What We Vote for? Answer Selection from User Expertise View in Community Question Answering

Shanshan Lyu
Yongqing Wang
Wentao Ouyang
Huawei Shen
Xueqi Cheng
CAS Key Laboratory of Network Data Science and Technology,
Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China
University of Chinese Academy of Sciences, Beijing, China
{lvshanshan,wangyongqing,ouyangwt,shenhuawei,cxq}@ict.ac.cn

ABSTRACT

Answer selection is an important problem in community question answering (CQA), as it enables the distilling of reliable information and knowledge. Most existing approaches tackle this problem as a text matching task. However, they ignore the influence of the community in voting the best answers. Answer quality is highly correlated with semantic relevance and user expertise in CQA. In this paper, we formalize the answer selection problem from the user expertise view, considering both the semantic relevance in question-answer pair and user expertise in question-user pair. We design a novel matching function, explicitly modeling the influence of user expertise in community acceptance. Moreover, we introduce latent user vectors into the representation learning of answer, capturing the implicit topic interests in learned user vectors. Extensive experiments on two datasets from real world CQA sites demonstrate that our model outperforms state-of-the-art approaches for answer selection in CQA. Furthermore, the user representations learned by our model provide us a quantitative way to understand both the authority and topic-sensitive interests of users.

CCS CONCEPTS

• **Information systems** → **Collaborative and social computing systems and tools**; • **Computing methodologies** → *Artificial intelligence.*

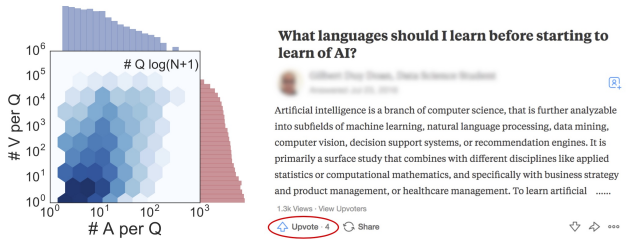
KEYWORDS

User expertise, community question answering, answer selection

ACM Reference Format:
Shanshan Lyu, Yongqing Wang, Wentao Ouyang, Huawei Shen, and Xueqi Cheng. 2019. What We Vote for? Answer Selection from User Expertise View in Community Question Answering. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3308558.3313510

WWW '19, May 13–17, 2019, San Francisco, CA, USA

ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313510



(a) Votes distribution. (b) Example of question, answer, and vote mechanism in CQA.

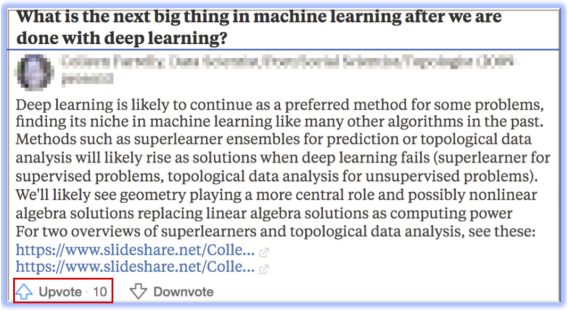
Figure 1: Illustration of vote mechanism in CQA.

1 INTRODUCTION

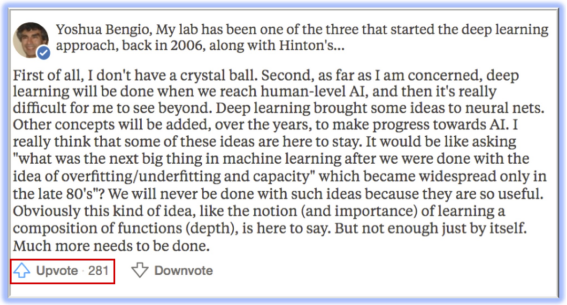
Community question answering (CQA) sites, such as Quora[1] and Zhihu[2], are Internet-based crowdsourcing services which enable users to post questions and seek answers from other users. The success of CQA has attracted much research attention in recent years [14, 26, 28, 31, 34, 38–41]. Most users would prefer to put forward open questions without strict standard answers in CQA sites. By this means, the criterion of answer quality is difficult to be unified in CQA systems. For example, the answers would be various from different users for the question "*What languages should I learn before starting to learn of AI?*". Thus, the voting mechanism is implemented in CQA sites, where the answer quality is indicated by the votes from the whole online communities. However, the answers elected by communities have low reliability due to the scarce votes in real applications. We take statistics in Quora and draw the correlation on the number of votes per question (#V per Q) and the number of answers per question (#A per Q) in Figure 1. As shown in the figure, most questions with a few answers receive less than 10 votes correspondingly. Therefore, automatic answer selection, which aims to rank the answers in descending order of quality, is a crucial task in CQA.

Many approaches have been proposed for answer selection, most of which tackle this problem as a text matching task. Early approaches are based on the form of exact string matches for n-grams,

[1]https://www.quora.com/
[2]https://www.zhihu.com/



(a) An common answer of the question.



(b) The best answer of the question.

Figure 2: Example of user expertise. Users with high expertise usually provide high-quality answers.

as such they fail to detect similar meaning conveyed by synonymous words [6, 7, 12, 19, 20]. Some approaches focus on constructing features from textual clues [36, 46], such as lexical and syntactic features. However, these methods has limited performance as the high computation cost for capturing context features. Recently, deep architectures are proposed to learn distributed representations for words and sentences. These approaches encode the questions and answers from a high-dimensional and sparse representation into low-dimensional and continuous dense vectors according to comprehensive context efficiently, e.g. convolutional neural networks (CNN) or recurrent neural networks (RNN). Based on the learned representations, semantic similarity [31, 33, 38] or matching [17, 24, 34, 37] functions can be learned from an end-to-end learning strategy.

However, traditional solutions of answer selection on QA systems cannot be directly applied in CQA as these methods ignore the influence of user expertise accepted by the community. In CQA systems, the answer rank is highly correlated with the expertise of users. We take an example of question in Quora, showing the difference between two answers whose users have different expertise in community. The open question and two corresponding answers are described in Figure 2(a) and 2(b). In the example, the two users give the answers from different aspects on "*What is the next big thing in machine learning after we are done with deep learning?*". The two answers are both highly relative to the question. However, the answer by Yoshua Bengio receives 281 votes which is much higher than the other one. The phenomenon is approved in [5, 49]. According to the literatures, the main reason is that Yoshua Bengio is a well-known expert in the community of deep learning and machine learning, and thus his answer tends to be voted by others.

