Knowledge-Based Systems 257 (2022) 109933

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

# Towards comprehensive expert finding with a hierarchical matching network

Qiyao Peng [a], Wenjun Wang [b,d], Hongtao Liu [c], Yinghui Wang [b], Hongyan Xu [b], Minglai Shao [a,*]

[a] School of New Media and Communication, Tianjin University, Tianjin, China
[b] College of Intelligence and Computing, Tianjin University, Tianjin, China
[c] Du Xiaoman Financial, Beijing, China
[d] College of Information Science and Technology, Shihezi University, Xinjiang, China

ARTICLE INFO

*Article history:*
Received 15 March 2022
Received in revised form 6 September 2022
Accepted 20 September 2022
Available online 30 September 2022

*Keywords:*
Expert finding
Hierarchical matching
Personalized
Community question answering

ABSTRACT

In Community Question Answering (CQA) websites, expert finding aims to seek relevant experts for answering questions. The core of expert finding is to match candidate experts and target questions precisely. Most existing methods usually learn a single feature vector for the expert from the historically answered questions, and then match the target question, which would lose fine-grained and low-level semantic matching information. In this paper, instead of matching with a unified expert embedding, we propose an expert finding method with a multi-grained hierarchical matching framework, named EFHM. Specifically, we design a word-level and question-level match encoder to learn the fine-grained semantic matching between each historical answered question and target question, and then propose an expert-level match encoder to learn an overall expert feature for matching the target question. Through the hierarchical matching mechanism, our model has the potential to capture the comprehensive relevance between candidate experts and target questions. Experimental results on six real-world CQA datasets demonstrate that the proposed method could achieve better performance than existing state-of-the-art methods.



## 1. Introduction

Community Question Answering websites (CQAs) such as Stack Overflow,[1] Zhihu[2] have already obtained popular use in reality, and users can ask questions or post answers for questions they are interested in [1–4]. However, these websites have accumulated millions of questions [5,6] and many questions wait to be answered over time [7]. Hence, expert finding is a core task to find appropriate users for answering questions, which could help users receive satisfying answers and then improve the user experience on the CQA websites.

The key to the expert finding lies in the accurate matching of the expert's interest and target questions. The same expert usually has diverse interests, which are reflected in different historical questions she/he has answered. Meanwhile, the important semantic matching clues are implied of different granularity, which could contain word-level, question-level and expert-level. Fig. 1 illustrates the challenges with an example. As illustrated in Fig. 1, the expert has answered a series of questions historically. We can find that the expert has the potential to answer $q_t^1$ because his/her historically answered question $q_u^1$ contains fine-grained clues (**i. word-level clues**), such as the words "Pytorch", "framework" which can be matched with the target question, though the question sentence semantics would not match. Hence, if the expert finding model only learns a single embedding from histories for matching the target question, the important signals of these words would be weakened.

Meanwhile, not all target questions can be matched with the expert word-level fine-grained features. For instance, the target question $q_t^2$ should be recommended to the expert because the historically answered questions $q_u^2$ could provide question-level semantic signals which are relevant to the target question (**ii. question-level clues**). However, as shown in Fig. 1, the target question $q_t^3$ cannot match any fine-grained features of the candidate expert. Fortunately, the coarse-grained overall expert features can provide clues for matching (**iii. expert-level clues**). Therefore, it is necessary to capture different grained signals and integrate them effectively to match the target questions and candidate experts for expert finding.

Most existing expert finding methods, however, usually learn a single expert feature by integrating all historical questions

* Corresponding author.
*E-mail address:* shaoml@tju.edu.cn (M. Shao).

1 https://www.stackoverflow.com
2 https://www.zhihu.com

https://doi.org/10.1016/j.knosys.2022.109933

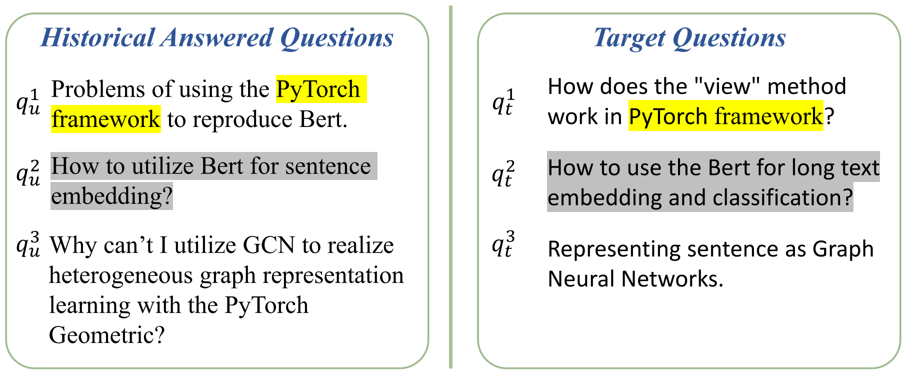


**Fig. 1.** Examples of one expert historically answered question and target questions.

that the user has answered and then matching the final expert vector with the target question vector [8–11]. For example, Li et al. [11] utilized a heterogeneous network embedding and an LSTM architecture to learn a single representation for the expert and matched that with the target question. TCQR [12] learned the expert representations in the context of both the semantic and temporal information for matching with the target question. Fu et al. [13] utilized the reasoning memory cell to explore the implicit relevance between a requester's question and candidate experts' historical records. Despite the improvements of these methods in expert finding performance, they are limited in capturing different grained expert-question matching signals, which could not match expert latent interests with the target questions comprehensively until the final step of prediction.

To this end, in this paper we propose an Expert Finding method with a Hierarchical Matching network on CQA websites, named **EFHM**, for capturing more comprehensive matching signals between experts and target questions. The cores of EFHM are to represent the expert as multi-grained features and match the expert different grained representations with the target question. Firstly, we design a *question encoder* to learn the question-level semantic feature. Secondly, our approach utilizes a hierarchical matching network to capture the multi-grained relevance between the expert and the target question, i.e., a *word-level match encoder* to compute the word-level semantic relevance, a *question-level match encoder* to learn the question-level relevance for measuring the semantic relationship between the historically answered questions and the target question, and a personalized attention-based *expert-level match encoder* to integrate the historically answered questions for learning the overall expert feature and matching with the target question. Furthermore, to avoid matching signals loss, we employ a controller to complementarily unify the multiple matching signals derived from the hierarchical matching network, which could control the different grained importance. Then, the model could calculate the final relevance score between the candidate expert and the target question.

The contributions of this paper are summarized:

- We propose an expert finding method with a hierarchical matching mechanism, aiming to more precisely match the expert and target questions for routing suitable experts to answer questions on CQA websites.
- We design a word-level match encoder, a question-level matching encoder and an expert-level match encoder to capture multi-grained relevance between experts and target questions, which could complementarily and comprehensively capture expert-question matching signals.
- We conduct extensive experiments on six real-world datasets and the results show that our method could achieve better performance than existing baselines and validate the effectiveness of our approach EFHM.

The rest of the paper is arranged as follows. Section 2 presents the recent related works mainly in traditional expert finding methods and deep-learning based expert finding methods. In Section 3, we describe our overall model in detail. The experimental settings and result analysis are reported in Section 4. The conclusion is drawn in Section 5.

## 2. Related works

Recently, the research on Community Question Answering systems (CQA) has attracted more and more attention, such as expert finding [13], question–answer matching [2,6], etc. For example, Ghasemi et al. [14] combined expert relations with the question–answer similarity for improving expert finding. Hu et al. [5] proposed to design a multi-modal attentive graph pooling approach (MMAGP) to explores the multi-modal and redundant properties of CQA systems.

In this section, we will briefly review prior expert finding works on traditional expert finding methods and deep learning-based methods.

### 2.1. Traditional expert finding methods

Traditional expert finding methods mainly contain two categories, i.e., feature engineering-based methods and topic modeling-based methods.

