respectively, and all input vectors are padded with 0 to the max length.

All hyper-parameters are tuned on the development dataset with grid search. Specifically, the number of LSTM layers for modelling questions and answers is varied in the range [1,4], and the hidden size k of all LSTM layers and the dimension of hidden layers are amongst {32, 64, 128, 256, 512} separately. As for the CNNs that model user information, the filter size is tuned in the range [3, 5] while the number of filter map is set to the hidden size k. For alleviating overfitting, we apply dropout of 0.5 on the output of LSTMs and CNNs. As for the training configuration, the initial learning rate for Adam optimizer are searched in amongst {1e-4, 1e-3, 1e-2}. The batch size is fixed to 64 and the model is trained for a maximum of 100 epochs. We evaluate the model at every epoch and save the parameters for the top model.

Baselines

We compare our model with other baselines shown in the following lists. For fair comparison, we add additional features to some models as the authors mentioned in their works, such as counts of word overlap in (Tay et al. 2017), and the length of a question in (Zhang et al. 2017), and the hyperparameters are selected as specified in those works.

- **CNN** obtains representations of each sentence with CNN and concatenates them with the their similarity in the joint layer, then uses fully connected layers to produce the matching score (Severyn and Moschitti 2015).
- **CNTN** has the same architecture as CNN except that it uses neural tensor network for calculating the matching score (Qiu and Huang 2015).
- **HD-LSTM** finds sentence representations with LSTM and produces the matching score with holographic composition (Tay et al. 2017).
- **MV-LSTM** models sentence interactions and max-pool the top k values for sentence matching (Wan et al. 2016).
- **WEC** models sentence interactions with a matrix and applies CNNs and multi-layer perceptron (MLP) to produce the final matching score (Shen et al. 2017).
- **AI-CNN** computes the interactions of sentence pair, which results in a 3-D matrix. It then uses max-pooling and attention mechanism to summarize each sentence for final matching (Zhang et al. 2017).
- **AMRNL** finds representations for questions and answers with LSTM, and models users from question-user relations. Finally, it calculates question-answer and question-user similarity for answer selection (Zhao et al. 2017).
- **UIA-LSTM-CNN** our model that employs hybrid attention mechanism to model QA pairs, and explicitly models users for answer selection.

Comparison

We evaluate the listed models with a ranking metric, Mean Average Precision (MAP), and a classification metric, Accuracy. MAP is the average precision across all questions, and it is defined as $\frac{1}{|Q|}\sum_{q\in Q} AvgP(q)$.

Model	MAP	Accuracy
CNN	0.7557	0.7345
CNTN	0.7583	0.7375
HD-LSTM	0.7821	0.7581
MV-LSTM	0.7669	0.7409
WEC	0.7602	0.7325
AI-CNN	0.7891	0.7628
AMRNL	0.7043	0.6854
UIA-LSTM-CNN	0.8086	0.7817

Table 2: Experimental results of all deep learning models on dataset SemEval-2016 Task 3

We divide the baseline models into three group. The first group (i.e. CNN, CNTN, HD-LSTM) learns question and answer representations independently, and defers their interactions later in the neural network. On the contrary, the second group (i.e. MV-LSTM, WEC, AI-CNN) explicitly models the word-to-word interactions with a matrix, and applies different methods (e.g. max-pooling, convolution) to extract deep features. Finally, the third group (i.e. AMRNL) models users for answer selection. From Table.2 we can observe that our model outperforms the models in the first group significantly (3.4% higher than the best model in MAP, and 3.1% in accuracy). The reason is that sentence interactions are deferred until the sentence representations matures (Yang et al. 2016) in those models, as a result the sentence representations cannot be sufficiently learnt to characterize the semantic similarity between question-answer pairs. HD-LSTM achieves the best result in the first group, demonstrating the effectiveness of holographic composition for learning rich representations (Tay et al. 2017). Compared with models in the second group, our model achieves better performance with large margin (2.5% higher than the best model in MAP and accuracy). The reason is that word-to-word matches do not necessarily capture the most informative alignment for answer selection. For example, the top-k matching values max-pooled from the interaction matrix can be signalled by useless text (e.g. stop words), and they can mislead the sentence matching task. Although AI-CNN adopts attention mechanism, it only concatenates question and answer representations as a way of interaction, which incurs additional learning efforts and cannot capture word-to-word semantic alignment. In addition, its attention calculations are based on word-to-word relations and it suffers from the same problem as those models in the second group. Expectedly, AMRNL performs worst compared with all the other models, as it learns user representations from user-question relations, and it faces the problem of cold-start problem in conventional collaborative learning methods. On the contrary, we leverage user-written answers to bridge the gaps among user representations. Additionally, they learn the embedding vectors for a large number of users, and it can easily cause the problem of overfitting. Finally, they simply model sentences with LSTM and take the last hidden state for sentences matching, which faces the risk of long dependency and losing details.

