ITC 3/52
Information Technology and Control
Vol. 52 / No. 3 / 2023
pp. 789-805
DOI 10.5755/j01.itc.52.3.33449

Deep Semantic Understanding and Sequence Relevance Learning for Question Routing in Community Question Answering

Received 2023/02/20 | Accepted after revision 2023/05/25

HOW TO CITE: Li, H., Li, J., Li, G., Wang, C., Cao, W., Chen, Z. (2023). Deep Semantic Understanding and Sequence Relevance Learning for Question Routing in Community Question Answering. *Information Technology and Control*, 52(3), 789-805. https://doi.org/10.5755/j01.itc.52.3.33449

Deep Semantic Understanding and Sequence Relevance Learning for Question Routing in Community Question Answering

Hong Li
School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China; e-mail: lilyhong420@hbut.edu.cn
College of Computer Science and Technology, Hubei University of Technology, Wuhan 430068, China

Jianjun Li
School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China; e-mail: li0405@hust.edu.cn

Guohui Li
School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China; e-mail: guohuili@hust.edu.cn

Chunzhi Wang
College of Computer Science and Technology, Hubei University of Technology, Wuhan 430068, China; e-mail: chunzhiwang@163.com

Wenjun Cao
Hubei Key Laboratory of Solar Energy Efficient Utilization and Energy Storage Operation Control, Hubei University of Technology, Wuhan 430068, China; e-mail: 102010322@hbut.edu.cn

Zixuan Chen
College of Computer Science and Technology, Hubei University of Technology, Wuhan 430068, China; e-mail: 2110311226@hbut.edu.cn

Corresponding author: li0405@hust.edu.cn

Question routing (QR) aims to route newly submitted questions to the potential experts most likely to provide answers. Many previous works formalize the question routing task as a text matching and ranking problem between questions and user profiles, focusing on text representation and semantic similarity computation. However, these works often fail to extract matching features efficiently and lack deep contextual textual understanding. Moreover, we argue that in addition to the semantic similarity between terms, the interactive relationship between question sequences and user profile sequences also plays an important role in matching. In this paper, we proposed two BERT-based models called QR-BERT$_{rep}$ and QR-tBERT$_{int}$ to address these issues from different perspectives. QR-BERT$_{rep}$ is a representation-based feature ensemble model in which we integrated a weighted sum of BERT layer outputs as an extra feature into a Siamese deep matching network, aiming to address the non-context-aware word embedding and limited semantic understanding. QR-tBERT$_{int}$ is an interaction-based model that explores the interactive relationships between sequences as well as the semantic similarity of terms through a topic-enhanced BERT model. Specifically, it fuses a short-text-friendly topic model to capture corpus-level semantic information. Experimental results on real-world data demonstrate that QR-BERT$_{rep}$ significantly outperforms other traditional representation-based models. Meanwhile, QR-tBERT$_{int}$ exceeds QR-BERT$_{rep}$ and QR-BERT$_{int}$ with a maximum increase of 17.26% and 11.52% in MAP, respectively, showing that combining global topic information and exploring interactive relationships between sequences is quite effective for question routing tasks.

KEYWORDS: Community question answering, BERT, Question routing, Contextual language embedding, Topic model.

1. Introduction

Community Question Answering (CQA) is an online service that enables users to post questions and obtain answers from other users, which has proven to be a very effective way of sharing knowledge and experience. Recently, with the increasing number of questions that cannot be answered in time, much concern has arisen over the efficiency and answer quality of CQA services [28]. Therefore, routing the newly posted question to the right user for quick and accurate answer is an important strategy to maintain user engagement and the vibrancy of the CQA platform. Modeling the similarities and relevance between users' profiles and questions is critical in the textual content-based question routing approaches. User modeling is generally based on the user's historical answer record, and all the answers provided by the user in the past are collected to form the user's profile. When we treat users' profiles as documents and questions as queries, the question routing task can be viewed as a classic text matching and ranking problem [24]. Finding and sorting documents that match the query is equivalent to finding the best expert who can answer the question.

Text understanding plays a vital role in matching and ranking. Traditional methods mainly include language models and topic models, which heavily rely on lexical overlap or word co-occurrence [4, 38]. However, these methods have very limited text understanding ability and are usually unable to capture deep and complex semantics efficiently, leading to unsatisfactory results. Recently, along with the rapid development of distributed word embedding and deep learning, neural ranking networks have been applied to question routing tasks [25, 36, 1, 31, 15]. Most of these neural models are representation-based that first turn the question and user profile into vectors using word embedding (Word2Vec [21], GloVe [24]), and then use a typical neural network (e.g., CNNs or RNNs) to extract patterns and construct dense meaningful feature vectors separately. Finally, the semantic similarity is calculated for further ranking.

Although existing deep neural representation-based methods have achieved promising performance, they have several shortcomings: First, traditional word embedding is static, which means it fails to distinguish the term's meaning in different scenarios [9]. Second, feature extractors are mainly based on CNNs or LSTMs, however, CNN-based methods [36, 37] usually have a limited receptive field to capture long-distance depen-

dencies, and LSTM-based methods [7] are difficult to parallelize. Third, they mainly focus on matching the semantic similarity level while ignoring sequence interactions. In fact, text matching is very complicated in CQA, the relationships between the two sequences are also important factors in matching. For the specific task of question routing, matching questions and users can be seen as matching questions and answers since the user's profile is composed of answers. In general, questions and answers not only share terms and topics, but are often logically connected, and exploring the semantic similarity of terms alone is not enough to achieve good matching performance.

Recently, the pre-trained bidirectional contextual language model BERT [8] has brought unprecedented performance gains in text understanding tasks, and we expect to adopt it to improve the performance of question routing in community Q&A as well. However, there are some special challenges that need to be addressed. First, the low average participation rate of users and the relatively short length of questions in Q&A communities lead to severe data sparsity problems, which result in insufficient textual content for user modeling and question modeling. According to studies [18, 20], most answers come from very few users, in Quora, a well-known community Q&A website, 90% of the questions got less than 10 answers, more than 30% of users did not answer any questions, and only 16.74% of users answered more than 4 times [30]. Second, modeling the similarities and relevance between question-user pairs is challenging due to a large number of domain-specific terms and the fact that there is little direct lexical overlap between question sequences and user profile sequences.

Based on the above analysis, in this paper, we propose two novel models from different perspectives to address the question routing task: a tag-word topic-enhanced interaction-based method called QR-tBERT$_{int}$ and a combined representation-based model called QR-BERT$_{rep}$. Specifically, in QR-tBERT$_{int}$, we take questions and user profiles as query-document pairs, and they are concatenated into a longer sequence as input. By fine-tuning BERT on our task-specific dataset, contextual semantic learning and question-profile pair relationship exploration are integrated into a unified model. In addition, we innovatively incorporate a tag-word topic model to handle domain-specific terms in QR-tBERT$_{int}$. And in another model QR-BERT$_{rep}$, we incorporate the contextualized embeddings learned from the pre-trained model into an existing Siamese deep learning-based matching model to enhance the semantic understanding.

The main contributions of this paper are as follows:

- We propose two novel BERT-based deep neural models to solve the question routing task from different perspectives: representation-focused and interaction-focused. Specifically, we adopt different strategies for modeling similarities between questions and user profiles in different models and propose to explore the interactive relationships between question sequences and user profile sequences. Our research can provide experiences and references for other domain-specific text understanding tasks in CQA.
- We incorporate a corpus-level tag-word topic model to learn the global matching feature and topic semantic information in QR-tBERT$_{int}$ to help handle domain-specific cases, and we combine the contextualized embeddings with the traditional word embeddings to construct more meaningful representations in QR-BERT$_{rep}$ to enhance the matching performance.
- We conducted a detailed experimental study using a real-world dataset from Stack Overflow. We evaluated the performance of our two methods and compared them with several baseline approaches. The experimental results show that our approaches yield satisfactory performance and significantly outperform the baseline approaches.