Feature engineering was a popular technology for realizing expert finding in the past [15–20]. For example, Zhou et al. [16] extracted local and global features which can capture different aspects of questions, users, and their relations, then fed them to the SVM [21] for recommending suitable answerers. Although this kind of method has shown good results in expert finding, it relies heavily on the tedious hand-crafted features, which are time-consuming and unable to capture the complex answerer interest and question feature well. Considering that there are rich off-the-shelf texts of question contents (e.g., question title) containing important semantic information which is beneficial for modeling questions and reflecting answerer interests, many works exploiting question contents have been proposed to enhance expert finding.

The topic modeling technique such as Latent Dirichlet Allocation (LDA) [22] is widely utilized for learning topics from texts, some works exploited that to extract features from question contents for representing respondents and questions [8,9,23–29]. For example, Chang et al. [8] used LDA to extract the semantic information from question contents as question representations for expert finding. Yang et al. [27] proposed CQARank combining both textual content model results and link structures to simultaneously measure user topical expertise and interests for

**Table 1**
Characteristic of different methods.

| | CNTN [30] | NeRank [11] | TCQR [12] | RMRN [13] | UserEmb [14] | **EFHM** |
|---|---|---|---|---|---|---|
| Attention | × | × | √ | √ | × | √ |
| Personalized | × | √ | √ | × | √ | √ |
| Word-level match | × | × | × | × | × | √ |
| Question-level match | × | × | × | √ | × | √ |
| Expert-level match | √ | √ | √ | × | √ | √ |

improving expert finding. These methods have achieved outstanding performance on expert finding. However, the topic modeling methods are based on Bag-Of-Word (BOW) mechanism, which would ignore the word order and could not learn accurate semantic features from question contents [12,13].

### 2.2. Deep learning-based methods for expert finding

In recent, we witness the development of deep learning in various research fields, many works employ neural networks such as Convolutional Neural Network (CNN) to learn question features from question contents, and then learn a single expert feature for matching the target question feature [30–34]. For example, CNTN [30] integrated sentence modeling and semantic matching into a single model to capture useful information for community question answering. NeRank utilized an LSTM to learn question features and a heterogeneous information network embedding model [35] to learn single representations for raiser and answerer, then used a convolutional recommender system for expert finding. These neural network methods have achieved better performance than topic modeling-based methods due to the superior capability of capturing semantic features from question contents. However, these methods always ignore the importance of the different information for modeling the answerer, which would be insufficient to represent the answerer.

Attention mechanism in deep learning has been popular in many areas, e.g., natural language processing, and computer vision. This mechanism is based on an intuition that the model should attend to a certain part while processing a large amount of information. RMRN [13] utilized a recurrent memory reasoning network to focus on different parts of a question, and accordingly retrieved information from the histories of the candidate expert for modeling a single feature for the expert, which is the state-of-the-art expert finding method. TCQR [12] used a novel context-aware model in multiple granularities of temporal dynamics for modeling temporal aware single expertise representation to recommend the expert for target questions.

However, these methods mostly integrate expert historical answered questions into an abstract expert feature vector, which could be insufficient for learning comprehensive match information between candidate experts and target questions.

### 2.3. Compared with existing methods

In this paper, we proposed the EFHM aims to capture multi-grained relevance between target questions and experts. We summarize the recent deep learning-based methods under the following fashions: attention mechanism, expert ID (i.e., Personalized), word-level match, question-level match, and expert-level match characteristics. From Table 1, we can find that our method EFHM equipped with all the five characteristics has the superiority to comprehending relevance between target questions and candidate experts comprehensively, which has the potential to improve expert finding.

## 3. Proposed method

We first present the problem definition of expert finding on CQA websites. Next, we introduce our method for expert finding with a hierarchical matching network, denoted as EFHM As shown in Fig. 2, the hierarchical matching network has three main modules: a word-level match encoder to capture the word-level match clues, a question-level match encoder to capture the question-level match clues and an expert-level match encoder to capture overall match clues. Then the different granularity signals are aggregated via a predictor to obtain an overall relevance score. Furthermore, we design a question encoder to learn question-level representations and a personalized attention mechanism to learn personalized expert representations based on expert histories.

### 3.1. Problem definition

In this paper, the primary objective is to explore the most suitable expert, which could provide the "accepted answer" for a question on CQA websites. Suppose that there is a target question $q^t$ and a candidate expert set $U = \{u_1, u_2, \ldots, u_m\}$, where $m$ represents the number of experts. For an expert $u$ in $U$, he/she had answered historical questions can be denoted as $Q_u = \{q_u^1, q_u^2, \ldots, q_u^n\}$, where $n$ is the length of the set $Q_u$. And for the $i$th question $q_u^i$ in $Q_u$, it can be denoted as $q_u^i = \{w_1^i, w_2^i, \ldots, w_l^i\}$, where $l$ is the number of words in a question. Similarly, for the target question $q^t$, it can be denoted as $q^t = \{w_1^t, w_2^t, \ldots, w_l^t\}$, which contains $l$ words. The expert who provides the "accepted answer" is the ground truth in our method.

### 3.2. Question encoder

In this section, we design a *Question Encoder* to extract question-level semantic features from questions. Firstly, as shown in Fig. 2, given the target question or any question in $Q_u$, we utilize a word-embedding layer to convert each word in the question to a low-dimensional feature vector $\mathbf{s} \in \mathcal{R}^{d_w}$, then we stack these features as a word-embedding matrix $\mathbf{S} \in \mathcal{R}^{l\times d_w}$, denoted as:

$$\mathbf{S} = \{\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_l\}. \tag{1}$$

Secondly, considering that the Transformer [36] has the advantage to learn representations of long sequences, we employ $n_l$ Transformer encode layers in our method to capture contextual question-level semantic features, and each layer contains a multi-head self-attention layer and a position-wise feed-forward layer. The representation of $g$th word feature $\mathbf{s}_g$ in $\mathbf{S}$ learned by $k$th attention head is denoted as:

$$\mathbf{s}_g = \mathbf{s}_g + (\mathbf{s}_g)_p \ , g \in \{1, 2, \ldots, l\} \ , \tag{2}$$

$$\alpha_{g,j}^k = \frac{\exp(\mathbf{s}_g\mathbf{Q}_k(\mathbf{s}_j\mathbf{K}_k)^T)}{\sum_{x=1}^{l}\exp(\mathbf{s}_g\mathbf{Q}_k(\mathbf{s}_x\mathbf{K}_k)^T)} \ , \tag{3}$$

$$\mathbf{h}_{g,k} = (\sum_{j=1}^{l}\alpha_{g,j}^k\mathbf{s}_j)\mathbf{V}_k \ , \tag{4}$$