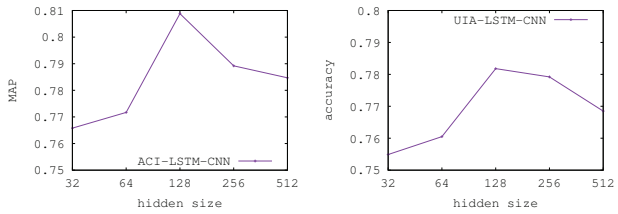

Figure 2: The influence of hidden size on model performance

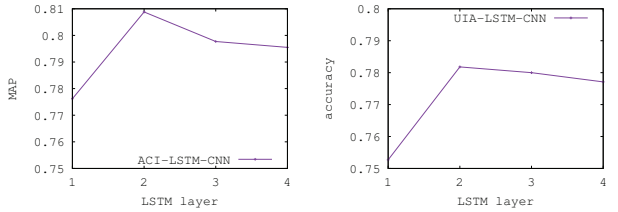

Figure 3: The influence of LSTM layer number on model performance

Parameters Analysis

In this subsection, we analyze the model sensitivity to the hidden size and the number of LSTM layers, and the results are presented in Fig.2 and Fig.3 respectively. From the figures we can observe that there is always a tradeoff between model complexity and overfitting. Specifically, to capture the deep representational features for sentence matching, we need to increase the model complexity to avoid information loss. However, too many parameters can render the model to overfit the training data and not generalized enough to the testing data. Therefore, it is not surprising to see that our model achieves the best result when we set the hidden size to 128 with two stacked LSTMs. Another interesting observation is that, our model is less sensitive to the number of LSTM layers. The reason is that the number of parameters (e.g. transformation matrices) in our model is $O(n)$ to the number of LSTM layers, and it is $O(n^2)$ to the hidden size. Therefore, the squarely growth of parameters can cause overfitting more quickly than the linearly growth of parameters.

Attention Analysis

Fig.4 shows an example of attention visualization in which the attentions with red background is the combination of individual and interactive attentions between the question-answer pair while the attentions with blue background is the user's attentions over the question. From the visualization we have the following observations. First, the informative parts for sentence matching are aligned well in our model. Examples include ('massage', 'where', 'best', 'place', '1000', 'QR') in the question, and ('massage', 'next', 'to', '120', 'riyal', 'not' , 'bad') in the answer. Second, some semantically related words are endowed with less importance because they are not useful for sentence alignment. These examples include ('me', 'you', 'it'), and ('you',

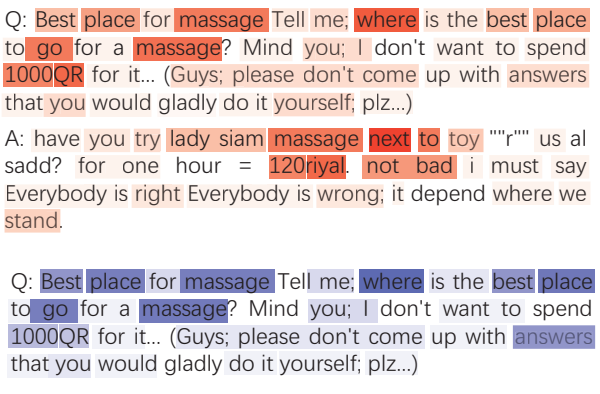

Figure 4: An example of attention visualization

'us', 'everybody', 'where'). Notice that the word 'where' in the question gains more attention than it does in the answer, as the word's occurrence in a question indicates the question topic, and its presence in an answer does not provide any information. Finally, the user's expertise is explicitly modelled through his/her attentions over the question. This user usually provides good answers suggesting good places to go, such as shopping place, nightclub and medical spa. Therefore, the words such as ('best', 'place', 'go', 'answer') gain more attention. However, the user also attends to other words, this is because we include the attended question for calculating the user's attention over the question (i.e. α, β in Eq.(9)).