The rest of this paper is organized as follows. In Section 2, we review the related work. Section 3 details the proposed two models. In Section 4, we present our dataset and experimental setting. Finally, we present our experimental results and discussion in Section 5. Section 6 concludes this paper.

2. Related Work

2.1. Question Routing

Question routing is a fundamental task that has been widely studied in social communities and is also referred to as expert finding or expert recommendation in many studies. Statistical language models [4, 3, 40] and topic models [39, 32] have played an essential role for a long time. Although they can solve question rout-

ing tasks, they all lack deep textual understanding and fail to capture complex semantic features.

Recently, deep learning technologies have brought a revolutionary way to solve question routing issues with more concise and efficient architectures [25, 36, 1, 31, 15]. A method to directly apply deep neural networks to question routing was proposed by Azzam et al. [1] based on Deep Semantic Similarity Model [11]. In this model, questions and the users' profiles are mapped to a low-dimensional semantic space through a deep neural network, and the similarity score is computed using the cosine similarity function. Later, CNNs and LSTMs were gradually introduced into the NLP field, bringing significant improvements. Wang et al. [31] designed a variant of CNN architecture to capture the semantics of the text for expert recommendation tasks. Chen et al. [37] described an effective convolutional neural network with three filters of different sizes to learn the representations of questions and answers to identify experts. In another work [36], the LSTM(long-short term memory) [10] network has been employed to learn the question embedding instead of CNN in the Quora dataset. More recently, Li et al. [15] proposed to combine the embedding of the question raiser learned by a heterogeneous information network representation algorithm with the embedding of the question content to enhance the characterization of the question.

However, the performance of these representation-based deep learning methods often suffers from data sparsity and inefficient feature extraction. In addition, they encode the question and the user's profile as two separate sequences, facing the risk that the interaction between the text sequences could be ignored. Different from the above studies, in this paper, we not only incorporate contextualized embeddings obtained from an efficient self-attentive mechanism-based feature extractor into traditional word embeddings to improve representation performance but also propose to explore the interactive relationships between question and user sequences in addition to focusing on text semantic learning.

2.2. Pre-training Language Models

The pre-training language model aims to learn word embeddings or representations with prior semantic knowledge by performing pre-training tasks from a large number of unlabeled corpora. Researchers from the Google company released an exciting bidirectional language representation model BERT [8], aiming to solve the unidirectional constraints in GPT [26] and extend the model to multi-layer bidirectional Transformer [29] blocks, achieving the best performance in many NLP tasks such as machine translation, text classification, and question retrieval.

Many recent works have also introduced BERT to solve question routing or expert recommendation tasks and achieved relatively good results beyond the traditional approaches. However, these works mainly use pre-trained BERT models as encoders and feature extractors, and the potential of BERT models is not fully exploited, which has a limited effect on improving the overall question routing performance. For example, Zhang et al. [35] conducted a Temporal Context-aware Question Routing model in which BERT is only used to encode the question content. Peinelt et al. [23] proposed a semantic enhancement approach that combines BERT embeddings with LDA-based topics for semantic similarity prediction on the Quora dataset, which achieves better performance than vanilla BERT. However, the above approach cannot be directly applied to our task due to the need to learn programming-specific terms in our dataset and the fact that the length of the questions is too short which leads to difficulty in topic derivation. Therefore, we use a more targeted topic model, the tag-word topic model, to learn specific domain terms and provide corpus-level semantic information.

3. Our Proposed Models for Question Routing

In this section, we will describe two BERT-based models named QR-BERT$_{rep}$ and QR-tBERT$_{int}$ to address the question routing task. In brief, QR-BERT$_{rep}$ designs a feature ensemble method in which each text sequence goes through the pre-trained BERT network separately. Then, the outputs by the last four highest-level Transformer layers corresponding to each input token in different positions are extracted as additional textual features and incorporated into a Siamese neural matching network. Compared to QR-BERT$_{rep}$, QR-tBERT$_{int}$ concatenates two text sequences to a longer sequence and adopts a more flexible way by fine-tuning to learn the interaction between questions and users from the beginning. In

Figure 1

The overall framework of the topic-enhanced interaction-based model QR-tBERT$_{int}$

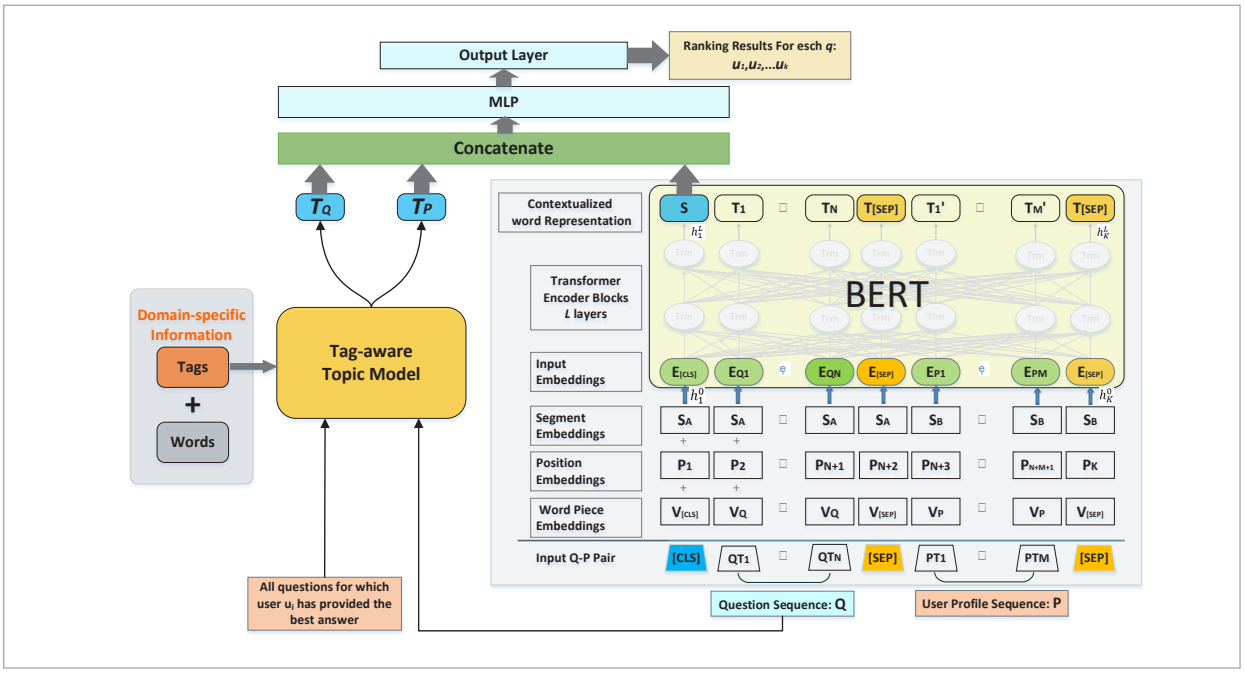
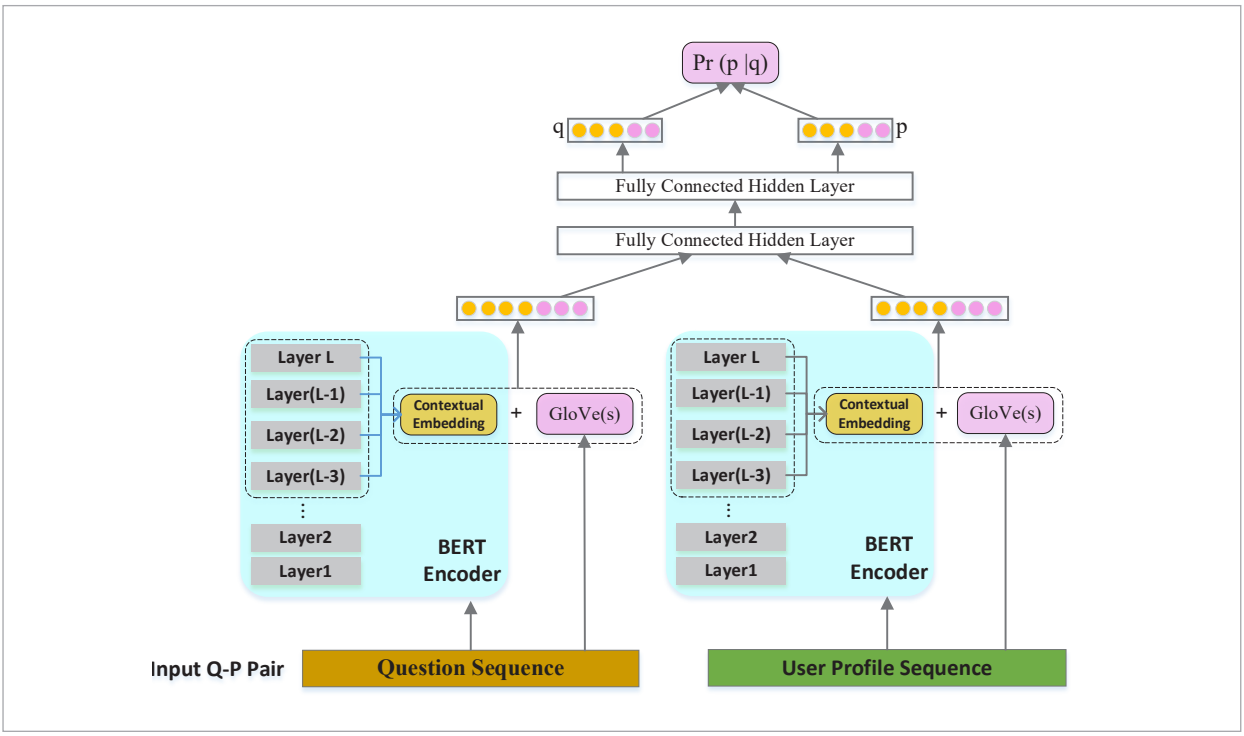