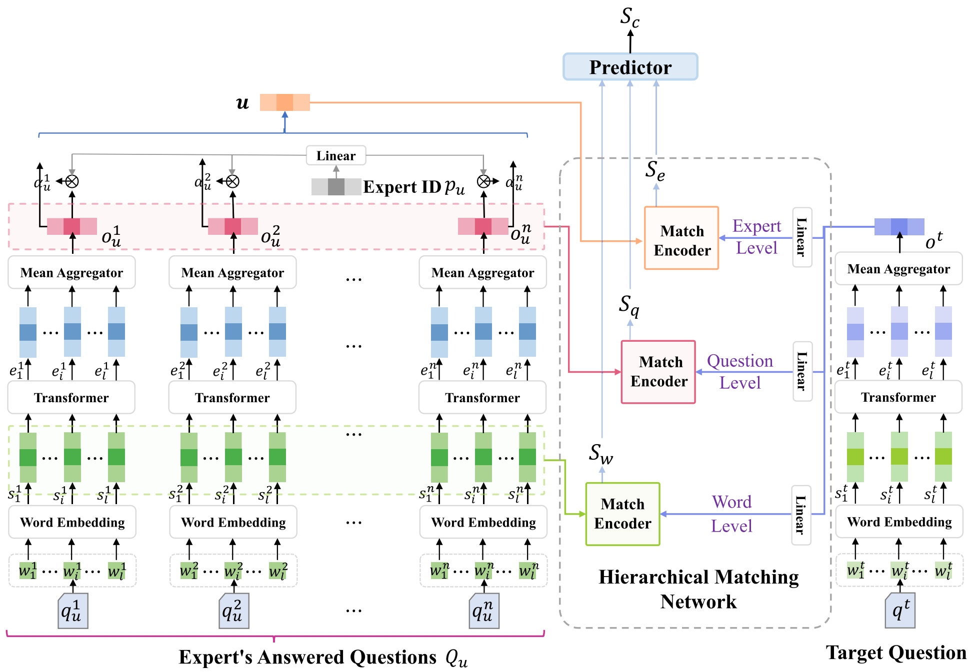


**Fig. 2.** Overview of our method **EFHM**.

where $\mathbf{s}_g \in \mathcal{R}^{d_w}$ and $(\mathbf{s}_g)_p \in \mathcal{R}^{d_w}$ is the word embedding and position embedding for $g_{th}$ word respectively, and $\mathbf{Q}_k$, $\mathbf{K}_k$, $\mathbf{V}_k$ are the parameters in the $k_{th}$ self-attention head, and $\alpha^k_{g,j}$ indicates attention score between the $g_{th}$ word and the $j_{th}$ word. The multi-head representation $\mathbf{h}_g$ of the $i_{th}$ token is the concatenation of the representations produced by $h$ different self-attention heads, i.e., $\mathbf{h}_g = \mathbf{h}_{g,1} \oplus \mathbf{h}_{g,2} \oplus \cdots \oplus \mathbf{h}_{g,h}$. Then the FFN layer and the layer normalization are followed. Afterward, we stack $l$ word contextual semantic features as a matrix $\mathbf{E} \in \mathcal{R}^{d_w \times l}$ and employ a mean pooling operator to obtain the question-level semantic feature for the question, which is computed as follows:

$$\mathbf{o} = mean(\mathbf{E}), \tag{5}$$

In this way, we can obtain the expert $u$ question-level semantic features $[\mathbf{o}^1_u, \mathbf{o}^2_u, \ldots, \mathbf{o}^n_u]$ and the target question $q^t$ semantic feature $\mathbf{o}^t \in \mathcal{R}^{d_w}$. Then, we will elaborate that how to construct three-tier matching signals between the candidate expert and target questions.

### 3.3. Hierarchical matching network

As denoted above, modeling candidate experts in different granularity and matching with target questions have the potential to recommend experts more accurately. In this section, we will introduce our hierarchical matching network, which could help the model capture different granularity matching signals between the candidate expert and the target question.

#### 3.3.1. Word-level match encoder

In this part, we introduce the *Word-Level Match Encoder*, which can employ the target question representation and expert historical question word embeddings effectively to extract the word-level fine-grained relevance signal for question-expert matches.

Based on the word-embedding layer in the *Question Encoder*, given the expert $u$, we transform his/her historical answered questions as word embedding matrices (i.e., $\mathbf{S}^1, \mathbf{S}^2, \ldots, \mathbf{S}^n$). Given the target question t, we employ the expert word embedding matrices and the target question $q^t$ question-level feature $\mathbf{o}^t$ to compute the word-level matching signal. This is because there are many articles (e.g., a/an/the) and some verbs (do/does) existed in the question title, which could impact word-level matching. And the question title is usually short, hence the question-level feature could represent the target question semantic feature.

Firstly, we use the dense layer to project the target question feature $\mathbf{o}^t$ into the word-level space, which is calculated as:

$$\mathbf{o}^t_w = \mathbf{W}_w \mathbf{o}^t + \mathbf{b}_w, \tag{6}$$

where $\mathbf{W}_w \in \mathcal{R}^{d_w \times d_w}$ and $\mathbf{b}_w \in \mathcal{R}^{d_w}$ are parameters.

Secondly, for the $g$th question $q^g_u$, we compute the $g$th word-level matching signal $S^g_w$ between each word in $\mathbf{S}^g$ and the target question representation $\mathbf{o}^t_w$, which is computed as:

$$S^g_w = max((\mathbf{S}^g) \times \mathbf{o}^t_w), \; g \in \{1, 2, \ldots, n\}, \tag{7}$$

where $\mathbf{S}^g \in \mathcal{R}^{l \times d_w}$ is the $g$th question word embedding matrix and $\mathbf{o}^t_w \in \mathcal{R}^{d_w}$ is the target question feature in the word-level space.

Finally, we perform the max-pooling operation on $n$ question word-level relevance scores to obtain the final word-level matching signal $S_w$ between the expert and the target question as follows:

$$S_w = max(S^1_w, \ldots, S^n_w). \tag{8}$$

In this way, the model could capture more important word-level expert information for matching the target question, and help the model predict more suitable experts through the word-level matching signal $S_w$.

#### 3.3.2. Question-level match encoder

In this part, we propose to utilize the *Question-level Match Encoder* to capture the question-level relevance signal between the expert and the question. Compared with the word-level match encoder, the question-level match encoder pays more attention to contextual semantic information in a question title, which is high-level. For example, the semantic feature of "apple" in "I like to eat apple" and "The apple phone is very smooth" is different. Hence, capturing the contextual semantic features is necessary for question-expert matches.

Given the expert $u$, we have obtained the historical question semantic feature $\mathbf{O}_u = [\mathbf{o}_u^1, \ldots, \mathbf{o}_u^n] \in \mathcal{R}^{n\times d_w}$ and the target question $q^t$ feature $\mathbf{o}^t$ from the question encoder. Then, we employ the dense layer to project the target question feature $\mathbf{o}^t \in \mathcal{R}^{d_w}$ to the question-level space as $\mathbf{o}_q^t$ and match that with $\mathbf{O}_u$, which is calculated as:

$$\mathbf{o}_q^t = \mathbf{W}_t\mathbf{o}^t + \mathbf{b}_t,\ \ \mathbf{S}_q = (\mathbf{O}_u) \times \mathbf{o}_q^t, \tag{9}$$

where $\mathbf{o}_q^t \in \mathcal{R}^{d_w}$ is the target question feature, and $\mathbf{S}_q \in \mathcal{R}^n$ is relevance scores between the expert historical answered questions and the target question.

Finally, we employ the max pooling to calculate the question-level relevance signal, which is denoted as follows:

$$S_q = max(\mathbf{S}_q). \tag{10}$$

In this way, the model could select more important expert historical behaviors for matching the target question, and help the model predict more suitable experts through the question-level matching signal $S_q$.

#### 3.3.3. Expert-level match encoder

In this part, we introduce the *Expert-level Match Encoder*, which can calculate an overall matching signal. Considering that different expert has his/her unique characteristic and different histories have different contributions to him, we utilize the personalized attention mechanism to aggregate expert historically answered question semantic features for obtaining the overall expert representation. Then, we match the target question feature with an overall expert feature for calculating expert-level signals.

Firstly, given the historical answered question features $[\mathbf{o}_u^1, \mathbf{o}_u^2, \ldots, \mathbf{o}_u^n]$ and the expert ID $p_u$, we utilize the ID embedding layer for projecting $p_u$ to $\mathbf{p}_u$ with $d_w$ as the dimension. Then, we utilize the personalized attention mechanism to aggregate the histories $[\mathbf{o}_u^1, \ldots, \mathbf{o}_u^n]$ for obtaining the overall expert feature, and the attention weight $\alpha_u^g$ of the $g$th question $\mathbf{o}_u^g \in \mathcal{R}^{d_w}$ is computed as follows:

$$\alpha_u^g = \frac{\exp(l_u^g)}{\sum_{j=1}^n \exp(l_u^j)},\ \ l_u^g = (\mathbf{p}_u)^T \odot \mathbf{o}_u^g, \tag{11}$$

where $l_u^g$ is the attention score of the $g$th question and $\odot$ is the dot product operator. Afterward, we aggregate the historically answered questions according to their different attention weights, which are calculated as follows:

$$\mathbf{u} = \sum_{g=1}^n \alpha_u^g \mathbf{o}_u^g, \tag{12}$$

where $\mathbf{u} \in \mathcal{R}^{d_w}$ represents the expert overall feature.