However, user expertise is still questionable as its definition and quantitation during the progress of answer selection. Thus, we propose a user-expertise specific learning framework to model the dynamics of answer selection in CQA. The goal of our work tries to learn user expertise of each user by latent factors so as to better capture the rank of answers. Firstly, our proposed method defines latent user vectors in terms of answer ranking, explicitly modeling the relevance between question-user pair. A novel score function is introduced to evaluate both the relevance on question-answer and corresponding question-user pairs accepted by communities. Then, we introduce latent user vectors into the representation learning of answer, capturing the implicit topic interests in learned user vectors. Our proposed method encodes latent user vectors into the answer representation by hierarchical attention mechanisms within Long-Short Term Memory (LSTM) [16] network, which is more informative to user-specific interests. Experimental results on two datasets from real world CQA sites demonstrate that our proposed method outperforms other state-of-the-art approaches for answer selection. Moreover, the extensive experimental results prove that the learned user vectors can indicate the authority accepted by communities and capture the user interests in topic level, showing the great potentials on understanding user expertise in CQA.

Our contributions in this paper are summarized as follows:

- We propose a novel score function to directly evaluate the comprehensive relevance on both question-answer and question-user pairs, explicitly modeling the influence of user expertise in community acceptance.
- Our proposed model encodes latent user expertise into answer representation by hierarchical attention mechanisms, capturing implicit topic interests of each user.
- We conduct extensive experiments on two real-world datasets to verify the effectiveness of our model and compare it to other state-of-the-art approaches for answer selection in CQA. Moreover, the user representations learned by our proposed method provide us a quantitative way to understand topic-related user expertise in CQA. Besides, our implementation codes are open to public[3].

The remainder of this paper is organized as follows. Section 2 reviews the prior work related to answer selection. Section 3 formally defines the problem of answer selection. Section 4 introduces our proposed model in details. Section 5 presents the experimental results on two real-world datasets. Finally we conclude our work in Section 6.

2 RELATED WORK

In this section, we first review the prior work in community question answering. Then, we introduce the attention-based neural networks.

[3]https://github.com/Sunshine1007472173/UEAN

2.1 Community Question Answering

Early works in information retrieval, including language model (LM) [6, 7, 12, 19], and Okapi model (Okapi) [20], can also be used for answer selection. These approaches are purely based on the form of exact string matches for n-grams, as such they fail to detect similar meaning conveyed by synonymous words. A retrieval model was proposed in [44] that combined a translation-based language model for the question part with a query likelihood approach for the answer part. The authors formulated answer sentence selection as a semantic matching problem with a latent word-alignment structure in [46]. The authors studied the use of syntactic and semantic structures obtained with shallow and deeper syntactic parsers for the answer passage reranking task in [36].

With the rapid development of distributed word representations [18, 23, 27] and deep neural networks, many approaches have been proposed for answer selection. Convolutional neural network (CNN) was introduced by [17] to model the sentence matching in question answering. In [33], the authors demonstrated a unified architecture that trains a convolutional neural network together with a multi-layer perceptron to rank question-answer pairs. The authors uses a stacked bidirectional LSTM network to sequentially read words from question and answer sentences, and then outputs their relevance score [38]. Grid-wise similarity matrices within neural architectures is also adopted to model the complicated matching relations between question and answer [24, 34, 37]. The neural tensor network architecture was proposed to model the diverse relationships between question and answer in [31, 35].

However, most of the existing works merely model the semantic relevance between question-answer pairs and ignore the main characteristics in CQA, i.e., **expertise of users**. Semantic relevance and user expertise are jointly considered in our model. Furthermore, our model encodes user expertise information in the answer representation by attention mechanisms to focus on crucial segments of answers hierarchically. As a result, more important answer content and the expertise of users is captured. Recently, the authors proposed an asymmetric multi-faceted ranking network learning framework considering the answerers' authority in [48]. Nevertheless, they just model the user expertise in the score function and do not consider the influence of user expertise to the answer.

2.2 Attention-based Neural Networks

The attention mechanism has shown good performance in many tasks [2, 3, 8–11, 15, 25, 29, 32, 42, 43]. Specifically in machine translation [3, 25], given a sentence in the original language, the attention mechanism aims to adjust the weights of input words to predict the words in the target language. In [45], the authors proposed hierarchical attention networks for document classification. However, the answer selection task in CQA is different from the language translation and the document classification tasks. Moreover, our way of introducing user-aware hierarchical attentions is not explored in [3, 25, 45]. Recently, the authors in [47] proposed to first learn representations of question and answer by neural network architectures and then apply *one-level* attentive pooling network on an interaction tensor. Our approach differs in that our attention mechanisms are used for constructing the answer representation directly rather than for post-processing an interaction tensor. Moreover, our model jointly capturing the user expertise and semantic relevance, while [47] just focus on semantic matching.

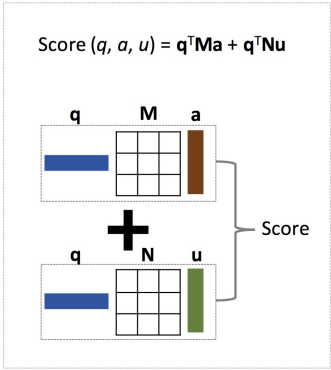


Figure 3: The score function of our model, jointly exploiting the semantic relevance between question-answer pairs and the user expertise to the given question.

3 PROBLEM FORMULATION

Before diving into detail of our model, we first describe symbols used in the answer selection problem. Given a question q, a set of its candidate answers $A_q = \{a_{q,1}, a_{q,2}, \ldots, a_{q,N}\}$ and users who gave the answers $U = \{u_1, u_2, \ldots, u_N\}$, answer selection aims to rank the answers in A_q in descending order of quality. We let $(s^a_1, s^a_2, \ldots, s^a_{N_a})$ represent the sequence of sentences in an answer a. Meanwhile, the word sequences in a sentence s is denoted as $(w^s_1, w^s_2, \ldots, w^s_{N_s})$. For ease of presentation, scalars, vectors and matrices are denoted by lower-case letters, boldface lower-case letters and boldface capital letters in this paper.

Since a good answer should well address a question semantically, the answer selection problem can be defined into an interactive process between a question-answer pair. In this way, the score function can be formalized as follows:

$$s(q_i, a_{q_i,j}) = \sigma(f(q_i), g(a_{q_i,j})), \tag{1}$$

where the function f and g map the question q_i and answer $a_{q_i,j}$ into a common vector space. The function σ measures matching degree between each question-answer pair. Generally, the score function σ could be in the form of cosine similarity, bilinear function or tensors [17, 31, 47]. For briefly describing the models, we use vector $\mathbf{q}_i = f(q_i)$ and $\mathbf{a}_{q_i,j} = g(a_{q_i,j})$ as the representations of mapping results in the rest of the paper.

For the question q_i, given two answers $a_{q_i,j}$ and $a_{q_i,m}$ where answer $a_{q_i,j}$ has a higher quality than $a_{q_i,m}$, we aim to learn the score function that the following inequality holds for every tuple $(q_i, a_{q_i,j}, a_{q_i,m})$: $s(q_i, a_{q_i,j}) > s(q_i, a_{q_i,m})$. Then, we can rank the answers for a question in descending order of quality according to the score function.