Figure 2

The overall framework of the contextualized representation-based model QR-BERT$_{rep}$



addition, it combines a tag-word topic model to enhance the semantic understanding and handle the domain-specific terms.

3.1. Community Question Answering: Stack Overflow

First, we introduce the necessary background of Stack Overflow, including the main characteristics and the question-answering mechanism. There are several important components in a Q&A thread:

1. ***Questions*** are the central element of Stack Overflow, which includes the title, body, and tags. The life cycle of a question begins with an open state in which any user can provide an answer to the question. Subsequently, when the questioner chooses the best answer, or other users choose the best answer by voting, the question is considered solved and no more answers are received.
2. ***Answers*** are provided by different users and can be voted on by other users. The more votes an answer receives, the higher the approval and the better the quality. In addition to the best answer, all other answers are sorted in a thread in descending order of votes.
3. ***Best (Accepted) Answer*** is selected by the questioner or selected by other users in which the answer received the largest number of votes. Each question has only one best answer. It is at the top of the answer list.
4. ***Users*** include questioners and respondents whose basic information is displayed under questions or answers related to them.
5. ***Tags*** are assigned by the questioner and represent the knowledge area relevant to the question. A CQA website has accumulated an enormous number of question-answer threads that provide a plethora of textual information for us to explore.

3.2. Problem Statement

As detailed above, a CQA dataset is built upon the static archive of the CQA website, which preserves all the question-answer threads accumulated over time. Let Q be a question set $Q = \{q_1, q_2, ..., q_n\}$ (n is the number of questions) and U be an answerer set $U = \{u_1, u_2, ..., u_j\}$ (j is the number of users). For each answerer u_j, a document p_i is a combination of all the best answers provided by u_j, and p_i is referred to as the profile of that answerer.

Using the above notations, we formalize the question routing task as a text match and ranking problem and define it as follows: Given a newly posted question q, let a set of $C \in U$ be a candidate set $C = \{c_1, c_2, ..., c_k\}$ (k is the number of candidates), and let a set of candidate profiles $P = \{p_1, p_2, ..., p_k\}$ We need to rank users in C and route q to the highly ranked users, who are most suitable to answer the question q with the required knowledge. An essential part of this task is learning the match patterns and capturing the relationships between question q and the profile of the candidate $p_i \in P$. Specifically, in our work, we need to estimate a score r_i of how relevant a candidate user's p_i is to a newly posted question q or we need to calculate a probability Pr of how likely a user's profile p_i is given the question q.

3.3. Tag-word Topic Enhanced Interaction-based Approach: QR-tBERTint

In this section, we design a topic-enhanced BERT-based model named QR-tBERT$_{\text{int}}$ for the question routing task, the overall framework of which is illustrated in Figure 1. To learn the structural and textual relevance, we assemble the question sequence and user profile sequence into a longer text sequence and encode it with the stacked Transformer blocks. We take the special embedding of the first token in the last layer as the fusion relevance representation of the combined sequences. Meanwhile, a tag-word topic model TTM [6] is adopted to derive high-quality topics by building tag-word co-occurrence on the corpus level, thereby helping to enhance the domain-specific knowledge understanding and relieve the data sparsity problem. Based on previous studies which successfully combined the corpus-level topic with neural networks [23, 33, 22], we take the concatenation of the sequence pair fusion representation S obtaining form BERT and sequence-level topic representations T_Q and T_P obtaining from the tag-word topic model as the final representation F and send it to the next task-specific ranking layers:

$$F = Concat(S, T_Q, T_P), \quad (1)$$

where $S \in \mathbb{R}^e$, $T_Q \in \mathbb{R}^K$, and $T_P \in \mathbb{R}^K$. K denotes the number of topics.

Corpus-level topic representation module. Topic models have been shown to provide additional information to enhance text understanding and matching in earlier feature engineering-based models, and are

particularly effective for dealing with domain-specific terms [38, 32]. In recent years, many deep neural methods have achieved impressive performance on many NLP tasks such as domain recommendation [33], semantic analysis [23, 22], and machine translation [5] by combining topic models. However, extracting topics from relatively short texts in CQA and constructing an efficient fusion model to combine the corpus-level topic information is very challenging in the question routing task. According to the characteristics of Q&A threads described in Section 3.1, we use the unsupervised learning tag-word topic model [6] to derive corpus-level topical representations $T_Q \in \mathbb{R}^K$ and $T_P \in \mathbb{R}^K$ for questions and users.

First, we construct a tag-word pool by combining a word and a tag. For example, a question with two tags (t_1, t_2) and three words (w_1, w_2, w_3) will generate six tag-words in the form of {(t_1, w_1), (t_1, w_2), (t_1, w_3), (t_2, w_1), (t_2, w_2), (t_2, w_3)}. Second, by considering the entire corpus as a mixture of topics whose distribution over topics comes from a Dirichlet allocation with priors α and assuming there are K topics whose distribution over tags and words are sampled from Dirichlet allocations with prior γ and β. The joint probability of a tag-word can be formulated as:

$$P(t_i, w_j) = \sum_k P(k)P(t_i|k)\,P(w_j|k) = \sum_k \theta_k\, \varphi_{i|k}\phi_{j|k}, \quad (2)$$

where $k \in [1, K]$ denotes a topic, $\theta \sim Dir(\alpha)$ denotes a topic distribution for the whole collection, $\varphi_k \sim Dir(\gamma)$ denotes a topic-specific tag distribution, and $\phi_k \sim Dir(\beta)$ denotes a topic-specific word distribution. We take the tag-word as the basic unit of the topic model and aggregate all tag-words from the whole corpus for training. After obtaining the tag-word topic model, the question sequence and the user profile sequence are passed to the topic model to infer topic-level embedding per sequence:

$$T_Q = TTM([QT_1, QT_2, \ldots, QT_N]) \in \mathbb{R}^K, \quad (3)$$

$$T_P = TTM([PT_1, PT_2, \ldots, PT_M]) \in \mathbb{R}^K, \quad (4)$$

where K denotes the number of topics, [QT_1, QT_2, ..., QT_N] denotes the questions sequence, and [PT_1, PT_2, ..., PT_M] denotes the user profile sequence.