Finally, we utilize a dense layer to project the target question feature $\mathbf{o}^t$ to the expert-level space and match that with the overall expert feature $\mathbf{u}$, which is calculated as follows:

$$\mathbf{o}_e^t = \mathbf{W}_e\mathbf{o}_e^t + \mathbf{b}_e,\ \ S_e = (\mathbf{u})^T \odot \mathbf{o}_e^t, \tag{13}$$

where $\mathbf{W}_e \in \mathcal{R}^{d_w\times d_w}$ and $\mathbf{b}_e \in \mathcal{R}^{d_w}$ are the parameters, and $S_e$ represents the overall relevance score, which can provide coarse-grained clues for matching.

Above all, we design a hierarchical matching network for capturing more comprehensive relevance signals between the candidate expert and the target question (i.e., the world-level relevance $S_w$, the question-level relevance $S_q$ and the expert-level relevance $S_e$), which could help recommend more suitable experts for questions accurately.

### 3.4. Predictor

Considering that different granularity matching signals have different contributions to calculating the overall matching score, we design a controller to unify the three-tier relevance signals as follows:

$$S_c = w'_w S_w + w'_q S_q + w'_e S_e, \tag{14}$$

where $w'_w, w'_q, w'_e \in (0,1), w'_w + w'_q + w'_e = 1$ are trainable parameters for controlling the relative importance of different level signals.

Lastly, the candidate expert with the highest overall matching score will be recommended as the most suitable expert for answering the target question.

### 3.5. Model training

In this section, we utilize the negative sampling [37] method to train our model. The expert who provides the “accepted answer” is a positive sample for each question. And we sample other $K$ experts as negative samples. Therefore, the training objective is to minimize the cross-entropy loss between the true labels and predicted labels as follows:

$$\bar{S_c} = \frac{S_c}{\sum_{g=1}^{K+1} \exp(S_g)},\ \ c \in \{1, 2, \ldots, K+1\}, \tag{15}$$

$$Loss = -\sum_{c=1}^{K+1} \hat{S_c} log \bar{S_c}, \tag{16}$$

where $\hat{S_c}$ denotes the ground truth and $S_c$ represents the predicted probability of the $c$th sample, and $\bar{S_c}$ is the normalization probability of the $c$th sample.

## 4. Experiments

In this section, we evaluate the performance of our proposed model using six real-world datasets. We will introduce datasets and experimental settings firstly, then present experimental analyses in detail.

### 4.1. Datasets and experimental settings

We randomly select six real-world CQA datasets from StackExchange,[3] including six different domains, i.e., **AI Print**, **History**, **Biology**, **English** and **Bioinformatics**. Each dataset contains all questions raised before June 2019. And each dataset includes a question set, in which, each of the questions is associated with its title, and corresponding answerers, including the lists of answerers and the expert who provided the “accepted answers”. We refer to the preprocessing method in the previous work [11], and filter the answerers who provided less than 5 answers out of the dataset to avoid the cold-start problem. We construct a candidate expert set including 20 experts for each question containing its original answerers (including one expert) and others randomly selected from the answerer set. The detailed statistical characteristics of the datasets are shown in Table 2. We split each dataset into a training set, a validation set and a testing set, with ratios 80%, 10% and 10% respectively in chronological order. The number of historical answered questions is set to 30 (i.e., $n = 30$). For each question, the word length of its title is 15 (i.e., $l = 15$).

3 https://archive.org/details/stackexchange

**Table 2**
Statistical details of the datasets.

| Datasets | # Questions | # Answerers | # Answers | Avg.title length |
|---|---|---|---|---|
| AI | 1205 | 195 | 1719 | 10.97 |
| Print | 1033 | 112 | 1686 | 9.51 |
| History | 4904 | 471 | 9452 | 12.38 |
| Biology | 8704 | 630 | 11,411 | 9.84 |
| English | 46,692 | 4781 | 104,45 | 9.68 |
| Bioinformatics | 958 | 113 | 1489 | 9.93 |

#### 4.1.1. Hyper-parameter setting

In our experiment, we utilize the validation set to tune hyper-parameters. We set the dimensions of the word-level feature, question-level feature and expert-level feature to 100. The number of Transformer encoder heads (i.e., $h$) is 2 (tuning in $[1, 2, 4, 5]$) and the number of Transformer encode layers (i.e., $n_l$) is 2 (tuning in $[1, 2, 3, 4]$). The batch size is set to 64. We independently repeat each experiment 5 times and report the average results. All experiments are implemented using Pytorch frame and using a 24 GB-memory RTX 3090 GPU server with Intel(R) Xeon(R)@2.20 GHz CPU. We release our code and data in an anonymous link for reproducing.[4] To alleviate the overfitting problem, we utilize a dropout technology [38] and set the dropout ratio to 0.25. We adopt an Adam [39] optimization strategy to optimize our model and set the learning rate to 0.001, and the weight decay is 0.0005.

#### 4.1.2. Evaluation metrics

To verify the effectiveness of our proposed model, Mean Reciprocal Rank (MRR), Precision and Normalized Discounted Cumulative Gain (NDCG) are utilized as our evaluation metrics following the previous work [12]. It is noticed that the higher MRR, Precision and NDCG indicate better performance. Next, we will introduce these three evaluation metrics in detail.

- Mean Reciprocal Rank (MRR): The average multiplicative reciprocal of the accepted answerer (i.e., expert) ranking is denoted as:

$$\mathrm{MRR} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{r_i}\ , \tag{17}$$

  where $N$ is the sample number and $r_i$ is the rank of the expert among the candidate answerers for each of the questions.
- Precision@K: The proportion of predicted instances where the expert appears in the top-$K$ result among candidate answerers. For instance, P@1 indicates the percentage of cases where the expert appears in the top-1 ranked results.
- Normalized Discounted Cumulative Gain (NDCG@K): The normalized gain of each expert based on his/her ranking position in the results. The $DCG_K$ is computed as:

$$DCG_K = \sum_{i=1}^{K}\frac{r_i}{log_2(i+1)}\ , \tag{18}$$

  where $r_i$ is 1 when the $i$th answerer is the expert. $IDCG_K$ is the ideal discounted cumulative gain. The NDCG@$K$ is computed as: NDCG@$K$ $= \frac{DCG_K}{IDCG_K}$. We set the value of $K$ to 10 in our experiments (i.e., NDCG@10).

In summary, MRR, P@1 and NDCG@10 are different evaluation metrics for the ranking quality of the expert predicted by a model. We perform t-tests and the results show that EFHM significantly outperforms other baseline methods at significance level $p$-value $<0.05$.

[4] https://github.com/CodesAnonyMous/KBS_EFHM

#### 4.1.3. Baselines

We compare our method EFHM with recent competitive methods including:

- (1) Score: A trivial method recommends the answerers that have a larger number of "accepted answer".
- (2) Doc2Vec: This method recommends the experts who have answered the historical question most similar to the target question.
- (3) BM25 [15]: BM25 is utilized to estimate the relevance based on word matching between documents and a given query.
- (4) CNTN [30]: CNTN utilizes the convolution neural networks to learn the question semantic features and compute the relevance.
- (5) NeRank [11]: NeRank learns features of questions, raisers and experts with a heterogeneous network embedding algorithm, and utilizes a CNN network to compute the ranking scores of potential experts.
- (6) TCQR [12]: This method utilizes a temporal context-aware model in multiple granularities for capturing temporal-aware expertise to realize multi-faceted expert learning.
- (7) RMRN [13]: The model utilizes a novel recurrent memory reasoning network for expert finding by exploring the implicit relevance between target questions and a candidate expert.
- (8) UserEmb [14]: Ghasemi et al. combine experts' community relations with the similarity between their questions and existing answers for improving the expert finding.

### 4.2. Performance comparison

In this section, we compare our method with the above baselines. Tables 3 and 4 summarize the results over all the datasets. First of all, the neural network models (i.e., Doc2Vec, CNTN, NeRank, TCQR, RMRN, UserEmb) usually achieve excellent performance than traditional expert finding methods (i.e., Score, BM25). The reason is that neural networks could capture more deep semantic information from complicated sentences (e.g., question title) and promote semantic interactions between questions and experts' historical answered questions, compared with traditional methods. Secondly, we could find that the methods considering the question-level relevance signals between experts and questions (e.g., RMRN, EFHM) usually achieve better performance than those not (e.g., CNTN, TCQR, UserEmb). This is because different historical questions of an expert would have different informativeness to modeling the expert when facing the new question.