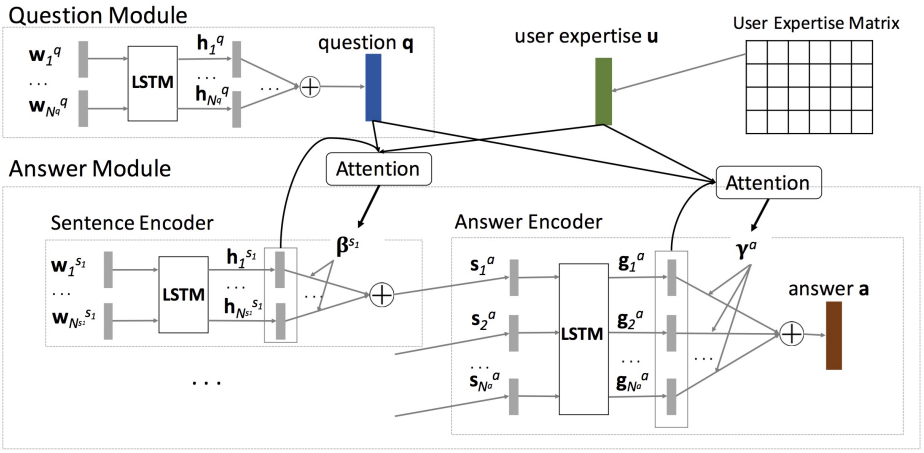


Figure 4: The Overview of our framework. Our model encodes user expertise in the answer representation, contributing to capture the implicit topic interests in learned user vectors.

4 UEAN FOR ANSWER SELECTION

In this section, we present User-Expertise aware Attention Networks (UEAN) model in detail whose framework is shown in Figure 4. We first explain our specific-designed user-expertise aware matching function in Section 4.1, explicitly modeling the influence of user expertise in community acceptance. Then, we discuss how to obtain answer and question representation in Section 4.2 and 4.3 respectively. Finally, we elaborate the learning process of our model in Section 4.4.

4.1 User Expertise Aware Matching Function

Most of the existing works merely model the semantic relevance between question-answer pairs and ignore the expertise of users when modeling the answer selection in CQA. Besides the semantic relevance, users with high expertise contribute to achieve acceptance from the community. Thus, we design a novel score function jointly capturing the semantic relevance and the expertise of users (shown in Figure 3). We divide our matching function into two parts: the semantic matching part and expertise part.

Let a tuple $(q_i, a_{q_i,j}, u_k)$ denote an answer $a_{q_i,j}$ provided by u_k for question q_i. The score function for a tuple $(q_i, a_{q_i,j}, u_k)$ is designed as follows:

$$s(q_i, a_{q_i,j}, u_k) = \mathbf{q}_i^T \mathbf{M} \mathbf{a}_{q_i,j} + \mathbf{q}_i^T \mathbf{N} \mathbf{u}_k. \tag{2}$$

where $\mathbf{q}_i, \mathbf{a}_{q_i,j}$, and $\mathbf{u}_k$ is the representation of the question q_i, answer $a_{q_i,j}$, and user expertise u_k respectively. The representation of answer $a_{q_i,j}$ and the question q_i are generated from answer and question in Section 4.2 and 4.3. The representation of user expertise $\mathbf{u}_k$ is a latent vector learned during training. In Eq. (2), $\mathbf{q}_i^T \mathbf{M} \mathbf{a}_{q_i,j}$ represents the semantic matching score between answer $a_{q_i,j}$ and question q_i, where $\mathbf{M}$ is a matrix to transform the question representation into the space of answer representation. $\mathbf{q}_i^T \mathbf{N} \mathbf{u}_k$ represents the expertise score of user u_k to the question q_i, where $\mathbf{N}$ is a matrix to transform the question representation into the representation space of user expertise. In this way, our matching function takes both the interactions of question-user pair and question-answer pair into consideration.

4.2 User Expertise Aware Answer Module

Answer module aims at generating representations for input answers. For the purpose of capturing the implicit topic interests in learned user vectors, we introduce latent user vectors into the representation learning of answers. We encode user expertise in the answer representation by hierarchical attention mechanisms within LSTM networks, contributing to capture content more informative to user-specific interests.

4.2.1 ***Sentence encoder.*** For the t-th sentence s_t with N_{s_t} words in an answer, we first represent each word $w_j^{s_t}$ as a K_w-dimensional vector $\mathbf{w}_j^{s_t}$ by pre-trained word embeddings [30]. We then adopt the widely exploited LSTM enabling to deal with variable-length sequence input, to encode a sentence into a fixed-length vector. We utilize the LSTM in [21] to take each word vector $\mathbf{w}_j^{s_t}$ as input and updates the corresponding hidden state $\mathbf{h}_j^{s_t}$ as follows:

$$\mathbf{h}_j^{s_t} = LSTM(\mathbf{h}_{j-1}^{s_t}, \mathbf{w}_j^{s_t}). \tag{3}$$

In Eq. (3), the hidden vector $\mathbf{h}_j^{s_t}$ is learned from the previous hidden states and the j-th input word.

In a sentence, some words are relevant to the semantic information of user expertise. For instance, the word "Java" or "Python" can capture the expertise of the user who is skilled in "Programming Language". The word "LSTM" or "CNN" can capture the the expertise of the user who is skilled in "Deep Learning". Hence, we introduce the latent user expertise into the representation learning of answer sentence to capture the implicit topic interests in learned user vectors. Meanwhile, we also introduce the question information to capture the words relevant to the question.

We encode the user expertise in answer representation by attention mechanisms [3]. We first concatenate the user expertise vector $\mathbf{u}$, question vector $\mathbf{q}$, and hidden vector $\mathbf{h}_j^t$ of answer word. Then we select *tanh* as the activation function and calculate the weights

by a multi-layer perceptron (MLP) [4]. Finally, the attention weight $\beta_j^{s_t}$ is generated as follows:

$$e_j^{s_t} = \mathbf{v}_w^T \tanh(\mathbf{W}_w[\mathbf{u}; \mathbf{q}; \mathbf{h}_j^{s_t}] + \mathbf{b}_w),$$
$$\beta_j^{s_t} = softmax(e_j^{s_t}), \quad (4)$$

where $softmax(x_i) = \frac{x_i}{\sum_j x_j}$ is taken to normalize the weights, and $\mathbf{W}_w$, $\mathbf{b}_w$ and $\mathbf{v}_w$ are parameters in word-level attention that to be learned. Then the sentence representation $\mathbf{s}_t$ can be calculated based on the weights obtained from Eq. (4) and the hidden states from $\mathbf{h}_1^{s_t}$ to $\mathbf{h}_{N_{s_t}}^{s_t}$ as follows:

$$\mathbf{s}_t = \sum_{j=1}^{N_{s_t}} \beta_j^{s_t} \mathbf{h}_j^{s_t}. \quad (5)$$