Sequence Pair Fusion Representation Module based on BERT. We take the linear concatenation of the question tokens and the profile tokens as input. For a given token v_i, its final input representation $h_i^0 \in \mathbb{R}^e$ is constructed by summing word piece embedding v_i, the segment embedding s_i, and position embedding p_i of the same dimension:

$$h_i^0 = v_i + p_i + s_i. \quad (5)$$

We truncate the question to have at most 64 tokens, and the user profile is truncated to ensure that the concatenation of the question, profile, and separator token has a maximum length of 512 tokens.

As illustrated in Figure 1, when the input sequence passes through the multi-layer Transformer encoder blocks, the tokens of the entire sequence are read by each Transformer encoder at once and learned by the self-attention mechanism that results in contextualized embeddings at different positions in each layer. Specifically, each Transformer layer *Trm* has two sublayers: MultiSelf and PFFN. The former is a multi-head self-attention mechanism-based network, while the latter is a position-wise fully connected feed-forward network which consists of two linear transformations with Gaussian Error Linear Unit (GELU) activation in between. In our task, we believe that the multi-head attention mechanism can capture different types of token relationships by using different attention matrices, and the self-attention mechanism spans the entire sequence of questions and user profiles so that question-profile interactions are learned. The specific formulations of these two sublayers can be found in the [29] and will not be repeated here. Based on the two sublayers, a residual connection around each of the two sub-layers and dropouts to the output of each sub-layer is applied [2]. In summary, the hidden representation of each layer is shown as follows:

$$H^l = \mathrm{Trm}(H^{l-1}), \quad (6)$$

$$\mathrm{Trm}(H^{l-1}) = \\ = \mathrm{LayerNorm}\left(M^{l-1} + \mathrm{Dropout}\left(PFFN(M^{l-1})\right)\right), \quad (7)$$

$$M^{l-1} = \\ = \mathrm{LayerNorm}\left(H^{l-1} + \mathrm{Dropout}\left(MultiSelf(H^{l-1})\right)\right). \quad (8)$$

After L layers that hierarchically exchange information across all positions in the previous layer, we obtain the final output H^L for all tokens of the input

sequence. And next, we should perform candidate answerer ranking by using this feature embedding and then route the newly posted question to the candidate answerers that are ranked higher.

Output. We use the softmax function to obtain the probability of the profile being relevant:

$$r_i = \text{softmax}(W^f \cdot F + b^f), \tag{9}$$

where $W^f \in \mathbb{R}^{C\times(e+2k)}$ is the learnable projection matrix and b^f is bias terms. C is the number of labels. We compute this probability for each candidate independently and obtain the final list of experts (profiles) by ranking them with respect to these probabilities.

Fine-tune and Training. We use a $\text{BERT}_{\text{BASE}}$ model (hidden size of 768, 12 Transformer blocks, and 12 self-attention heads) as a binary classification model. We start training from it and fine-tune it to our question routing task using the cross-entropy loss. Specifically, limited by the size of our training corpus, we freeze the weights of the first few layers of the pre-trained network during fine-tuning. We believe that a well-trained BERT model fully incorporates context information at each token position and contains sentence relationship information in the embedding of [CLS]. The loss is shown in Equation (10).

$$L = -\left(\sum_{i\in I_+} \log(r_i) + \sum_{i\in I_-} \log(1-r_i) \right), \tag{10}$$

where r_i denotes the relevant score of the question and user, I_+ is the set of indexed answerers (positive label) and I_- is the set of indexed random non-answerers (negative label). The model is fine-tuned by minimizing the cross-entropy loss.

3.4. Contextual Representation-based Approach: QR-BERT_{rep}

Different from the above method that focuses on exploring interactions between sequences through BERT and incorporating tag-word topic models to enhance understanding of corpus-level semantic information, QR-BERT_{rep} incorporates the weighted sum of the outputs of different layers of BERT as an additional feature into a traditional Siamese deep matching model. By combining contextualized embeddings with word embeddings, the representations of question sequences and user profile sequences can imply richer semantic knowledge and patterns, helping to improve the expert discovery effect obtained by similarity computation. The overall framework of the contextual representation-based model QR-BERT_{rep} is shown in Figure 2.

BERT contextualized embedding. Instead of concatenating the question tokens and the profile tokens into a single sequence as input, in this method, the question tokens and the profile tokens are fed into the pre-trained $\text{BERT}_{\text{BASE}}$ model separately to obtain the contextualized embedding layer by layer.

Encoding layer. Since BERT generates L-layer hidden states for all BPE tokens in a sequence, and each hidden layer contains different features and information, we employ a weighted sum of these hidden states to obtain more delicate embedding. Specifically, we take the hidden states of the last four layers in BERT. Suppose a word w is tokenized to n BPE tokens $w = \{b_1, b_2, ..., b_n\}$, and h_i^l represents the token embedding in the l-th layer of BERT, $1 \le l \le L$, $1 \le i \le n$. Then, the contextualized embedding of word w, ConEM_w, is calculated as the weighted sum average of the embedding of the last four layers.

$$\text{ConEM}_w = \sum_{l=1}^{L} \delta_l \frac{\sum_{t=1}^{n} h_i^l}{n}, \tag{11}$$

where δ_l denotes the weight for each layer. Then, we concatenate the 300-*dim* GloVe embedding and contextualized embedding ConEM_w together to build a richer representation for each word. Therefore, the input vector for each word in the question sequence and profile sequence is $w = [\text{GloVe(w)}; \text{ConEM}_w]$.

Siamese neural ranking model. After encoding each word into a fixed-length fusion vector, we represent the question sequence and profile sequence by the fusion embeddings and feed them into a Siamese neural ranking model, which consists of two fully connected hidden layers with 300 nodes. This model is used to map word vectors to their semantic concept vectors for further similarity calculation. In detail, if we denote x as the input word vector, y as the output vector, h_i as the hidden layer vector, W_i as the i^{th} weight matrix, and b_i as the i^{th} bias term, the mathematical formulas for each layer are described as follows:

$$h_1 = f(W_1 x + b_1), \tag{12}$$

$$h_i = f(W_i h_{i-1} + b_i), \tag{13}$$

$$y = f(W_N h_{N-1} + b_N), \quad (14)$$

where the i value goes from the first hidden layer i = 2 to the output layer $i = N$, and we use the tanh as the activation function:

$$f(x) = \frac{1-e^{-2x}}{1+e^{-2x}}. \quad (15)$$

Output layer. The output layer consists of 128 nodes. We measure the semantic similarity between question q and profile document p as:

$$\text{Sim}(q,p) = \text{cosine}(y_q, y_p) = \frac{y_q^T y_p}{\|y_q\|\|y_p\|}, \quad (16)$$

where y_q and y_p are the concept vectors of the question and the user's profile, respectively. We apply the *softmax* function on the output to covert the similarity relevance score into a probability of the user's profile given the question as shown below:

$$\Pr(p|q) = \frac{e^{cosine(y_q,y_p)}}{\sum_{n=1}^{K} e^{consine(y_q,y_n)}}, \quad (17)$$

where K denotes the number of candidates to be ranked. We approximated K to be the list of the answerers, which includes the actual answerers and three randomly selected non-answerers. We detail the construction method of negative examples and positive examples for training in Section 4.1.

Training. In training, the model parameters are estimated to maximize the likelihood of positive answerers given the questions across the training set. Put another way, we need to minimize the loss function, as shown in Equation (18).

$$L(W_i, b_i) = -\log \prod_{i \in I_+} Pr(p_i|q), \quad (18)$$

where I_+ is the set of indexed answerers (positive labels).