Thirdly, our method EFHM achieves the best results over all datasets in terms of all metrics. This indicates EFHM has a stronger ability than other baseline methods which learn a single expert feature for matching the expert with a target question. The reasons and clues can be found in Table 1. Our method represents each expert as multi-grained features and proposes a hierarchical matching framework to capture three-tier relevance signals between experts and target questions, which can model interactions

**Table 3**
Expert finding results (%) of different methods on three datasets (i.e., AI, Bioinformatics and Print). The best performance of the baselines is underlined. Our method results are in bold.

| Datasets | AI | | | Bioinformatics | | | Print | | |
|---|---|---|---|---|---|---|---|---|---|
| Method | Metric | | | | | | | | |
| | MRR | P@1 | NDCG@10 | MRR | P@1 | NDCG@10 | MRR | P@1 | NDCG@10 |
| Score | 22.32 | 9.851 | 23.46 | 17.89 | 10.23 | 23.98 | 19.32 | 10.67 | 23.12 |
| BM25 | 35.64 | 15.47 | 41.32 | 28.73 | 15.16 | 41.02 | 35.68 | 18.72 | 43.27 |
| Doc2Vec | 38.21 | 17.78 | 45.79 | 32.64 | 18.13 | 45.01 | 39.77 | 19.62 | 44.23 |
| CNTN | 45.06 | 27.78 | 51.03 | 43.37 | 25.31 | 52.38 | 47.32 | 29.11 | 55.31 |
| NeRank | 49.89 | 33.03 | 53.97 | 45.01 | 28.11 | 52.66 | 51.33 | 31.65 | 59.58 |
| TCQR | 41.96 | 27.78 | 49.06 | 39.46 | 26.01 | 43.51 | 44.25 | 26.67 | 52.51 |
| RMRN | 45.24 | 27.83 | 51.29 | 44.44 | 26.58 | 52.62 | 48.44 | 29.79 | 56.41 |
| UserEmb | 41.01 | 23.35 | 46.77 | 37.89 | 25.05 | 43.28 | 36.39 | 28.34 | 50.52 |
| **EFHM** | **51.25** | **33.33** | **56.21** | **46.23** | **30.65** | **53.31** | **52.26** | **36.17** | **60.33** |

**Table 4**
Expert finding results (%) of different methods on three datasets (i.e., History, English, Biology). The best performance of the baselines is underlined. Our method results are in bold.

| Datasets | History | | | English | | | Biology | | |
|---|---|---|---|---|---|---|---|---|---|
| Method | Metric | | | | | | | | |
| | MRR | P@1 | NDCG@10 | MRR | P@1 | NDCG@10 | MRR | P@1 | NDCG@10 |
| Score | 18.97 | 6.411 | 23.12 | 17.21 | 7.761 | 19.86 | 19.73 | 10.17 | 20.47 |
| BM25 | 25.91 | 15.41 | 33.78 | 20.13 | 14.21 | 35.11 | 27.12 | 14.86 | 33.22 |
| Doc2Vec | 28.43 | 17.43 | 42.35 | 23.26 | 15.23 | 35.22 | 29.12 | 16.08 | 35.19 |
| CNTN | 39.31 | 25.34 | 43.24 | 29.68 | 18.37 | 42.25 | 33.58 | 20.17 | 36.91 |
| NeRank | 46.75 | 27.73 | 56.43 | 48.95 | 27.16 | 56.41 | 41.71 | 23.86 | 49.91 |
| TCQR | 40.21 | 27.37 | 49.56 | 34.25 | 19.27 | 48.22 | 39.62 | 24.06 | 47.16 |
| RMRN | 52.21 | 33.51 | 61.26 | 46.77 | 25.22 | 56.75 | 43.62 | 24.53 | 50.87 |
| UserEmb | 40.39 | 26.39 | 44.53 | 31.73 | 19.56 | 45.51 | 32.23 | 19.87 | 35.96 |
| **EFHM** | **56.66** | **36.86** | **63.99** | **50.87** | **31.66** | **61.22** | **48.50** | **29.53** | **52.29** |

**Table 5**
Effect of the hierarchical matching network.

| Datasets | AI | | | Print | | | English | | |
|---|---|---|---|---|---|---|---|---|---|
| Variant | Metric | | | | | | | | |
| | MRR | P@1 | NDCG@10 | MRR | P@1 | NDCG@10 | MRR | P@1 | NDCG@10 |
| w/o Word | 47.69 | 31.12 | 52.82 | 47.17 | 32.01 | 57.15 | 47.88 | 29.76 | 55.29 |
| w/o Question | 46.39 | 29.78 | 49.74 | 49.67 | 33.28 | 58.01 | 48.86 | 29.05 | 56.13 |
| w/o Expert | 46.54 | 30.01 | 51.77 | 45.62 | 30.39 | 56.39 | 47.46 | 29.19 | 55.78 |
| **EFHM** | **51.25** | **33.33** | **56.21** | **52.26** | **36.17** | **60.33** | **50.87** | **31.66** | **61.22** |
| Datasets | Biology | | | Bioinformatics | | | History | | |
| Variant | Metric | | | | | | | | |
| | MRR | P@1 | NDCG@10 | MRR | P@1 | NDCG@10 | MRR | P@1 | NDCG@10 |
| w/o Word | 45.69 | 27.77 | 51.82 | 45.17 | 29.01 | 50.15 | 53.15 | 33.76 | 61.29 |
| w/o Question | 46.11 | 26.78 | 50.88 | 45.67 | 29.28 | 52.01 | 54.87 | 34.97 | 61.63 |
| w/o Expert | 46.23 | 25.89 | 51.77 | 45.62 | 30.39 | 51.39 | 54.67 | 35.19 | 62.78 |
| **EFHM** | **48.50** | **29.53** | **52.29** | **46.23** | **30.65** | **53.31** | **56.66** | **36.86** | **63.99** |

between them comprehensively and help seek experts more precisely. Although the two very competitive works NeRank [11] and RMRN [13] have achieved great performance, our method outperforms them. The reason is both of them integrate word embeddings to an abstract question-level vector, which is incapable of capturing the word-level fine-grained match signals. Overall, the experimental results meet our motivation in Section 1 and demonstrate the effectiveness of our method in expert finding.

### 4.3. Analysis of EFHM

In this section, we further explore the effects of the important components in our model in the following two aspects.

#### 4.3.1. Effect of the hierarchical matching network

The hierarchical matching network is the core to match the expert and the target question, which could capture multi-grained matching signals. Hence, we study the effect of the hierarchical matching network introduced in our method. We design three variants (*w/o Word*, *w/o Question*, and *w/o Expert*) that remove the three kinds of relevance ($S_w$, $S_q$ and $S_e$ in Eq. (14)) respectively. The experimental results are reported in Table 5. We can find that removing any components in the hierarchical matching network would degrade the model performance, which indicates the effectiveness of capturing multi-grained matching signals between experts and target questions. Besides, though the variants of the model lack different level matching signals, most of them still perform better than baselines. This is because there are various matching clues between candidate experts and target questions. And it is difficult that existing methods to employ a single expert embedding to capture these matching signals. This phenomenon consistences of the motivation demonstrated in Section 1 and verify the effectiveness of the architecture design of our model.