4.2.2 ***Answer encoder.*** Assume that an answer a is comprised of a sequence of N_a sentences, i.e. $(s_1^a, \ldots, s_{N_a}^a)$, each of which is encoded as a K_s-dimensional vector $\mathbf{s}_t^a$ by Eq. (5). As the sentences of an answer can be regarded as a sequence with variable length, we can obtain the answer representation in a similar way using the LSTM at sentence-level. The hidden state $\mathbf{g}_t^a$ is calculated as follows:

$$\mathbf{g}_t^a = LSTM(\mathbf{g}_{t-1}^a, \mathbf{s}_t^a), \quad (6)$$

where the updating of hidden state is depended on the sentence vector $\mathbf{s}_t^a$ and the hidden state $\mathbf{g}_{t-1}^a$ in each iteration.

Similar to the words, some sentences can capture the implicit topic interests of learned user vectors. Hence, we encode user expertise and question content in the answer representation by a sentence-level attention mechanism. In this way, more informative sentences will be emphasized and less informative sentences will be weakened. Thus, we can estimate weights from attention layer according to the latent factor of users and the representation of question. Then each attention weight γ_t^a for the sentence s_t^a is calculated through attention mechanism by

$$e_t^a = \mathbf{v}_s^T \tanh(\mathbf{W}_s[\mathbf{u}; \mathbf{q}; \mathbf{g}_t^a] + \mathbf{b}_s),$$
$$\gamma_t^a = softmax(e_t^a), \quad (7)$$

where $\mathbf{W}_s$, $\mathbf{b}_s$ and $\mathbf{v}_s$ are parameters in sentence-level attention that to be learned. The overall answer representation vector $\mathbf{a}$ is calculated by aggregating all hidden vectors from sentences weighted by their corresponding attention weights as

$$\mathbf{a} = \sum_{t=1}^{N_a} \gamma_t^a \mathbf{g}_t^a. \quad (8)$$

4.3 Question Module

For a question q with a sequence of N_q words, we first represent each word w_i^q as a K_w-dimensional vector $\mathbf{w}_i^q$ by pre-trained word embeddings [30]. We then employ the LSTM by taking each word vector $\mathbf{w}_i^q$ as input and updating the corresponding hidden state $\mathbf{h}_i^q$ as follows:

$$\mathbf{h}_i^q = LSTM(\mathbf{h}_{i-1}^q, \mathbf{w}_i^q). \quad (9)$$

We observe that each word contributes differently to the representation of a question. For example, in a question "*Where is the theater?*", "where" and "theater" play more critical roles in summarizing this question. Therefore, we employ the attention mechanism to learn the question representation by focusing on more informative words. The attention weight α_i^q is based on the hidden state $\mathbf{h}_i^q$ of word w_i^q. Afterwards, the question representation $\mathbf{q}$ is generated as the weighted sum of hidden vectors from words according to their corresponding importance as follows:

$$\alpha_i^q = softmax(\mathbf{m}^T \mathbf{h}_i^q), \quad (10)$$

$$\mathbf{q} = \sum_{i=1}^{N_q} \alpha_i^q \mathbf{h}_i^q, \quad (11)$$

where $\mathbf{m}$ is a parameter to be learned.

4.4 Learning Process

We employ a discriminative training strategy with a large margin objective to train our model. Given each pair of tuples $(q_i, a_{q_i,j}, u_k)$ and $(q_i, a_{q_i,m}, u_n)$ related to question q_i in training set, the relevance score of each answer can be calculated by the matching function in Eq. (2). Suppose that the quality of answer $a_{q_i,m}$ is higher than the quality of answer $a_{q_i,j}$, we aim to maximize the following ranking-based loss under observed ranking order as follows,

$$L = \sum_{(i,j,k) \prec (i,m,n)} \max(0, c + s(q_i, a_{q_i,m}, u_n) - s(q_i, a_{q_i,j}, u_k)),$$

where $0 < c < 1$ is a slack variable. During the training process, question and answer share the pre-trained word embeddings. All the weight parameters are randomly initialized and trained by the back propagation of training loss. Besides, the ground truth of ranking order is inferred by votes in real CQA sites.

5 EXPERIMENT

In this section, we conduct experiments to evaluate the effectiveness of our model and compare it with several state-of-the-art methods for answer selection, showing that our proposed model outperforms other state-of-the-art approaches. Moreover, we investigate the learned representation of user expertise, showing the potentials on understanding user expertise in CQA.

5.1 Experimental Setups

5.1.1 ***Datasets.*** To demonstrate the effectiveness of our approach in real application, the following two datasets are adopted in our experiment.

- **Quora dataset**. We collected a large dataset from Quora which is one of the most popular CQA sites. We randomly crawled 800,772 questions and 3,523,485 answers posted from January 2016 and July 2017. The votes and users related to these answers are also collected. For statistical reliability, we remove the questions if the maximum number of votes received by the following answers is smaller than 10. We also remove questions with answers less than 2. Finally, our dataset contains 76,158 questions, 467,681 answers, 31,802,631 votes, and 46,470 users.
- **Stack Overflow dataset**. The dataset is available from StackOverflow[4] in public, containing 214,803 questions and 586,725

[4] https://stackoverflow.com/

answers. We preprocess the dataset following the similar strategy used in Quora dataset. After preprocessing, the dataset contains 19,944 questions, 67,011 answers and 6,340 users.

The two datasets are both randomly split into three subsets without overlapping for the purpose of training, validation and testing in the following experiments. The proportion of training, validation and testing dataset is equal to 80%, 10% and 10%. We also vary the size of training dataset (40%, 60% and 80%) in order to validate the robustness of our proposed methods in different experimental setting.

5.1.2 ***Evaluation Metric.*** We evaluate the performance of our model using three widely-used ranking metrics, which are Precision@1, Mean Reciprocal Rank (MRR), and normalized Discounted Cumulative Gain (nDCG). The three metrics are formulated as follows,

- *Precision@1* takes into account the position of relevant answer at top 1, calculated by

$$Precision@1 = \frac{|\{q \in Q | rank'_{best} = 1\}|}{|Q|},$$

  where $|Q|$ is the number of questions in test dataset and $rank'_{best}$ is the predicted position of the best answer given by algorithm.
- *MRR* is the average of the reciprocal ranks corresponding to the most relevant answer in questions, calculated by

$$MRR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{rank'^{i}_{best}},$$

  where $rank'^{i}_{best}$ is the position of the best answer in the i-th question given by algorithm.
- *NDCG* is a measure of ranking quality. As to question q, NDCG is calculated by

$$nDCG = \frac{DCG}{IDCG},$$
$$\text{where } DCG = rel_1 + \sum_{i=2}^{|A_q|} \frac{rel_i}{\log_2 i},$$

  where IDCG is ideal discounted cumulative gain, $|A_q|$ is the number of answers in question q and rel_i is the relevance between question and answer at position i. We report the average of all the nDCG values in all test questions.