4. Experiments

4.1. Dataset

We constructed our dataset from a Stack Overflow snapshot by following all conditions mentioned in [12] and [27]. To reduce the size of the dataset while exhibiting the same properties according to the original dataset, we use the 21 tags reported in [12] to create a subset that is mainly selected according to the following criteria. Each selected question is an archived question with an accepted answer (i.e., best answer), and it has at least 2 answers, and at least one of its tags matches the selected 21 specific tags. All questions are lowercase, and we only keep the questions with at least 2 words left after removing the stop words. The purpose of these selection operations is to filter out low-quality posts. As a result, the final subset contains 92,411 CQA sessions. According to the posted timestamp, the first 12 months of data are used as the training data, and the remaining data are used for testing. Therefore, the training and testing data do not overlap. There were 81,295 sessions in the training set and 11,116 sessions in the test set.

Given the need to predict the best answerer and the reality that only a few users are responsible for the vast majority of answers in CQA, three user sets D_x were constructed based on the number of answers X provided by users in the training set (X = 10, 15, and 20 in this work). As can be seen from Table 1, set D_{20} includes 2,977 users, indicating that these users provided at least 20 answers in this training set. Moreover, for each of the 8371 training questions, the questioner, the best answerer, and at least one other answerer are among these 2977 users. For the 517 test questions, they were routed to these 2977 users.

Table 1
The summary of three datasets

The Set Name	# of questions answered by user U	# of Users U	# of Training Questions Q_{Trn}	# of Test Questions Q_{Tst}
D_{10}	10	5,761	16,021	1,151
D_{15}	15	3,971	11,177	746
D_{20}	20	2,977	8,371	517

Ground Truth: The list of answerers in D_x who actually provided an answer to the test question is the ground truth in our experiment.

Creating Examples: To train the model, we need to create positive and negative examples. According to our collected data, a training set consists of threads (question, asker, best answer, best answerer, other answers, other answerers .....). Following [1], if user u_i is in the list of answerers of q (the list includes the best answerer and other answerers of one thread), we con-

sider (q, u_i) as a positive example; otherwise, we consider (q, u_j) as a negative example. We obtained 54,218 positive training pairs for D_{10}, 36,238 positive training pairs for D_{15}, and 26,354 positive training pairs for D_{20}. To train efficiently and reduce the training scale, we randomly select three non-answerers based on the NCE sampling strategy to construct the negative samples. The definitions of negative and positive examples are listed in Table 2.

Table 2
Negative and positive examples for training

Question-User Pair	Label/Class
(q, answerer$_1$)	Positive
(q, answerer$_2$)	Positive
...	Positive
(q, answerer$_n$)	Positive
(q, random-non-answerer$_1$)	Negative
(q, random-non-answerer$_2$)	Negative
(q, random-non-answerer$_3$)	Negative

4.2. Baseline Methods and Experimental Setting

Baseline Methods. To evaluate the performance of our proposed models, we use the following three different types of baselines for comparison: the traditional information retrieval model, topic-based model, and deep learning-based model.

1 Traditional IR model

TF-IDF: TF-IDF [27] is a standard measure of computing the importance and relevance of a word document based on the frequency of that word in the document and the inverse proportion of documents containing the word over the entire document corpus. For the question routing and expert finding task, we represent the posted question and user profile as vectors of their TF-IDF weights and then calculate the cosine similarity between each user profile and question vector.

2 Topic-based model

LDA: LDA [17] is a three-level hierarchical Bayesian model that has been widely applied to address the term mismatch problem in IR. It mainly relies on word co-occurrence relationships and takes semantic information into account. In our experiments, all questions answered by a user are concatenated to build the user profile. We use Gibbs-LDA++ [14] with topic size K=100 to conduct LDA training. We set the LDA hyper-parameters $\alpha = 0.5$ and $\beta = 0.1$, respectively.

MLQR: MLQR [6] is a multi-objective learning-to-rank approach in which a tag-word topic model was proposed and applied to address the question routing problem. In this experiment, we set the number of topics $K = 80$, $\alpha = 0.7$, $\beta = 0.01$, and $\gamma = 0.01$. Gibbs Sampling is run for 1000 iterations.

3 Deep learning-based model

QR-DSSM: QR-DSSM [1] is a typical deep neural Siamese Network based on DSSM [11] to capture the semantic similarity between the profiles of the candidates and the posted question. In our experiment, to facilitate subsequent comparisons, we use GloVe embedding to represent the sequence instead of using the word hash embedding method. The code blocks are removed from the dataset. The number of iterations of the neural network is 100, and the learning rate is 0.02.

CNN-based method: A CNN-based method [31] treats question routing as a classification problem and takes the best answerer of each question as a positive training example as well as the ground truth. We adopt the CNN-non-static [13] to capture the semantics of the text for best answerer prediction, which uses filter windows of 3, 4, and 5 with 100 feature maps each. The dropout rate is 0.5, and the mini-batch size is 50.

Experimental Setting. We use the English uncased BERT-Base model released by Google, which has 12 layers, 768 hidden states, and 12 heads. Models are implemented with TensorFlow using TPUs. Regarding the selection of hyperparameters, we fixed some empirically, such as choosing the Adam weight decay optimizer for the optimization with L2 weight decay of 0.01, $\beta 1 = 0.9$, and $\beta 2 = 0.999$. The dropout probability is always kept at 0.1. Some hyperparameters were set to different values during the training and were chosen according to their impact on performance. The initial learning rate and batch size are set to [1e-3, 2e-5, 1e-7] and [16, 32, 64], respectively. In the tag-word topic model, we set α=0.8, β=0.01, and γ=0.01. The number of topics varies from 20 to 90. In addition, since the randomness of the parameter initialization leads to different results each time, we averaged the results for 10 runs.

4.3. Evaluation Metrics

The evaluation criteria measure how well the system ranks the correct pair (q, answerer$_i$) against the other random candidates for the same question (q, random-non-answerer$_j$). Therefore, we adapt several standard metrics for expert finding and question routing to evaluate the performance as follows.

1 Precision at N (P@N): The precision at N reports the percentage of predicted positive users/experts observed at the top N retrieved results. In other words, it is the ratio of the number of positive users to the total number of candidates until N. For example, Precision@1(P@1) aims to compute the percentage of times the system ranks the correct answerers as the top item. More specifically, if our model returns 10 users for a given question, the relevant users are ranked at 1, 2, 4, 6, and 9. Then, the P@5 is 3/5 and the P@10 is 5/10 in this case.

2 Mean Reciprocal Rank (MRR): The MRR computes the inverse of the rank of the correct answerer among other answerers averaged for all queries. Alternatively, we can describe it as reflecting the average ranking of the actual answerer's first appearance in a given test set question. For a given query set Q, we use the following formula to calculate MRR.

$$MRR = \frac{1}{N}\sum_{j=1}^{N}\frac{1}{rank_j}, \tag{19}$$

where N is the number of queries and $rank_j$ is the position of the correct answerer.

3 Mean Average Precision (MAP): The MAP shows the overall retrieval quality score, which is the arithmetic mean of the average precision score for each test set question.

5. Experiment Results and Analysis

This section presents the effectiveness of our proposed models on question routing tasks comparing three types of different models over our dataset.

5.1. Performance Analysis of Our Proposed Model Compared to Baseline Models

The results are summarized in Tables 3-5. We can see that all BERT-based models perform much better than the existing traditional retrieval models and the recently proposed neural network-based models. In detail, several main observations can be concluded from these tables.

