ExpertPLM: Pre-training Expert Representation for Expert Finding

Qiyao Peng
School of New Media Communication, Tianjin University, Tianjin, China
qypeng@tju.edu.cn

Hongtao Liu
Du Xiaoman Financial, Beijing, China
liuhongtao01@duxiaoman.com

Qing Yang
Du Xiaoman Financial, Beijing, China
yangqing@duxiaoman.com

Abstract

Expert Finding is an important task in Community Question Answering (CQA) platforms, which could help route questions to potential users to answer. The key is to learn representations of experts based on their historical answered questions accurately. In this paper, inspired by the strong text understanding ability of Pretrained Language modelings (PLMs), we propose a pre-training and fine-tuning expert finding framework. The core is that we design an expert-level pre-training paradigm, that effectively integrates expert interest and expertise simultaneously. Specifically different from the typical corpus-level pre-training, we treat each expert as the basic pre-training unit including all the historical answered question titles of the expert, which could fully indicate the expert interests for questions. Besides, we integrate the vote score information along with each answer of the expert into the pre-training phrase to model the expert ability explicitly. Finally, we propose a novel *reputation-augmented* Masked Language Model (MLM) pre-training strategy to capture the expert reputation information. In this way, our method could learn expert representation comprehensively, which then will be adopted and fine-tuned in the down-streaming expert-finding task. Extensive experimental results on six real-world CQA datasets demonstrate the effectiveness of our method.

1 Introduction

Community Question Answering (CQA) websites have become a popular platform, which can help people share their knowledge in the form of questions and answers. Some large portals such as Stack Exchange [1] have extremely attracted millions of users (Fu et al., 2020), which can raise their questions or post answers for questions they

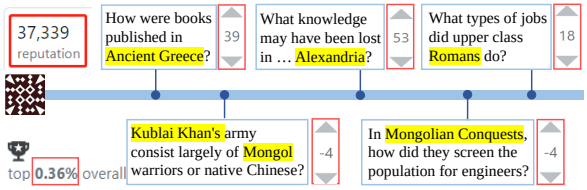

Figure 1: Several historical questions were answered by an expert (user ID *3353* in the History of StackExchange). The blue boxes represent original questions and the red boxes represent vote scores provided by the CQA community for answers. The higher the vote score of the answer, the more professional expertise. The reputation is *37,339*, which can reflect the expert overall capabilities (obtains top **0.36%** in History domain).

are interested in or good at. Due to the large participation, there are too many questions to wait for answers (Zhao et al., 2017; Yuan et al., 2020). Hence, it is a great challenge to route questions to a suitable expert for providing satisfactory answers (Chang and Pal, 2013; Zhao et al., 2014). Expert finding in CQA websites can effectively route questions and help raisers receive high-quality answers quickly, which has attracted considerable attention recently.

Generally speaking, accurate learning expert representation is the central problem in expert finding. Most existing methods usually infer expert interest representation based on her/his historical answered questions, then measure the matching score between experts and the new questions. For example, PMEF (Peng et al., 2022a) designs a title-body-tag multi-view paradigm to learn representations of questions and experts respectively. It is noted that most existing methods focus on modeling expert interests and ignore whether the expert has the ability to answer the question, i.e., the expertise.

Recently, Pretrained Language Models (PLMs), e.g., BERT, pre-train general corpus-level language knowledge and fine-tune on the downstream task, which have achieved great success in various areas (Wu et al., 2021; Qiu et al., 2021). Motivated by this, **"Can we pre-train expert-level represen-**

[1] https://stackexchange.com

Qiyao Peng and Hongtao Liu are equal contribution.
Qing Yang is the corresponding author.

Findings of the Association for Computational Linguistics: EMNLP 2022, pages 1043–1052
December 7-11, 2022

tation on CQA domains, and then fine-tune on the downstream expert finding task?" Different from corpus-level pre-training in Natural Language Processing (NLP), which focuses on learning general language knowledge, expert pre-training needs to consider the following two core capabilities:

1) *Interest modeling*. We could infer the expert interests from the historical answered questions. As shown in Figure 1, the expert has answered multiple questions related to "Ancient Greece" and "Alexandria", which reflect that his interest about the history of Ancient Greece. However, simply adopting the existing PLMs or further pre-training over the CQA corpus could not effectively capture the expert-level interest.

2) *Expertise modeling*. The expertise of experts plays an important role in expert findings. From Figure 1, we can find that the expert is interested in Ancient Greece and Mongol. And the vote scores obtained for these two types of questions are very different (e.g., $+39$ and -4), which indicate the expert different expertise of different questions. However, most existing PLMs fail to model the ability of experts in answering different questions.

Hence it is necessary to design more effective pre-training framework for learning comprehensive expert representations.