The larger results received by the above evaluation metrics means the better performance of the algorithms in answer selection.

5.1.3 ***Comparative Methods.*** We compare our model to 6 typical state-of-the-art methods which are briefly described below.

- **Okapi BM25** [20] is a bag-of-words retrieval model used in information retrieval. We use the model to rank candidate answers based on the question terms appearing in each answer.
- **TransLM** [44] is a retrieval model that combines a translation-based language model for the question part with a query likelihood approach for the answer part. We use TransLM to calculate the similarity between every question-answer pair and rank candidate answers based on the similarity.
- **Doc2Vec** [23] is an unsupervised algorithm that learns fixed-length feature representations from variable-length pieces of texts. We first use Doc2Vec to learn representations of questions and answers. The matching score of a question-answer pair is calculated with a MLP with the representations of the question and answer as input.
- **CNTN** [31] is a convolutional neural tensor network architecture to encode the question and answer in semantic space and model their interactions with a tensor layer.
- **AI-CNN** [47] first learns representations of questions and answers by CNN and then applies one-level attentive pooling network on an interaction tensor of a question-answer pair.
- **AMRNL** [48] learns representations of questions and answers by deep recurrent neural networks and ranks the answers by semantic relevance and the answers' authority.

Table 1: The Precision@1 of all methods on Quora dataset with different settings of training data size.

Method	Precision@1 (%)		
	40%	60%	80%
Okapi BM25	26.71	26.71	26.71
TransLM	29.10	29.63	30.24
Doc2Vec	31.74	32.40	32.42
CNTN	35.95	36.02	36.17
AI-CNN	33.95	34.52	35.03
AMRNL	59.10	59.19	61.14
OURS	**61.63**	**63.38**	**64.54**

Table 2: The MRR of all methods on Quora dataset with different settings of training data size.

Method	MRR (%)		
	40%	60%	80%
Okapi BM25	51.23	51.23	51.23
TransLM	52.23	52.61	53.12
Doc2Vec	53.82	53.99	54.13
CNTN	57.63	57.84	58.20
AI-CNN	56.38	56.63	57.07
AMRNL	74.23	74.48	75.64
OURS	**76.47**	**77.63**	**78.29**

5.1.4 ***Parameter settings.*** Okapi BM25, TransLM, Doc2Vec and AMRNL are implemented according to the source code provided by authors. The baselines of CNTN and AI-CNN are carefully implemented with hyperparameter tuning according to the original papers. In our proposed method, we use GloVe [30] pre-trained word embeddings. The dimension of word embedding is 50. The questions and answers are tokenized and lemmatized using NLTK[5]. Tokens that did not appear in the pre-trained word embeddings are initialized randomly. The dimension of LSTM network embedding in our experiment is 128 and the dimension of parameter $\mathbf{v}$ for

[5]https://www.nltk.org/

Table 3: The nDCG of all methods on Quora dataset with different settings of training data size.

Method	nDCG (%)		
	40%	60%	80%
Okapi BM25	76.37	76.37	76.37
TransLM	76.11	76.35	76.72
Doc2Vec	76.40	76.55	76.57
CNTN	79.80	80.18	80.66
AI-CNN	79.22	79.36	79.60
AMRNL	89.52	89.88	90.22
OURS	**91.31**	**91.91**	**92.04**

attention mechanism in Eq. (4) and Eq. (7) is 64. Besides, the dimension of user representation is 128. We set the margin $c = 0.5$ in the ranking loss. At last, we choose Adam [22] as the optimization strategy to update parameters with initial learning rate 0.001.

5.2 Prediction Performance on Answer Selection

In this section, we compare the performance of our model with the chosen state-of-the-art methods in order to validate the effectiveness of our proposed method in predicting the rank of best answers. Table 1, 2 and 3 present the experimental results on Quora dataset in terms of Precision@1, MRR and nDCG respectively. Table 4, 5 and 6 present the experimental results on Stack Overflow dataset. We summarize our observations as follows:

- Methods based on distributed representations all perform better than Okapi BM25. It proves that word embeddings can capture more semantic information than traditional bag-of-words approaches.
- CNTN and AI-CNN outperform Doc2Vec and TransLM, indicating that it is effective to model fine-granularity interaction between a pair of question and answer.
- Approaches with consideration of user expertise, i.e., AMRNL and our proposed method, perform better than these methods that merely consider semantic relevance, demonstrating that modeling the influence of user expertise is helpful for answer selection in CQA.
- In addition, our proposed method is better than AMRNL, proving the effectiveness of modeling user expertise in both explicit and implicit ways for answer selection.
- At last, our proposed method significantly outperform the baselines in all cases. The results show that our proposed method is well suitable to answer selection problem in CQA.

In order to validate the effectiveness of explicitly modeling the relevance of question-user pair in our model, we introduce variants of our model (with user expertise and without) and compare the performance. Table 7 presents the experimental results.It is observed that our model with user expertise significantly outperform our model without user expertise. Furthermore, we conduct case studies to analyze the advantages of our proposed model in answer selection. We find out that short answers prefer to receive lower relevant score with merely modeling question-answer relevance.

Table 4: The Precision@1 of all methods on Stack Overflow dataset with different settings of training data size.

Method	Precision@1 (%)		
	40%	60%	80%
Okapi BM25	39.72	39.72	39.72
TransLM	42.78	43.38	44.03
Doc2Vec	40.04	41.20	41.36
CNTN	46.70	47.57	47.92
AI-CNN	44.98	45.99	47.62
AMRNL	50.15	50.41	50.71
OURS	**53.90**	**55.46**	**56.56**

Table 5: The MRR of all methods on Stack Overflow dataset with different settings of training data size.

Method	MRR (%)		
	40%	60%	80%
Okapi BM25	65.67	65.67	65.67
TransLM	66.96	66.98	67.52
Doc2Vec	66.30	66.47	66.48
CNTN	69.45	69.87	69.98
AI-CNN	67.91	68.69	69.50
AMRNL	71.08	71.16	71.22
OURS	**72.73**	**73.89**	**74.22**

Table 6: The nDCG of all methods on Stack Overflow dataset with different settings of training data size.