Related Work

Answer selection plays an important role in CQA, and it has been formulated as a classification or ranking problem in previous research with machine learning methods. The key problem in answer selection is how to model the representations of QA text pairs and the interactions between them. Early works heavily rely on hand-craft features for representing QA text pairs and calculating the matching score between them. Representational works include (Filice et al. 2016; Tymoshenko and Moschitti 2015). The disadvantage of these works is that they require hand-crafted syntactic and semantic feature engineering, linguistic tools, external resources and domain expertise.

Recently, deep learning methods have been applied in answer selection in CQA. In those methods, deep neural networks are applied to learn the latent representations of question and answer sentence independently, and a function is utilized to give the matching score of two texts. The most profound works include (Wang and Nyberg 2015; Yu et al. 2014; Severyn and Moschitti 2015; Hu et al. 2014; Qiu and Huang 2015; Tay et al. 2017). However, these methods learn the representations for QA pairs independently and defer their interactions in the later phase, which results in insufficient exploitation of the semantic correlation between the respective representation of QA pairs.

To explore the deep correlation between QA pairs, many research explicitly model the interactions between question and answer sentences. The interactions are usually formulated as matrices containing results of word-to-word operations (e.g. concatenation, dot product), and then further operations (e.g. convolution, max-pooling) are employed to retrieve the similarity between QA pairs. Representative works include (Hu et al. 2014; Yang et al. 2016; Shen et al. 2017; Wan et al. 2016; Zhang et al. 2017; Yin et al. 2015). However, those methods statically model the interactions between sentence pairs (e.g. cosine similarity of word embedding of word-to-word pairs), and the model performance is limited by the way that the interactions are defined. By contrast, we propose a hybrid attention mechanism that is able to capture the most informative parts for sentence matching. In addition, we model users from user-generated answers and attend them to the useful parts in questions, thus we can explicitly exploit user expertise for answer selection. Some works (Fang et al. 2016; Zhao et al. 2017) also model users for answer selection. They simply utilize user-question relations to independently model users, which suffers from data sparsity problem. While we model users from user-written texts and learn user representations in a way that it can attend to informative parts in questions for answer selection.

Conclusion

In this paper, we propose a hybrid attention deep neural network for answer selection in CQA. Specifically, we calculate attentions for each individual sentence (i.e. question or answer), which indicate the important parts for summarizing the sentence. Simultaneously, we calculate attentions between question-answer pairs based on their interactions, which signify the importance of a word in a sentence with respect to the counterpart sentence. This hybrid attention mechanism enables us to find the most informative words for sentence matching. In addition, we model users in the proposed model based on the fact that users are more likely to provide correct answers in their areas of expertise. As such, users are first modelled from the user-written answers to void data sparsity problem, and then they are attended to the informative parts in questions for question-answer matching. Finally, all the parameters in our end-to-end neural network are collaboratively learnt towards the optimization of the pre-defined objective function. We validate the proposed model on a public dataset and demonstrate its advantages over the state-of-the-art baselines with thorough experiments.

References

AlessandroMoschitti, P. L.; AbedAlhakimFreihat, W. H.; Glass, J.; and Randeree, B. 2016. Semeval-2016 task 3: Community question answering. *Proceedings of SemEval* 525–545.

Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Bengio, Y.; Ducharme, R.; Vincent, P.; and Jauvin, C. 2003. A neural probabilistic language model. *Journal of machine learning research* 3(Feb):1137–1155.

Bowman, S. R.; Angeli, G.; Potts, C.; and Manning, C. D. 2015. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*.

Elkahky, A. M.; Song, Y.; and He, X. 2015. A multi-view deep learning approach for cross domain user modeling in recommendation systems. In *Proceedings of the 24th International Conference on World Wide Web*, 278–288. International World Wide Web Conferences Steering Committee.