**Table 6**
Effect of the personalized attention mechanism.

| Dataset | AI | | | Print | | | English | | |
|---|---|---|---|---|---|---|---|---|---|
| Variant | Metric | | | | | | | | |
| | MRR | P@1 | NDCG@10 | MRR | P@1 | NDCG@10 | MRR | P@1 | NDCG@10 |
| EFHM-Mean | 46.69 | 30.13 | 51.91 | 46.37 | 31.31 | 56.55 | 46.78 | 28.15 | 52.13 |
| EFHM-Max | 47.39 | 31.01 | 52.74 | 46.67 | 31.98 | 55.28 | 48.76 | 30.36 | 55.36 |
| EFHM-Random | 49.52 | 31.78 | 53.77 | 48.62 | 33.39 | 57.01 | 50.67 | 31.19 | 57.71 |
| **EFHM** | **51.25** | **33.33** | **56.21** | **52.26** | **36.17** | **60.33** | **50.87** | **31.66** | **61.22** |

**Table 7**
Effect of the different signals controller. The recommended expert by the model is the ground truth. The controlling score represents $w'_w$, $w'_s$ and $w'_e$ respectively in order.

| Dataset | Target questions | Recommended expert | Controlling score |
|---|---|---|---|
| History | What were French actions against Germany during WW2? | Expert A | [0.240, 0.309, 0.451] |
| | What were the racial views of the Nazis towards the Greeks? | Expert B | [0.235, 0.426, 0.339] |
| Bioinformatics | Why would someone use a CRAM instead of a BAM? | Expert C | [0.402, 0.311, 0.287] |
| | How exactly is "effective length" used in FPKM calculated? | Expert D | [0.413, 0.298, 0.289] |

#### 4.3.2. Effect of the personalized attention mechanism

Next, we further study the effect of the personalized attention mechanism in our method. In this part, we design three model variants to verify their effects respectively: *EFHM-Mean* aggregates the expert historical features by a mean operator and combines that with the expert ID embedding as the overall expert feature. *EFHM-Max* uses a similar operation and only replaces the mean operator with a max operator. *EFHM-Random* initializes the query vectors of the personalized attention mechanism randomly instead of deriving from the expert ID. Table 6 shows the ablation results. We can find that utilizing the mean or max operator instead of the attention mechanism would degrade the performance even if we combine the expert ID embedding.

The reason could be that these operators are unable to select more important historical questions for modeling experts. And the results of *EFHM-Random* indicate the effectiveness of the expert ID embedding, which could take personalization into account. Furthermore, we can find that the results of *EFHM-Random* are superior to those of *EFHM-Mean* or *EFHM-Max*, this phenomenon indicates the necessity of the attention mechanism in the expert-level match encoder. In this way, the model could adaptively personalized capture more important historical information with respect to the specific expert.

#### 4.3.3. Effect of the different signals controller

In this section, we analysis the effect of the controller, which is designed to unify three-level relevance signals adaptively. We demonstrate the controlling scores of different relevance signals (i.e., $w'_w$, $w'_s$, $w'_e$) for better demonstrating the function of the controller intuitively. We select target questions from different datasets randomly, and save recommended experts by the model and corresponding controlling scores by Eq. (14). Table 7 shows the experimental results. We can find that for different datasets or different experts, the weights of relevant signals are different.

In History dataset, for the Expert A, the expert-level relevance signal is dominant for matching with the target question "What were French ... during WW2?" while the question-level relevance signal is dominant for the Expert B. The reason is that the Expert A has answered some questions about WW2, but the question-level semantic feature is vaguely similar to the target questions. For the second target question, the Expert B has answered one very similar question "What was the Nazi view towards other nations", which provides a question-level clue for the model to recommend the Expert B. Unlike the History dataset, in the Bioinformatics dataset, the word-level relevance signals are dominant to expert-question matching. The reason is that proper nouns (e.g., CRAM, FPKM, BAM, etc.) are hardly influenced by the context semantics in Bioinformatics questions, and these nouns are appearing frequently, which dominate the matching.

In a word, our model EFHM could dynamically and adaptively adjust controlling scores of different level relevance scores for computing the final score to complete expert-question matching.

### 4.4. Hyper-parameter analysis

In our model, the size of three-tier matching features (i.e., word-level feature, question-level feature, and expert-level feature) is important for capturing a multi-grained relationship between experts and target questions. In this section, we conduct hyper-parameter analysis experiments to explore the effects of three-tier feature sizes. As shown in Fig. 3, the overall trend of three-tier characteristics is that the performance of the model increases first and then decreases along with the feature size increases. When the feature size is too small, the feature vector may not adequately model the information needed for the question-expert match. However, when it is too large, the model may suffer from overfitting.

### 4.5. Efficiency analysis

In this section, we conduct experiments to explore the training efficiencies of **EFHM** and three recent state-of-the-art neural network baselines, i.e., NeRank, TCQR and RMRN. Firstly, we report the parameter numbers of different methods: (1) EFHM (**6.85M**), (2) TCQR (**7.20M**), (3) NeRank (**24.87M**), (4) RMRN (**31.49M**), **M** means million. In our experiments, these four models are trained on a single 24 GB-memory RTX 3090 GPU server with Intel(R) Xeon(R)@2.20 GHz CPU. The total runtime (i.e., training to convergence) results of each method are shown in Fig. 4. We can find that although TCQR has a smaller number of parameters than NeRank, it still takes more time to train because the original BERT weight already carries a large amount of corpus information while adapting it to CQA scenarios requires a longer training time. The RMRN equipped with Elmo obtains the slowest training speed. Finally, our method EFHM obtains the fastest training efficiency, which makes it more practical in real life.

### 4.6. Case study

In this section, we visualize the semantic relevance scores in the word- and question-level respectively to qualitatively study the behavior of our word- and question-level matching mechanism intuitively. In the word-level match encoder, we save the multiple relevance scores of words in histories under different

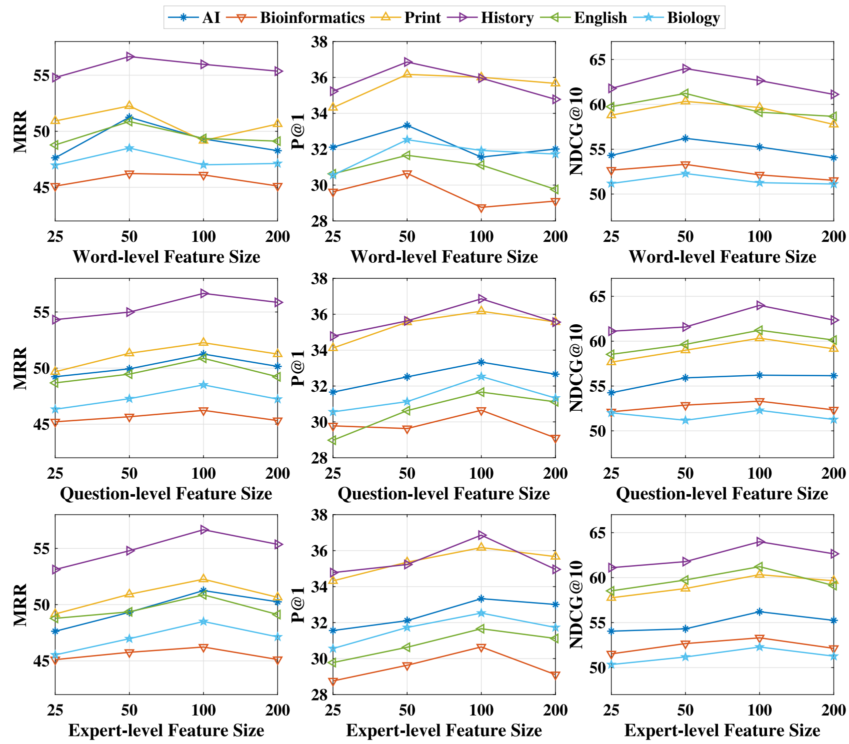


**Fig. 3.** The analysis of three-tier matching feature size.

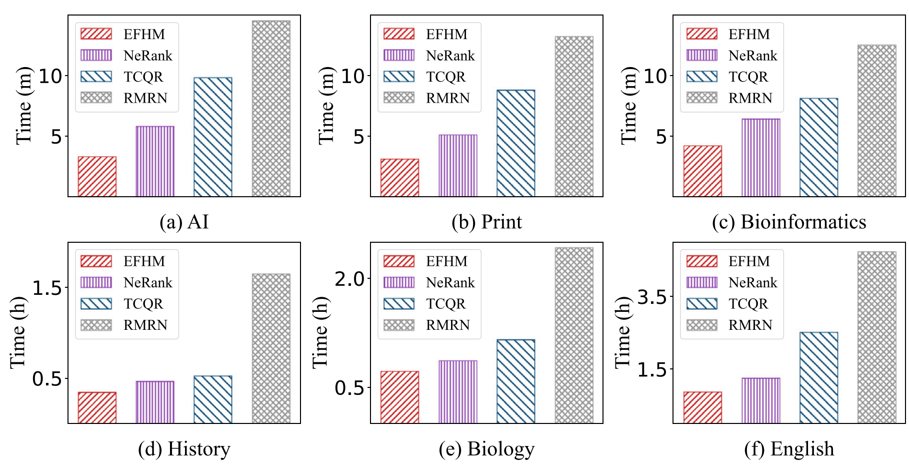


**Fig. 4.** The total runtime (i.e., training to convergence) comparisons of recent state-of-the-art methods and EFHM (minute/hour [m/h]).