Method	nDCG (%)		
	40%	60%	80%
Okapi BM25	89.36	89.36	89.36
TransLM	90.28	90.71	90.82
Doc2Vec	88.95	89.52	89.59
CNTN	91.41	91.76	91.77
AI-CNN	90.58	91.13	91.13
AMRNL	91.55	91.95	92.10
OURS	**91.70**	**92.34**	**92.79**

It is because that short answers usually have less semantic information than long answers. However, *short answer would be the best answer accepted by communities*. In the question "What are the similarities between GANs and reinforcement learning?" on Quora[6], the communities vote the shortest answer to be the best answer as the responser is "Yoshua Bengio". In CQA, the acceptance of community pays an important role in answer selection. The explicit modeling on the relevance of question-user pair can well model the scenario when the defined "user expertise" pays more important role in answer selection on CQA systems.

Next we will investigate the learning mechanism of user expertise in detail and give both qualitative and quantitative analyses on learned user expertise.

[6]https://www.quora.com/What-are-the-similarities-between-GANs-and-reinforcement-learning

Table 7: The performance (%) of our model (with user expertise and without) on datasets with 80% of the data for training. ("UE" - user expertise.)

Method	Quora dataset			Stack Overflow dataset		
	P@1	MRR	nDCG	P@1	MRR	nDCG
Without UE	36.84	58.02	79.84	48.58	69.99	91.17
With UE	**64.54**	**78.29**	**92.04**	**56.56**	**74.22**	**92.79**

5.3 What is Learned in User Vectors

In this section, we give both qualitative and quantitative analyses on learned user embedding vectors and illustrate the potentials of learned user expertise on presenting community acceptance and topic interests of users.

Authority-correlated user expertise. Average number of votes is positively related to authority accepted by the community in CQA platform, which is an important indicator for user expertise in long term. Therefore, we use the average number of votes to validate whether the learned user representations has positive correlation to the authority in practice. We randomly choose 3,000 users in Quora and plot their embeddings in Figure 5(a) whose dimensionality is reduced by PCA [13] from the original high dimensional space (128) to a visualization space (2). Each user is colored by their true authority, calculated by their average votes per question in Quora. In the figure, the users with different authority can be well separated by the user representations. We can differentiate three clusters with high, median and low expertise from the sampled users. We randomly choose several users at each cluster and show their profile. The blue/purple block on the right side can be regarded as the community with high reputation, where users with high votes and many followers are located, such as Balaji Viswanathan, Richard Muller, Jake Williams, Alon Ami and Dave Consiglio. The green block on the middle is the community with median authority, such as John L Ware and Eric Reid are located. The red block on the left side is the community with low expertise, where the users with few followers and zero vote are located. According to the experimental results, the learned user vectors have potentials on presenting authority. Furthermore, we reduce the dimensionality of user vectors from 128 to 1 by PCA and plot it with the average votes of users in Figure 5(b). It is observed that the user vectors and votes are positively correlated in linear and the correlation coefficient is about 0.27. Both the experimental results indicate that the learned user representation can positively reflect the users' authority in quantitative ways.

Topic-sensitive user expertise. The CQA systems would label users when the history of user behaviors is quite enough to be expertise in certain fields. In this way, we can use the systematic tags on users to verify whether the learned user expertise contains the topic interests. Firstly, we choose three random cases to qualitatively analyze how the relevance exists between the learned user expertise and topics. The cosine similarity is introduced to measure the difference between user vector and question representation, and we use the cosine similarity to explore the most similar questions of users. Then we choose 30 most similar questions of each user and extract the keywords in questions to validate if the keywords are relative to the user tags. The experimental results are shown in Figure 6. It is observed that the extracted keywords of each user are correlated to the user tags, indicating the topic-sensitive user representations learned by our proposed method. Besides, we randomly choose 1,000 users who are labeled on "programming" and "relationships" respectively. Then we apply K-means [1] to cluster the sampled users as $K = 2$ based on the learned user representations. The accuracy of clusters is up to 67%, giving the evidence of the topic interests contained in the learned user representations.

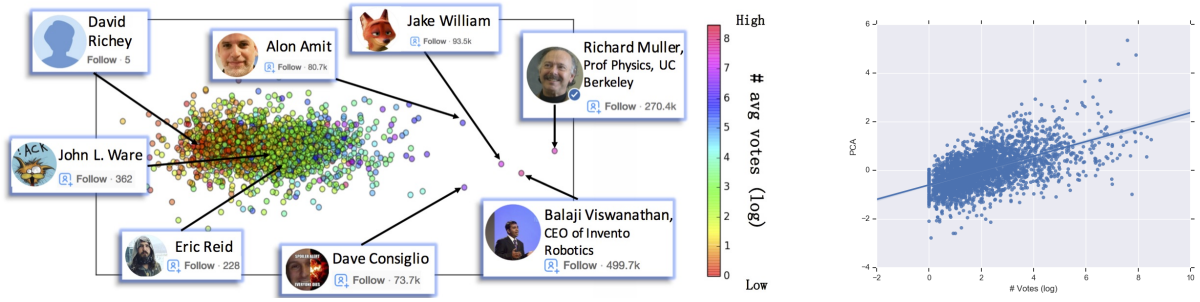
5.4 Effect of User Expertise in Answer Selection of Different Domains

In this section, we analyze the effect of user expertise in answer selection of different domains. We take four domains as examples from testing dataset, e.g. Movies, Experiences in Life, Physics, Police and Law Enforcement. We show the performance of our model (with user expertise and without) and the increase of metric by user expertise in Table 8. We notice that the increase of metric is more significant in domains like "Physics", "Police and Law Enforcement" than "Movies", "Experiences in Life". The experimental results demonstrate that user expertise plays a more important role in areas which need more domain knowledge. For instance, question "What is the most difficult thing you have ever watched?" in "Experiences in Life" can be answered by every user, but question "How far away from the nuclear plant is safe in case of nuclear meltdown?" in "Physics" is more professional.

5.5 Effect of User Expertise in Answer Representation Learning

In order to analyze the effect of user expertise in answer representation learning, we take an example and draw heat maps of sentence and word weights to demonstrate the difference between the setups with/without user expertise in answer representation learning (Figure 7). We pick up the question that "*What languages should I learn before starting to learn of AI?*" and its three answers as the example. There are many sentences in each answer and we only visualize the key sentences and their weights due to space limitation. The demonstration is shown in Figure 7, where Figure 7(a) and 7(d) depict weights by the learning mechanism with user expertise, Figure 7(b) and 7(e) describe the weights by the learning mechanism without user expertise respectively.

In the examples, we notice that both Figure 7(a) and 7(d) highlight the word "python" with the highest weight in the paragraph. Meanwhile, the word "AI" is given the highest weight in the paragraph shown in the Figure 7(b) and 7(e). Moreover, the sentences with python and programming are highlighted by the results from answer representation learning with user expertise, and the sentences with AI and ML are emphasized by the results from answer representation learning without user expertise. It is clear that the highlighted results from answer representation learning with user expertise are more relative to the question "What language should I learn before starting to learn of AI?". According to the results, we can observe that semantic relevance is less important than user expertise in such scenario. The answer would hardly point to the question by direct semantic relevance. In the examples, the better results from answer representation learning with user expertise are because of that the user_1 and user_2 have "computer programming"



(a) Visualization of user vectors. The blue/purple color means the higher expertise achieved in Quora, the green color means the median expertise, while the red color refers to the lower expertise.