Fang, H.; Wu, F.; Zhao, Z.; Duan, X.; Zhuang, Y.; and Ester, M. 2016. Community-based question answering via heterogeneous social network learning. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Filice, S.; Croce, D.; Moschitti, A.; and Basili, R. 2016. Kelp at semeval-2016 task 3: Learning semantic relations between questions and answers. In *SemEval@ NAACL-HLT*, 1116–1123.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Hu, B.; Lu, Z.; Li, H.; and Chen, Q. 2014. Convolutional neural network architectures for matching natural language sentences. In *Advances in neural information processing systems*, 2042–2050.

Kingma, D., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Pennington, J.; Socher, R.; and Manning, C. D. 2014. Glove: Global vectors for word representation. In *EMNLP*, volume 14, 1532–1543.

Qiu, X., and Huang, X. 2015. Convolutional neural tensor network architecture for community-based question answering. In *IJCAI*, 1305–1311.

Rocktäschel, T.; Grefenstette, E.; Hermann, K. M.; Kočiskỳ, T.; and Blunsom, P. 2015. Reasoning about entailment with neural attention. *arXiv preprint arXiv:1509.06664*.

Rush, A. M.; Chopra, S.; and Weston, J. 2015. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685*.

Severyn, A., and Moschitti, A. 2015. Learning to rank short text pairs with convolutional deep neural networks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 373–382. ACM.

Shen, Y.; Rong, W.; Jiang, N.; Peng, B.; Tang, J.; and Xiong, Z. 2017. Word embedding based correlation model for question/answer matching. In *AAAI*, 3511–3517.

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, 3104–3112.

Tay, Y.; Phan, M. C.; Tuan, L. A.; and Hui, S. C. 2017. Learning to rank question answer pairs with holographic dual lstm architecture. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 695–704. ACM.

Tymoshenko, K., and Moschitti, A. 2015. Assessing the impact of syntactic and semantic structures for answer passages reranking. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, 1451–1460. ACM.

van Dijk, D.; Tsagkias, M.; and de Rijke, M. 2015. Early detection of topical expertise in community question answering. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 995–998. ACM.

Wan, S.; Lan, Y.; Guo, J.; Xu, J.; Pang, L.; and Cheng, X. 2016. A deep architecture for semantic matching with multiple positional sentence representations. In *AAAI*, 2835–2841.

Wang, D., and Nyberg, E. 2015. A long short-term memory model for answer sentence selection in question answering. In *Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing*, 707–712.

Yang, L.; Ai, Q.; Guo, J.; and Croft, W. B. 2016. anmm: Ranking short answer texts with attention-based neural matching model. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, 287–296. ACM.

Yin, W.; Schütze, H.; Xiang, B.; and Zhou, B. 2015. Abcnn: Attention-based convolutional neural network for modeling sentence pairs. *arXiv preprint arXiv:1512.05193*.

Yu, L.; Hermann, K. M.; Blunsom, P.; and Pulman, S. 2014. Deep learning for answer sentence selection. *arXiv preprint arXiv:1412.1632*.

Zaremba, W.; Sutskever, I.; and Vinyals, O. 2014. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*.

Zhai, S.; Chang, K.-h.; Zhang, R.; and Zhang, Z. M. 2016. Deepintent: Learning attentions for online advertising with recurrent neural networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1295–1304. ACM.

Zhang, Q.; Gong, Y.; Wu, J.; Huang, H.; and Huang, X. 2016. Retweet prediction with attention-based deep neural network. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, 75–84. ACM.

Zhang, X.; Li, S.; Sha, L.; and Wang, H. 2017. Attentive interactive neural networks for answer selection in community question answering. In *AAAI*, 3525–3531.

Zhao, Z.; Lu, H.; Zheng, V. W.; Cai, D.; He, X.; and Zhuang, Y. 2017. Community-based question answering via asymmetric multi-faceted ranking network learning. In *AAAI*, 3532–3539.

Zheng, L.; Noroozi, V.; and Yu, P. S. 2017. Joint deep modeling of users and items using reviews for recommendation. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, 425–434. ACM.