1 Topic-based models exhibit much better performance than the traditional information retrieval approaches. This finding suggests that semantic understanding is important in the question routing task of text-based analysis. Approaches that rely on lexical matching without any text semantics have significant limitations. Moreover, MLQR consistently performs better than the LDA model, which

Table 3
Comparison of different methods for question routing (X=10)

Model Type	Model Name	D_{10}			
		P@5	P@10	MRR	MAP
Traditional IR	TF-IDF	0.0251	0.0159	0.0558	0.0281
Topic-based	LDA	0.0358	0.0230	0.0820	0.0386
	MLQR	0.0657	0.0411	0.1522	0.0699
Deep-learning	QR-DSSM	0.1033	0.0568	0.2035	0.0957
	CNN-based	0.0927	0.0701	0.2244	0.1017
Proposed	**QR-BERT$_{rep}$**	0.1629	0.1225	0.3223	0.1881
	QR-BERT$_{int}$	0.1771	0.1208	0.3168	0.2019
	QR-tBERT$_{int}$	**0.1935**	**0.1487**	**0.3592**	**0.2407**

Table 4
Comparison of different methods for question routing (X=15)

Model Type	Model Name	D_{15}			
		P@5	P@10	MRR	MAP
Traditional IR	TF-IDF	0.0315	0.0200	0.0645	0.0321
Topic-based	LDA	0.0431	0.0268	0.0895	0.0439
	MLQR	0.0813	0.0509	0.1776	0.0798
Deep-learning	QR-DSSM	0.1078	0.0662	0.2291	0.1093
	CNN-based	0.1173	0.0813	0.2616	0.1193
Proposed	**QR-BERT$_{rep}$**	0.1863	0.1477	0.3556	0.2312
	QR-BERT$_{int}$	0.2111	0.1724	0.3561	0.2296
	QR-tBERT$_{int}$	**0.2324**	**0.2112**	**0.3905**	**0.2723**

Table 5
Comparison of different methods for question routing (X=20)

Model Type	Model Name	D_{20}			
		P@5	P@10	MRR	MAP
Traditional IR	TF-IDF	0.0312	0.0195	0.0687	0.0353
Topic-based	LDA	0.0445	0.0279	0.0967	0.0493
	MLQR	0.0861	0.0551	0.1914	0.0899
Deep-learning	QR-DSSM	0.1158	0.0751	0.2492	0.1279
	CNN-based	0.1271	0.0965	0.2897	0.1637
Proposed	**QR-BERT$_{rep}$**	0.2226	0.1935	0.4156	0.2468
	QR-BERT$_{int}$	0.2562	0.2377	0.4440	0.2595
	QR-tBERT$_{int}$	**0.2920**	**0.2797**	**0.5012**	**0.2894**

indicates that taking advantage of the corpus-level topic information is quite effective for semantic understanding and can relieve the data sparsity problem in the question routing task.

2 Deep learning-based methods can significantly improve performance. These approaches mainly benefit from distributed word embeddings and efficient neural networks, which can capture more contextual semantic information through deeper and trainable architectures. Specifically, in Table 5, QR-DSSM achieves the best MRR of 0.2492, meaning that the question could be routed to only 5 users on average to obtain an answer, while LDA requires at least 11 users and MLQR requires at least 6 users. Moreover, the mean average precision of QR-DSSM is almost 42.26% higher than that of MLQR, and the CNN-based method achieved better performance than QR-DSSM, but the improvement was insignificant. This indicates that the CNN-based method has the ability to capture more contextual information and select more discriminative features through the exquisite convolutional layers.

3 The two methods we proposed significantly surpass all baseline methods on our datasets in terms of all metrics. This result is encouraging and indicates that our models are quite effective in addressing the question routing task. Specifically,

the best mean average precision of QR-BERT$_{rep}$ and QR-tBERT$_{int}$ is nearly 1.75 times and 2.22 times higher than MLQR, respectively. Moreover, QR-tBERT$_{int}$ proved its remarkable superiority over the neural baseline model QR-DSSM and the CNN-based method in modeling the semantic similarity and sequence relationship jointly. In Table 5, when X=20, the best value of MRR achieved by QR-tBERT$_{int}$ is 0.5012, which indicates that each test question will be answered if we route it to the top 2 users on average. In contrast, QR-DSSM requires at least 5 users, and the CNN-based model requires at least 4 users.

4 From Tables 3-5, we can observe that the D_{20} set achieves better results than the D_{15} set, and the D_{15} set achieves better results than the D_{10} set. This indicates that fewer negative samples can lead to better results in our dataset.

5 In addition, we note that the absolute values of model performance are relatively low in all three tables. We summarize the main reasons for this as follows. First, CQA faces a serious data sparsity problem, which leads to insufficient text for question modeling and user modeling. Not only the text lengths of the questions and answers are short, but we also can see from Table 1 that the number of questions is much larger than the number of answerers. With an average of only a few user comments per question and a very low average number of answers posted per user, the reality is that most users are not active in CQA. Second, we constructed our dataset by including the 21 most frequent tags, rather than including only a few tags. This makes our dataset more generalizable, but also more diverse in terms of the topics for which information is searched. As a result, finding the right expert to answer a specific question can be very challenging. Third, Stack Overflow is a vertical community Q&A with a complex composition of data, including code snippets, tables, domain-specific terms, and a few other discrete pieces of text. All of these factors contribute to the low absolute value of performance data.

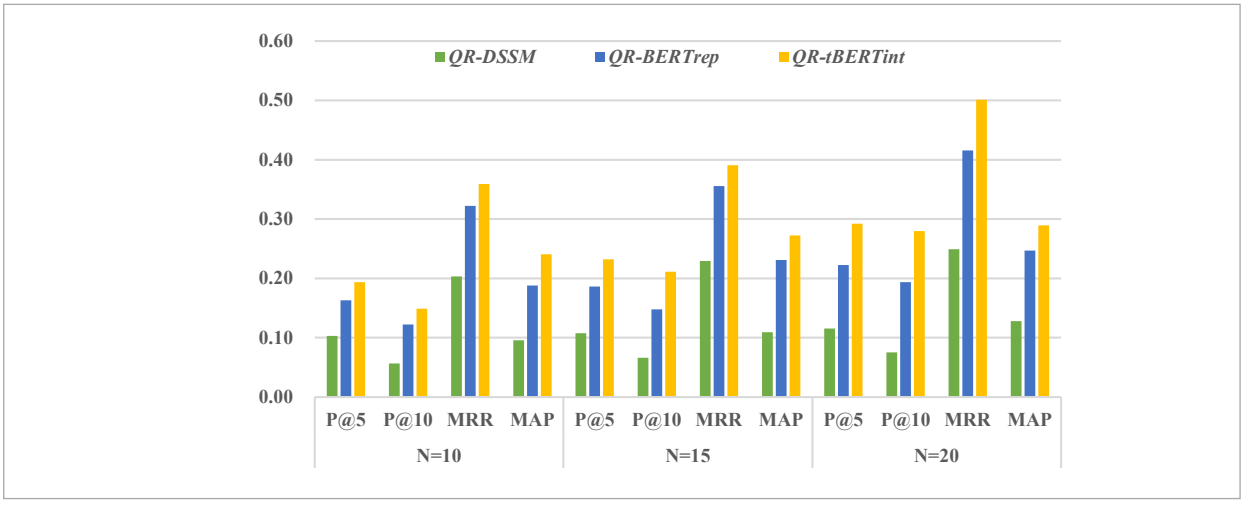
5.2. Analysis of Representation-based Methods and Interaction-based Methods

As mentioned before, our proposed two BERT-based models, QR-BERT$_{int}$ and QR-tBERT$_{rep}$, are both very effective on our dataset compared with three types of baselines. In this section, we will analyze their differences in more depth, and the performance comparison is shown in Figure 3.

As indicated in Figure 3, the topic-enhanced interaction-based model QR-tBERT$_{int}$ performs much better than representation-based models QR-DSSM and QR-BERT$_{rep}$. In addition, QR-tBERT$_{int}$ significantly exceeds QR-BERT$_{rep}$ with a maximum increase of 31.17% in p@5 and 44.54% in p@10, respectively. The main reasons can be summarized as follows: First, QR-tBERT$_{int}$ takes into account the interaction between sequences by connecting questions and user profiles in pairs as input, so that the hierarchical relationship between questions and profiles can be learned

Figure 3

Performance comparison of QR-DSSM, QR-BERT$_{int}$ and QR-tBERT$_{rep}$



as an essential feature of matching. In contrast, QR-BERT$_{rep}$ encodes the question sequence and profile sequence separately so that the interaction between the two sequences is deferred to the end of the matching process, risking the loss of details important for matching. Second, QR-tBERT$_{int}$ takes the fine-tuning strategy to learn cross-attention between terms by directly using Transformers located in BERT. In contrast, QR-BERT$_{rep}$ only uses the pre-trained network to construct sequence representations. Third, the tag-word topic model can provide corpus-level information to enhance the understanding of the semantic relevance of the text.