**Table 8**
Case study. The expert historically answered questions (HQ).

| | |
|---|---|
| Historical Question-1 | Why did not Germany attack Us-ports during WW2 ? |
| Historical Question-2 | Why US pro China and anti Japan before WW2 ? |
| Historical Question-3 | What did men wear in night in Middle ages in Europe |
| Historical Question-4 | How common were marital duels in medieval Europe ? |
| Historical Question-5 | What is the date and original source of this medieval picture ? |

target questions by Eq (7) (removing the **max** operation), and in the question-level match encoder, we save the multiple relevance scores of expert historical answered questions under different target questions by Eq (9). We sample target questions and part of an expert historically answered questions from *History* datasets for the case study. The results are shown in Tables 8 and 9 respectively. Note that the deeper the color is, the larger the relevance is.

As shown in Fig. 5, we can see that the scores of each word in histories are different based on different target questions. For the TQ-1, the words "middle", "ages", "Europe" are most informative for expert-question matches while the word "Germany" and "WW2" contribute great deals to the matching with TQ-3. This indicates the effectiveness of the word-level matching encoder, i.e., it can support our model to capture word-level relevance signals for expert-question matches.

Afterward, we visualize the question-level relevance scores under different target questions shown in Fig. 6. It is apparent that the matching scores of each question answered by the expert are disparate when faced with different target questions.

**Table 9**
Case study. Target questions (TQ).

| | |
|---|---|
| Target Question-1 | Where did Medieval Europe's gold come from ? |
| Target Question-2 | Was the Wehrmacht a mechanized army ? |
| Target Question-3 | What were French actions against Germany during WW2 ? |
| Target Question-4 | Why did not Germany blockade the Strait of Gibraltar during WW2 ? |
| Target Question-5 | Percentage of helmet owners in 15th century Europe |

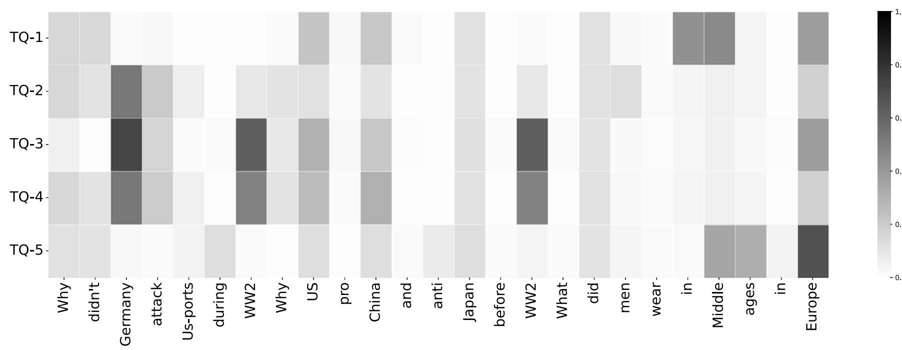


**Fig. 5.** Visualization for word-level semantic matching.

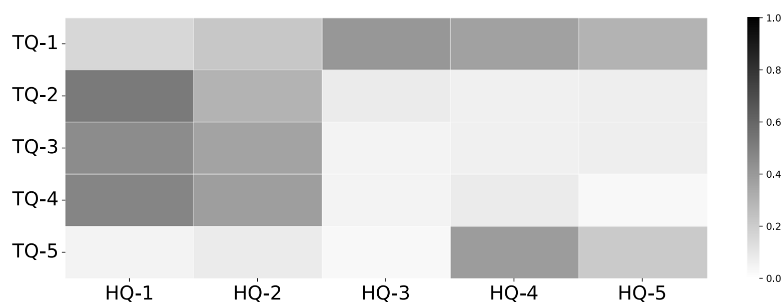


**Fig. 6.** Visualization for question-level semantic matching.

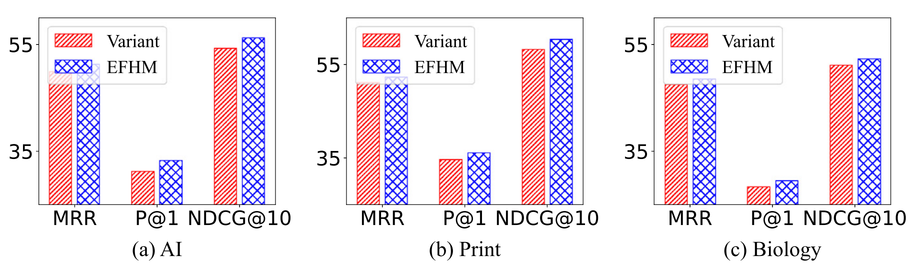


**Fig. 7.** Comparison results of the variant and EFHM.

The HQ-1 is most informative for matching with TQ-2. Nevertheless, under the TQ-5, the HQ-3 contributes the most to expert-question matches. This manifests the availability of the question-level matching mechanism, i.e., it can support our model to capture question-level matching signals based on different target questions. In a word, blending the different granularity matching mechanisms could help to provide more comprehensive matching clues for finding more suitable experts.

### 4.7. Discussion

Since the core of our proposed method is to effectively explore different granularity matching signals between the candidate expert and the target question, in this section, we further step into the inside of our model to discuss different strategies using different levels of target question information. Particularly, we design one model variant and conduct experiments on AI, Print, Biology datasets. The variant introduces the target question word-level features into the model, i.e., matching the target question word-level embedding matrix $\mathbf{S}^t$ with the expert historically answered questions word-level embedding matrices $(\mathbf{S}^1, \ldots, \mathbf{S}^n)$, then obtains the word-level matching signal via max operations. The encoders of the question-level match and expert-level match remain unchanged.

The results are shown in Fig. 7. We can find that replacing the question-level target question feature with the word-level target question feature degrades the model performance. The reason could be that there are many same articles (e.g., "the", "a", "an") and verbs (e.g., "are", "is", "do") causing large impacts on word-level matching. However, some keywords are only semantically similar but not identical (e.g., "Middle", "ages" and "Medieval" in Tables 8 and 9), which result in computing lower similarity scores. Furthermore, the question title is almost short (as shown in Table 2), and a single representation vector could express the semantics it contains. Hence, we employ the question-level target question representation to interact with the expert in different granularity, which could help the model to capture important word- and question-level match clues. In this way, the model could mitigate the effect of irrelevant words in the target question on word-level matching.

## 5. Conclusion

In this paper, we propose a hierarchical matching method for expert finding, named EFHM. Different from most existing methods that only focus on matching the experts and target questions, our proposed model elaborately designs a three-tier hierarchical matching network including word-level, question-level, and expert-level perspectives. In this way, the model could capture more comprehensive relationships between experts and target questions, and achieve more accurate expert finding on CQA websites. Extensive experiments are conducted on several real-world datasets and the results show that our method can effectively improve the performance of expert finding.

## CRediT authorship contribution statement

**Qiyao Peng:** Methodology, Software, Data curation, Writing – original draft. **Wenjun Wang:** Supervision. **Hongtao Liu:** Conceptualization, Investigation. **Yinghui Wang:** Revise Paper. **Hongyan Xu:** Visualization. **Minglai Shao:** Visualization.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgments

This work was supported by the China Postdoctoral Science Foundation (2022T150470, 2021M702448), the Intelligent Manufacturing Special Foundation of Tianjin, China (20201198) and the Key R&D Program of Tianjin, China (20YFZCSN01010).

## References


[1] S. Yuan, Y. Zhang, J. Tang, W. Hall, J.B. Cabotà, Expert finding in community question answering: a review, Artif. Intell. Rev. 53 (2) (2020) 843–874.