(b) The correlation between learned user expertise and the true expertise accepted by communities.

Figure 5: The visualization of learned user representations and their correlation to the true expertise accepted by communities (sampled by 3000 users in Quora).

(a) (b) (c)

Figure 6: Examples of topic interests in learned user vectors.

Table 8: The performance (%) of our model (with user expertise and without) in different domains on Quora dataset with 80% of the data for training. ("UE" - user expertise. "Increase" - the increase of metric by modeling user expertise.)

Domain	Movies			Experiences in Life			Physics			Police and Law Enforcement		
	P@1	MRR	nDCG	P@1	MRR	nDCG	P@1	MRR	nDCG	P@1	MRR	nDCG
Without UE	35.00	54.56	74.88	35.71	58.99	82.65	29.36	53.78	78.48	27.66	49.28	72.21
With UE	52.50	69.76	87.47	58.33	74.91	91.67	72.48	83.55	94.56	70.21	81.74	93.73
Increase	50.00	27.86	16.82	63.34	26.99	10.91	146.87	55.36	20.49	153.85	65.86	29.82

tag in user profile (see Figure 7(c) and 7(f)), and the embedded user expertise in answer representation learning can help to emphasize on the programming-related words. The answer representation learning with user expertise can both capture the semantic and specialized relevance in answering, improving the emphasis on more important words in contents.

Furthermore, we conduct experiments on Quora and Stack Overflow dataset to validate the effectiveness of different attention mechanism for answer representation learning with user expertise at different attention levels. The experimental results are recorded in Table 9. With 60% training dataset, the implementation of attention mechanism with user expertise at hierarchical level achieves 0.6338, 0.7763 and 0.9191 in Quora dataset, resulted by P@1, MRR and nDCG respectively. The experimental results are better than the results from the implementation of attention mechanism with user expertise at word and sentence levels in Quora dataset (achieving 0.6205 and 0.6291 in P@1, 0.7655 and 0.7732 in MRR and 0.9110 and 0.9178 in nDCG). Moreover, all the attention mechanisms with user expertise at different semantic levels achieve better performance than the evaluation results from the attention mechanism without user expertise. It is interested that the attention mechanism with user expertise at sentence level achieve better performance than the implementation of attention mechanism with user expertise at word level. It may indicate that the implementation of attention

(a) Heat map with user expertise of answer$_1$. (b) Heat map without user expertise of answer$_1$. (c) Profile of user$_1$.

(d) Heat map with user expertise of answer$_2$. (e) Heat map without user expertise of answer$_2$. (f) Profile of user$_2$.

Figure 7: Visualization of the effect of encoding user expertise in answer representation. Orange denotes the sentence weight and blue denotes the word weight. The segments with higher weights are endowed with darker color. (a) and (d) are heat maps of answers with encoding user expertise in answer representation. (b) and (e) are heat maps of answers without encoding user expertise in answer representation. (c) and (f) are the user profiles about their expertise respectively. Answer representation learning with user expertise is able to emphasize sentences and words on personalized knowledgement. Answer representation learning without user expertise tends to focus on sentences and words which match the question semantically, i.e., the word "*AI*".

mechanism with user expertise on higher level is more effective than the implementation on lower level. The same conclusion can be reproduced in the experimental results from Stack Overflow dataset. Overall, the experimental results demonstrate the effectiveness on the implementation of attention mechanism with user expertise at both word and sentence levels. The answer representation learning with user expertise can emphasize more on the important words by semantic and specialized relevance in answering, improving the rank prediction of answer selection in CQA.

6 CONCLUSION

Answer selection is the key problem when scarce votes are provided in one question on CQA platform. However, the ranking of answers is simply tackled as a text mathcing problem in existing works, ignoring the influence of community. In this paper, we propose a user-expertise specific learning framework to model the dynamics of answer selection in CQA. Our model learns user expertise of each user by latent factors, better capturing the rank of answers. A novel score function is introduced to evaluate the relevance on question-answer pair and user expertise in question-user pairs accepted by

Table 9: The performance (%) of answer representation learning with user expertise at different attention levels, i.e., word, sentence and hierarchical levels, on datasets with 60% of the data for training. ("No" - the vanilla LSTM without attention mechanism. "Word" - employ attentions only at word-level and utilize the vanilla outputs of sentence level LSTM as the answer representation. "Sentence" - employ attentions only at sentence-level and use the vanilla outputs from word level LSTM. "Hierarchical" - employ attentions at both levels.)

Method	Quora dataset			Stack Overflow dataset		
	P@1	MRR	nDCG	P@1	MRR	nDCG
No	61.66	76.34	91.07	52.48	72.48	91.67
Word	62.05	76.55	91.10	54.34	73.42	92.16
Sentence	62.91	77.32	91.78	54.49	73.62	92.21
Hierarchical	**63.38**	**77.63**	**91.91**	**55.46**	**73.89**	**92.34**

communities. Moreover, we introduce latent user vectors into the representation learning of answering, capturing the implicit topic interests in learned user vectors. The experiments show that our model outperforms all baselines in answer selection task. Furthermore, we analyze the user representations learned by our model. The extensive experimental results prove that the learned user vectors can indicate the authority accepted by communities and capture the user interests in topic level, provide us a quantitative way to understand user expertise in CQA.

ACKNOWLEDGMENTS

This work was funded by the National Natural Science Foundation of China under Grant Numbers 61602439, 61425016, 91746301, and the National Key Research and Development Program of China under grant number 2016QY03D0504. Wentao Ouyang is also funded by the CAS Pioneer Hundred Talents Program under Grant Number 2920164120.

REFERENCES

[1] David Arthur and Sergei Vassilvitskii. 2007. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 1027–1035.

[2] Jimmy Ba, Volodymyr Mnih, and Koray Kavukcuoglu. 2015. Multiple object recognition with visual attention. In *International Conference on Learning Representations (ICLR)*.

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations (ICLR)*.

[4] Yoshua Bengio et al. 2009. Learning deep architectures for AI. *Foundations and trends® in Machine Learning* 2, 1 (2009), 1–127.

[5] Mohamed Bouguessa, Benoît Dumoulin, and Shengrui Wang. 2008. Identifying authoritative actors in question-answering forums: the case of yahoo! answers. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 866–874.