5.3. Analysis of Tag-word Topic Representation Module

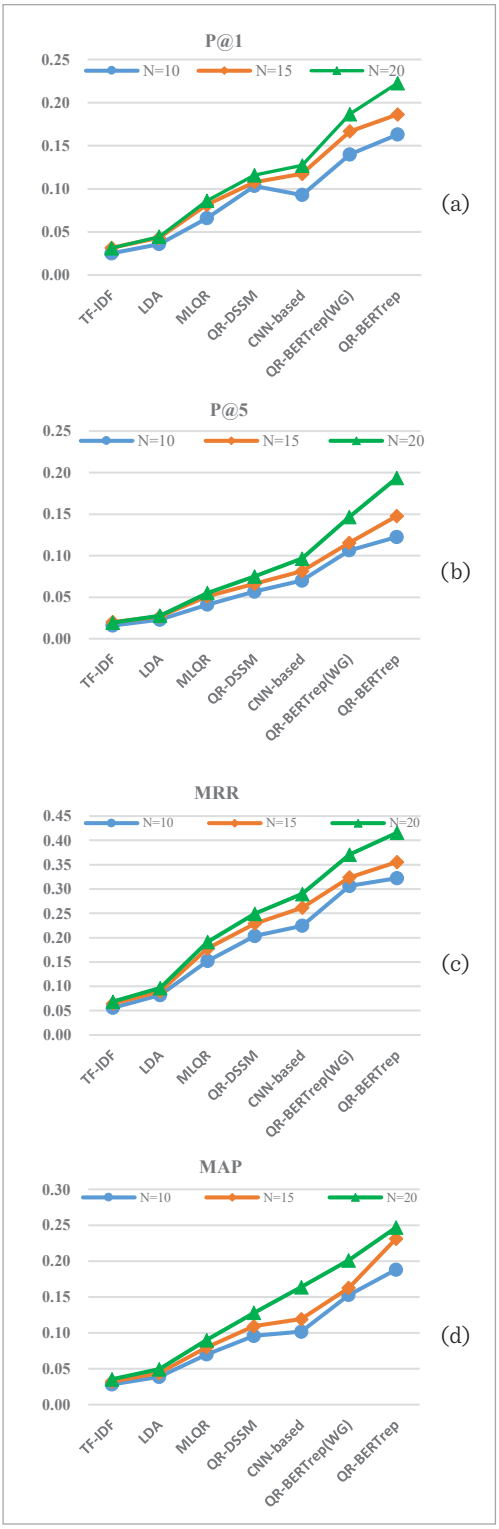
From Tables 3-5, we can see that combining tag-word topics can consistently improves the question routing performance across all metrics for all datasets. Specifically, without the tag-word topic model, the performance of QR-BERT$_{int}$ decreases by 11.41% and 10.33% in MRR and MAP, respectively, compared to QR-tBERT$_{int}$. The main reason we summarize is that Stack Overflow is a programming-specific Q&A community, where the ability to detect domain-specific terms is crucial for text semantic understanding and matching. However, the pre-training of BERT is based on general domain knowledge and is likely to fail to learn domain-specific words related to programming. Here, the tag-word topic model could serve as an additional source for dataset-specific information. Our findings are consistent with a lot of previous work that also confirms the effectiveness of incorporating topic models when dealing with semantic related tasks in specific knowledge domains, such as sentiment analysis in Microblogs [22] and machine translation [5].

5.4. Contextualized Embedding vs. Traditional Word Embedding

In this section, the proposed QR-BERT$_{rep}$ model is compared with the baselines in terms of word embedding representation, which is a crucial part that affects the performance of the representation-based models. To explore the respective effects of contextualized embedding and word embedding, we conducted an ablation experiment called QR-BERT$_{rep}$(WG) in which the GloVe embedding was removed. The performance comparison of different representation-based models is shown in Figure 4.

It can be seen that the methods using distributed word representation perform much better than the methods that represent the words in a sentence as a "bag of words". Therefore, TF-IDF has the lowest MRR and MAP. In addition, we can see that neural rankers such as QR-DSSM and CNN-based models are greatly facilitated by using pre-trained word embeddings (e.g., Word2Vec or GloVe) for sequence representation. In QR-BERT$_{rep}$, we concatenate the GloVe embedding and con-

Figure 4
The performance comparison of different representation-based models



textualized embedding together, the performance is dramatically boosted, almost doubling that of QR-DSSM. This result is consistent with previous observations in [19, 16], indicating that using contextualized language term embedding for text understanding and matching is very effective.

6. Conclusions

In this paper, we explore two different ways to address the question routing task for CQA based on a pre-trained contextual language model. QR-tBERT$_{int}$ is an interaction-based model that takes question-profile pairs as input and fine-tunes BERT to capture the relationship between sequence pairs. In addition, a tag-word topic model is incorporated as an additional source of dataset-specific information. QR-BERT$_{rep}$ is a representation-based model that combines contextualized embedding with traditional static word embedding to enhance the representation for semantic understanding and matching.

Experimental results on real-world data demonstrated that both of our proposed models greatly exceed state-of-the-art baselines. The best result indicates that a question will be answered if it is routed to the top 2 candidates. QR-BERT$_{rep}$ exceeds all representation-based baselines discussed in this paper, showing that contextualized word embedding can carry richer semantic information to enhance the representation in our task. Meanwhile, QR-tBERT$_{int}$ performs much better than QR-BERT$_{rep}$, which indicates that the question routing task benefits from sequence relationship learning and corpus-level topical semantic information.

Although we have made some progress in our work, in future work we would like to introduce more QA features (e.g., reputation, the willingness of experts) or non-QA features (e.g., number of followers and connected accounts on social networking sites) to enhance the performance. Moreover, taking advantage of the knowledge graph to improve the effectiveness of question routing is a very interesting work for the future.

References

1. Azzam, A., Tazi, N., Hossny, A. A Question Routing Technique Using Deep Neural Network for Communities of Question Answering. International Conference on Database Systems for Advanced Applications, Springer, 2017, 35-49. https://doi.org/10.1007/978-3-319-55753-3_3
2. Ba, J. L., Kiros, J. R., Hinton, G. E. Layer Normalization. Stat, 2016, 1050, 21.
3. Balog, K., Azzopardi, L., De Rijke, M. Formal Models for Expert Finding in Enterprise Corpora. Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 2006, 43-50. https://doi.org/10.1145/1148170.1148181
4. Balog, K., Azzopardi, L., de Rijke, M., Management. A Language Modeling Framework for Expert Finding. Information Processing, 2009, 45(1), 1-19. https://doi.org/10.1016/j.ipm.2008.06.003
5. Chen, W., Matusov, E., Khadivi, S., Peter, J.-T. Guided Alignment Training for Topic-Aware Neural Machine Translation. arXiv Preprint, 2016.
6. Cheng, X., Zhu, S., Su, S., Chen, G. A Multi-Objective Optimization Approach for Question Routing in Community Question Answering Services. IEEE Transactions on Knowledge Data Engineering, 2017, 29(9), 1779-1792. https://doi.org/10.1109/TKDE.2017.2696008
7. Dehghan, M., Biabani, M., Abin, A. A. Temporal Expert Profiling: With an Application to T-Shaped Expert Finding. ACM Transactions on Knowledge Discovery from Data Management Information Processing, 2019, 55(3), 1067-1079. https://doi.org/10.1016/j.ipm.2019.02.017
8. Devlin, J., Chang, M.-W., Lee, K., Toutanova, K. Bert: Pre-Training of Deep Bidirectional Transformers for Language Understanding, 2018, 4171-4186. arXiv preprint arXiv:1810.04805.
9. Dong, H., Wang, J., Lin, H., Xu, B., Yang, Z. Predicting Best Answerers for New Questions: An Approach Leveraging Distributed Representations of Words in Community Question Answering. 9th IEEE International Conference on Frontier of Computer Science and Technology, 2015, 13-18. https://doi.org/10.1109/FCST.2015.56
10. Hochreiter, S., Schmidhuber, J. Long Short-Term Memory. Neural Computation, 1997, 9(8), 1735-1780. https://doi.org/10.1162/neco.1997.9.8.1735
11. Huang, P.-S., He, X., Gao, J., Deng, L., Acero, A., Heck, L. Learning Deep Structured Semantic Models for Web Search Using Clickthrough Data. Proceedings of the 22nd ACM International Conference on Information & Knowledge Management, 2013, 2333-2338. https://doi.org/10.1145/2505515.2505665