[2] J. Hu, S. Qian, Q. Fang, C. Xu, Hierarchical graph semantic pooling network for multi-modal community question answer matching, in: Proceedings of the 27th ACM International Conference on Multimedia, MM 2019, Nice, France, October 21-25, 2019, ACM, 2019, pp. 1157–1165.

[3] J. Hu, S. Qian, Q. Fang, C. Xu, Attentive interactive convolutional matching for community question answering in social multimedia, in: 2018 ACM Multimedia Conference on Multimedia Conference, MM 2018, Seoul, Republic of Korea, October 22-26, 2018, ACM, 2018, pp. 456–464.

[4] Y. Li, W. Shen, J. Gao, Y. Wang, Community question answering entity linking via leveraging auxiliary data, in: Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022, ijcai.org, 2022, pp. 2145–2151.

[5] J. Hu, Q. Fang, S. Qian, C. Xu, Multi-modal attentive graph pooling model for community question answer matching, in: MM '20: The 28th ACM International Conference on Multimedia, Virtual Event / Seattle, WA, USA, October 12-16, 2020, ACM, 2020, pp. 3505–3513.

[6] J. Hu, S. Qian, Q. Fang, C. Xu, Heterogeneous community question answering via social-aware multi-modal co-attention convolutional matching, IEEE Trans. Multimedia 23 (2021) 2321–2334.

[7] Z. Zhao, H. Lu, V. Zheng, D. Cai, X. He, Y. Zhuang, Community-based question answering via asymmetric multi-faceted ranking network learning, in: Proceedings of the International Conference on Artificial Intelligence, Vol. 31, (1) 2017.

[8] S. Chang, A. Pal, Routing questions for collaborative answering in community question answering, in: Proceedings of the International Conference on Advances in Social Networks Analysis and Mining, IEEE, 2013, pp. 494–501.

[9] H. Zhu, E. Chen, H. Xiong, H. Cao, J. Tian, Ranking user authority with relevant knowledge categories for expert finding, World Wide Web 17 (5) (2014) 1081–1107.

[10] Z. Zhao, L. Zhang, X. He, W. Ng, Expert finding for question answering via graph regularized matrix completion, IEEE Trans. Knowl. Data Eng. 27 (4) (2014) 993–1004.

[11] Z. Li, J.-Y. Jiang, Y. Sun, W. Wang, Personalized question routing via heterogeneous network embedding, in: Proceedings of the International Conference on Artificial Intelligence, Vol. 33, (01) 2019, pp. 192–199.

[12] X. Zhang, W. Cheng, B. Zong, Y. Chen, J. Xu, D. Li, H. Chen, Temporal context-aware representation learning for question routing, in: WSDM, 2020, pp. 753–761.

[13] J. Fu, Y. Li, Q. Zhang, Q. Wu, R. Ma, X. Huang, Y.-G. Jiang, Recurrent memory reasoning network for expert finding in community question answering, in: WSDM, 2020, pp. 187–195.

[14] N. Ghasemi, R. Fatourechi, S. Momtazi, User embedding for expert finding in community question answering, in: Proceedings of the ACM Transactions on Knowledge Discovery from Data, Vol. 15, (4) ACM New York, NY, USA, 2021, pp. 1–16.

[15] S. Robertson, H. Zaragoza, The Probabilistic Relevance Framework: BM25 and beyond, Foundations and Trends in Information Retrieval, 2009, pp. 333–389.

[16] T.C. Zhou, M.R. Lyu, I. King, A classification-based approach to question routing in community question answering, in: Proceedings of the International Conference on World Wide Web, 2012, pp. 783–790.

[17] X. Cao, G. Cong, B. Cui, C.S. Jensen, Q. Yuan, Approaches to exploring category information for question retrieval in community question-answer archives, ACM Trans. Inf. Syst. (TOIS) 30 (2) (2012) 1–38.

[18] A. Pal, F.M. Harper, J.A. Konstan, Exploring question selection bias to identify experts and potential experts in community question answering, ACM Trans. Inf. Syst. (TOIS) 30 (2) (2012) 1–28.

[19] M. Shahriari, S. Parekodi, R. Klamma, Community-aware ranking algorithms for expert identification in question-answer forums, in: Proceedings of the International Conference on Knowledge Technologies and Data-Driven Business, 2015, pp. 1–8.

[20] A. Huna, I. Srba, M. Bielikova, Exploiting content quality and question difficulty in CQA reputation systems, in: Proceedings of the International Conference and School on Network Science, Springer, 2016, pp. 68–81.

[21] A.J. Smola, B. Schölkopf, A tutorial on support vector regression, Stat. Comput. 14 (3) (2004) 199–222.

[22] D.M. Blei, A.Y. Ng, M.I. Jordan, Latent dirichlet allocation, J. Mach. Learn. Res. 3 (2003) 993–1022.

[23] A. Daud, J. Li, L. Zhou, F. Muhammad, Temporal expert finding through generalized time topic modeling, Knowl.-Based Syst. 23 (6) (2010) 615–625.

[24] F. Riahi, Z. Zolaktaf, M. Shafiei, E. Milios, Finding expert users in community question answering, in: Proceedings of the International Conference on World Wide Web, 2012, pp. 791–798.

[25] G. Zhou, K. Liu, J. Zhao, Joint relevance and answer quality learning for question routing in community qa, in: Proceedings of the ACM International Conference on Information and Knowledge Management, 2012, pp. 1492–1496.

[26] Z. Ji, B. Wang, Learning to rank for question routing in community question answering, in: Proceedings of the ACM International Conference on Information & Knowledge Management, 2013, pp. 2363–2368.

[27] L. Yang, M. Qiu, S. Gottipati, F. Zhu, J. Jiang, H. Sun, Z. Chen, Cqarank: jointly model topics and expertise in community question answering, in: Proceedings of the ACM International Conference on Information & Knowledge Management, 2013, pp. 99–108.

[28] G. Zhou, J. Zhao, T. He, W. Wu, An empirical study of topic-sensitive probabilistic model for expert finding in question answer communities, Knowl.-Based Syst. 66 (2014) 136–145.

[29] X. Liu, S. Ye, X. Li, Y. Luo, Y. Rao, Zhihurank: A topic-sensitive expert finding algorithm in community question answering websites, in: Proceedings of the International Conference on Web-Based Learning, Springer, 2015, pp. 165–173.

[30] X. Qiu, X. Huang, Convolutional neural tensor network architecture for community-based question answering, in: Proceedings of the International Joint Conference on Artificial Intelligence, 2015.
[31] D. Wang, E. Nyberg, A long short-term memory model for answer sentence selection in question answering, in: Proceedings of the International Conference on the Association for Computational Linguistics (Volume 2: Short Papers), 2015, pp. 707–712.
[32] G. Zhou, Y. Zhou, T. He, W. Wu, Learning semantic representation with neural networks for community question answering retrieval, Knowl.-Based Syst. 93 (2016) 75–83.
[33] S. Wan, Y. Lan, J. Xu, J. Guo, L. Pang, X. Cheng, Match-srnn: Modeling the recursive matching structure with spatial rnn, in: Proceedings of the International Joint Conference on Artificial Intelligence, 2016.
[34] Y. Qian, J. Tang, K. Wu, Weakly learning to match experts in online community, in: Proceedings of the International Joint Conference on Artificial Intelligence, 2016.
[35] Y. Sun, J. Han, X. Yan, P.S. Yu, T. Wu, Pathsim: Meta path-based top-k similarity search in heterogeneous information networks, Proc. VLDB Endow. 4 (11) (2011) 992–1003.
[36] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, in: Proceedings of the International Conference of Neural Information Processing Systems, 2017, pp. 5998–6008.
[37] P.-S. Huang, X. He, J. Gao, L. Deng, A. Acero, L. Heck, Learning deep structured semantic models for web search using click through data, in: Proceedings of the Conference on Information and Knowledge Management, 2013, pp. 2333–2338.
[38] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (1) (2014) 1929–1958.
[39] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, in: Proceedings of the International Conference on Learning Representations, 2015.