[6] Xin Cao, Gao Cong, Bin Cui, and Christian S Jensen. 2010. A generalized framework of exploring category information for question retrieval in community question answer archives. In *Proceedings of the 19th international conference on World wide web*. ACM, 201–210.

[7] Xin Cao, Gao Cong, Bin Cui, Christian Søndergaard Jensen, and Ce Zhang. 2009. The use of categorization information in language models for question retrieval. In *Proceedings of the 18th ACM conference on Information and knowledge management*. ACM, 265–274.

[8] Huimin Chen, Maosong Sun, Cunchao Tu, Yankai Lin, and Zhiyuan Liu. 2016. Neural sentiment classification with user and product attention. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. 1650–1659.

[9] Edward Choi, Mohammad Taha Bahadori, Le Song, Walter F Stewart, and Jimeng Sun. 2017. GRAM: Graph-based attention model for healthcare representation learning. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 787–795.

[10] Jan Chorowski, Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. End-to-end continuous speech recognition using attention-based recurrent NN: first results. *arXiv preprint arXiv:1412.1602* (2014).

[11] Jan K Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio. 2015. Attention-based models for speech recognition. In *Advances in neural information processing systems*. 577–585.

[12] Huizhong Duan, Yunbo Cao, Chin-Yew Lin, and Yong Yu. 2008. Searching questions by identifying question topic and question focus. *Proceedings of ACL-08: HLT* (2008), 156–164.

[13] George H Dunteman. 1989. *Principal components analysis*. Number 69. Sage.

[14] Hanyin Fang, Fei Wu, Zhou Zhao, Xinyu Duan, Yueting Zhuang, and Martin Ester. 2016. Community-based question answering via heterogeneous social network learning. In *AAAI*.

[15] 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*. 1693–1701.

[16] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[17] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. In *Advances in neural information processing systems*. 2042–2050.

[18] Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*. Association for Computational Linguistics, 873–882.

[19] Jiwoon Jeon, W Bruce Croft, and Joon Ho Lee. 2005. Finding semantically similar questions based on their answers. In *SIGIR*. ACM, 617–618.

[20] Jiwoon Jeon, W Bruce Croft, and Joon Ho Lee. 2005. Finding similar questions in large question and answer archives. In *Proceedings of the 14th ACM international conference on Information and knowledge management*. ACM, 84–90.

[21] Rafal Jozefowicz, Wojciech Zaremba, and Ilya Sutskever. 2015. An empirical exploration of recurrent network architectures. In *International Conference on Machine Learning*. 2342–2350.

[22] Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[23] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*. 1188–1196.

[24] Pengfei Liu, Xipeng Qiu, Jifan Chen, and Xuanjing Huang. 2016. Deep fusion LSTMs for text semantic matching. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 1034–1043.

[25] Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 1412–1421.

[26] Shanshan Lyu, Wentao Ouyang, Huawei Shen, and Xueqi Cheng. 2017. Truth Discovery by Claim and Source Embedding. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 2183–2186.

[27] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[28] Preslav Nakov, Doris Hoogeveen, Lluís Màrquez, Alessandro Moschitti, Hamdy Mubarak, Timothy Baldwin, and Karin Verspoor. 2017. SemEval-2017 task 3: Community question answering. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. 27–48.

[29] Hyeonseob Nam, Jung-Woo Ha, and Jeonghee Kim. 2017. Dual Attention Networks for Multimodal Reasoning and Matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 299–307.

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

[31] Xipeng Qiu and Xuanjing Huang. 2015. Convolutional Neural Tensor Network Architecture for Community-Based Question Answering. In *IJCAI*. 1305–1311.

[32] Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A Neural Attention Model for Abstractive Sentence Summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 379–389.

[33] Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to rank short text pairs with convolutional deep neural networks. In *SIGIR*. ACM, 373–382.

[34] Yikang Shen, Wenge Rong, Zhiwei Sun, Yuanxin Ouyang, and Zhang Xiong. 2015. Question/Answer Matching for CQA System via Combining Lexical and Sequential Information. In *AAAI*. 275–281.

[35] Yi Tay, Minh C Phan, Luu Anh Tuan, and Siu Cheung Hui. 2017. Learning to rank question answer pairs with holographic dual LSTM architecture. In *SIGIR*. ACM.

[36] Kateryna Tymoshenko and Alessandro Moschitti. 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*. ACM, 1451–1460.

[37] Shengxian Wan, Yanyan Lan, Jiafeng Guo, Jun Xu, Liang Pang, and Xueqi Cheng. 2016. A Deep Architecture for Semantic Matching with Multiple Positional Sentence Representations. In *AAAI*, Vol. 16. 2835–2841.

[38] Di Wang and Eric Nyberg. 2015. A long short-term memory model for answer sentence selection in question answering. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, Vol. 2. 707–712.

[39] Gang Wang, Konark Gill, Manish Mohanlal, Haitao Zheng, and Ben Y Zhao. 2013. Wisdom in the social crowd: an analysis of quora. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 1341–1352.

[40] Xiaochi Wei, Heyan Huang, Chin-Yew Lin, Xin Xin, Xianling Mao, and Shangguang Wang. 2015. Re-Ranking Voting-Based Answers by Discarding User Behavior Biases. In *IJCAI*. 2380–2386.

[41] Yu Wu, Wei Wu, Zhoujun Li, and Ming Zhou. 2016. Improving Recommendation of Tail Tags for Questions in Community Question Answering. In *AAAI*.

[42] Zhen Wu, Xin-Yu Dai, Cunyan Yin, Shujian Huang, and Jiajun Chen. 2018. Improving Review Representations with User Attention and Product Attention for Sentiment Classification. *arXiv preprint arXiv:1801.07861* (2018).

[43] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*. 2048–2057.

[44] Xiaobing Xue, Jiwoon Jeon, and W Bruce Croft. 2008. Retrieval models for question and answer archives. In *SIGIR*. ACM, 475–482.

[45] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 1480–1489.

[46] Scott Wen-tau Yih, Ming-Wei Chang, Chris Meek, and Andrzej Pastusiak. 2013. Question answering using enhanced lexical semantic models. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL)*. 1744–1753.

[47] Xiaodong Zhang, Sujian Li, Lei Sha, and Houfeng Wang. 2017. Attentive Interactive Neural Networks for Answer Selection in Community Question Answering. In *AAAI*. 3525–3531.

[48] Zhou Zhao, Hanqing Lu, Vincent W Zheng, Deng Cai, Xiaofei He, and Yueting Zhuang. 2017. Community-Based Question Answering via Asymmetric Multi-Faceted Ranking Network Learning. In *AAAI*. 3532–3539.

[49] Zhou Zhao, Lijun Zhang, Xiaofei He, and Wilfred Ng. 2015. Expert finding for question answering via graph regularized matrix completion. *IEEE Transactions on Knowledge and Data Engineering* 27, 4 (2015), 993–1004.