12. Ji, Z., Wang, B. Learning to Rank for Question Routing in Community Question Answering. Proceedings of the 22nd ACM International Conference on Information and Knowledge Management, 2013, 2363-2368. https://doi.org/10.1145/2505515.2505670

13. Kim, Y. Convolutional Neural Networks for Sentence Classification, 2014. arXiv preprint arXiv:1408.5882. https://doi.org/10.3115/v1/D14-1181

14. Li, B., King, I., Lyu, M. R. Question Routing in Community Question Answering: Putting Category in Its Place. Proceedings of the 20th ACM International Conference on Information and Knowledge Management, 2011, 2041-2044. https://doi.org/10.1145/2063576.2063885

15. Li, Z., Jiang, J.-Y., Sun, Y., Wang, W. Personalized Question Routing via Heterogeneous Network Embedding, 2019. https://doi.org/10.1609/aaai.v33i01.3301192

16. Li, Z., Jiang, J.-Y., Sun, Y., Wang, W. Personalized Question Routing via Heterogeneous Network Embedding. Proceedings of the AAAI Conference on Artificial Intelligence, 2019, 33(01), 192-199. https://doi.org/10.1609/aaai.v33i01.3301192

17. Liu, M., Liu, Y., Yang, Q. Predicting Best Answerers for New Questions in Community Question Answering. International Conference on Web-Age Information Management, Springer, 2010, 127-138. https://doi.org/10.1007/978-3-642-14246-8_15

18. Liu, Z., Jansen, B. J. Analysis of Question and Answering Behavior in Question Routing Services. CYTED-RITOS International Workshop on Groupware, Springer, 2015, 72-85. https://doi.org/10.1007/978-3-319-22747-4_6

19. MacAvaney, S., Yates, A., Cohan, A., Goharian, N. Cedr: Contextualized Embeddings for Document Ranking. Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2019, 1101-1104. https://doi.org/10.1145/3331184.3331317

20. Mathew, B., Dutt, R., Maity, S. K., Goyal, P., Mukherjee, A. Deep Dive into Anonymity: Large Scale Analysis of Quora Questions. International Conference on Social Informatics, Springer, 2019, 35-49. https://doi.org/10.1007/978-3-030-34971-4_3

21. Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., Dean, J. Distributed Representations of Words and Phrases and Their Compositionality. Advances in Neural Information Processing Systems, 2013, 3111-3119.

22. Palani, S., Rajagopal, P., Pancholi, S. T-BERT--Model for Sentiment Analysis of Micro-blogs Integrating Topic Model and BERT. arXiv Preprint, 2021.

23. Peinelt, N., Nguyen, D., Liakata, M. tBERT: Topic Models and BERT Joining Forces for Semantic Similarity Detection. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 2020, 7047-7055. https://doi.org/10.18653/v1/2020.acl-main.630

24. Pennington, J., Socher, R., Manning, C. Glove: Global Vectors for Word Representation. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014, 1532-1543. https://doi.org/10.3115/v1/D14-1162

25. Qiu, X., Huang, X. Convolutional Neural Tensor Network Architecture for Community-Based Question Answering. 24th International Joint Conference on Artificial Intelligence, 2015.

26. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I. Improving Language Understanding by Generative Pre-Training. Preprint, 2018.

27. Riahi, F., Zolaktaf, Z., Shafiei, M., Milios, E. Finding Expert Users in Community Question Answering. Proceedings of the 21st International Conference on World Wide Web, 2012, 791-798. https://doi.org/10.1145/2187980.2188202

28. Sun, J., Zhao, J., Sun, H., Parthasarathy, S. EndCold: An End-to-End Framework for Cold Question Routing in Community Question Answering Services. Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, 2020, 3244-3250. https://doi.org/10.24963/ijcai.2020/449

29. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., Polosukkhin, I. Attention is All You Need. Advances in Neural Information Processing Systems, 2017, 5998-6008.

30. Wang, G., Gill, K., Mohanlal, M., Zheng, H., Zhao, B. Y. Wisdom in the Social Crowd: An Analysis of Quora. Proceedings of the 22nd International Conference on World Wide Web, 2013, 1341-1352. https://doi.org/10.1145/2488388.2488506

31. Wang, J., Sun, J., Lin, H., Dong, H., Zhang, S. Convolutional Neural Networks for Expert Recommendation in Community Question Answering. Science China Information Sciences, 2017, 60(11), 1-9. https://doi.org/10.1007/s11432-016-9197-0

32. Yang, L., Qiu, M., Gottipati, S., Zhu, F., Jiang, J., Sun, H., Chen, Z. Cqarank: Jointly Model Topics and Expertise in Community Question Answering. Proceedings of the 22nd ACM International Conference on Information and Knowledge Management, 2013, 99-108. https://doi.org/10.1145/2505515.2505720

33. Yang, N., Jo, J., Jeon, M., Kim, W., Kang, J. Semantic and Explainable Research-Related Recommendation System Based on Semi-Supervised Methodology Using BERT and LDA Models. Expert Systems with Applications, 2022, 190, 116209. https://doi.org/10.1016/j.eswa.2021.116209

34. Yuan, S., Zhang, Y., Tang, J., Hall, W., Cabotú, J. B. Expert Finding in Community Question Answering: A Review. Artificial Intelligence Review, 2019, 1-32. https://doi.org/10.1007/s10462-018-09680-6

35. Zhang, X., Cheng, W., Zong, B., Chen, Y., Xu, J., Li, D., Chen, H.Temporal Context-Aware Representation Learning for Question Routing. Proceedings of the 13th International Conference on Web Search and Data Mining, 2020, 753-761. https://doi.org/10.1145/3336191.3371847

36. Zhao, Z., Yang, Q., Cai, D., He, X., Zhuang, Y. Expert Finding for Community-Based Question Answering via Ranking Metric Network Learning. IJCAI, 2016, 16, 3000-3006.

37. Zheng, C., Zhai, S., Zhang, Z. A Deep Learning Approach for Expert Identification in Question Answering Communities. arXiv preprint arXiv:.05350, 2017.

38. Zhou, G., Lai, S., Liu, K., Zhao, J. Topic-Sensitive Probabilistic Model for Expert Finding in Question Answer Communities. Proceedings of the 21st ACM International Conference on Information and Knowledge Management, 2012, 1662-1666. https://doi.org/10.1145/2396761.2398493

39. Zhou, G., Zhao, J., He, T., Wu, W. An Empirical Study of Topic-Sensitive Probabilistic Model for Expert Finding in Question Answer Communities. Knowledge-Based Systems, 2014, 66, 136-145. https://doi.org/10.1016/j.knosys.2014.04.032

40. Zhou, Y., Cong, G., Cui, B., Jensen, C. S., Yao, J. Routing Questions to the Right Users in Online Communities. 2009 IEEE 25th International Conference on Data Engineering, 2009, 700-711. https://doi.org/10.1109/ICDE.2009.44