For alleviating these gaps, we propose an **Expert**-level **P**re-training **L**anguage **M**odel for expert finding (**ExpertPLM**), which could pre-train expert representation effectively. We empower the typical corpus-level pre-training paradigm in the following aspects: (1) *Expert interest modeling*. We re-construct the model input via aggregating the expert historical answered questions for pre-training. Compared with the corpus-level input paradigm (i.e., one line-one sentence) of PLMs, our approach employs expert-level input, which could learn more comprehensive interest features based on histories during pre-training. (2) *Expert abilities modeling*. Unlike modeling expert interest, the expert ability is not explicitly reflected in historical answered questions. Fortunately, the vote score the answer received could indicate CQA user satisfaction with the answer, which could reflect the ability to answer this question of the expert (i.e., the higher vote score, the higher expertise). Hence, we encode the vote score and integrate that with the corresponding historical answered question input embedding to indicate the expert ability for the question.

To further prompt the expertise learning, we introduce the expert reputation shown in Figure 1 which could indicate the expert overall ability and design a **reputation-augmented** Masked Language Model (MLM) pre-training strategy to capture the expert reputation information. In this way, our method could pre-train expert representation including interest and expertise effectively. In the fine-tuning we utilize the weight to encode expert and question, then accomplish the downstream expert finding task via a fine-tuning way.

In summary, the contributions of our method are:

- We propose a novel expert-level pre-training language model for the expert finding task in CQA websites, which could effectively pre-train the expert representations.
- We unify the historical question titles, vote scores during pre-training and design a reputation-augmented MLM task to empower the model for capturing the interest and expertise of experts.
- Extensive experiments on six real-world datasets show that our method could achieve better performance than existing baselines and validate the effectiveness of our approach ExpertPLM.

2 Related Works

In this section, we briefly review some related works about Pre-training for NLP, Pre-training for RS, and Expert Finding.

2.1 Pre-training for NLP

There is a long history of pre-training general language representations. Earlier methods, such as Word2Vec (Mikolov et al., 2013), Glove (Pennington et al., 2014) learned the word embedding by capturing word co-occurrence information, which can offer a significant improvement in various tasks. However, these methods were incapable of considering contextual information. Recently, a series of pre-training methods based on the Transformers (Vaswani et al., 2017), such as BERT (Devlin et al., 2018), BART (Lewis et al., 2020), have changed original training paradigm. Through pre-training and fine-tuning paradigms, they can jointly capture general language knowledge on a large corpus of text and task-specific knowledge, which could improve downstream task performance.

2.2 Pre-training for RS

Recently, some recommendation tasks (Sun et al., 2019; Wu et al., 2020) employ the pre-training technology to learn item co-occurrence information for improving recommendation performance. For example, BERT4Rec (Sun et al., 2019) employed the Cloze task to predict the masked items using the left and right context based on the Transformer structure, which could capture the contextual user interaction representations. Then, the model was fine-tuned on the pre-trained encoder to accomplish the next item recommendation, which obtained better performance.

However, different from the general recommendation task, the expert finding in CQA is always cold-start and have quite unique characteristics, such as the expertise modelling of experts.

2.3 Expert Finding

In CQA websites, expert finding aims to find capable experts for providing satisfactory answers to questions (Yuan et al., 2020; Liu et al.; Peng et al., 2022b). The majority of previous works fall into two categories: traditional methods and deep learning-based methods. Traditional methods mostly employed feature-engineering or topic-modeling to model the questions and experts and then routed questions to suitable experts. For example, Yang et al. (Yang et al., 2013) proposed a topic expertise model to jointly model expert topical interests and expertise for help better recommending. Deep learning-based methods employ the neural network to model experts and measure the matching relevance with target questions (Li et al., 2019; Fu et al., 2020; Ghasemi et al., 2021). For example, TCQR (Zhang et al., 2020) employed a question encoder to learn question words and learned the answerers' representation in the context of both the semantic and temporal information for expert representation learning.

3 Problem definition

In this section, we formulate the problem of expert finding in CQA websites. Suppose that there is a target question q^t and a candidate expert set $C^u = \{c_1^u, \cdots, c_M^u\}$ respectively, where M is the number of experts. Given a candidate expert $c_i^u \in C^u$ with r_i^u as the reputation, she/he is associated with a set of her/his historical answered questions, which can be denoted as $Q_i^u = \{q_1, \cdots, q_n\}$ where n is the number of historical questions. And vote scores corresponding to the expert historical answered questions can be denoted as $V_i^u = \{v_1, \cdots, v_n\}$. The question is represented by a question title, which consists of a sequence of words. The primary objective of the expert finding is to predict the most suitable expert for answering the target question. Note that the expert who provides the "accepted answer" for the question will be regarded as the ground truth. It is noted that one question only have one "accepted answer".

4 Proposed Method

In this section, we will introduce our method *ExpertPLM* in detail. The expert pre-training language model is demonstrated in Figure 2. In the pre-training stage, we pre-train the model based on the concatenated expert historical answered questions (i.e., one input line is one expert all historical question titles) from different CQA domains for capturing expert interest. For indicating the expert different abilities to answer different questions, we integrate the vote score embedding with the corresponding question input embedding. Furthermore, we design a reputation-augmented MLM pre-training task for capturing the expert overall expertise and CQA language knowledge. In the fine-tuning stage, as shown in Figure 3, via conducting the supervised expert finding task on expert and question representations generated by the pre-trained weight, we can obtain an improved expert finding model for a specific domain.

4.1 Pre-training Expert Representation

The goals of the pre-training stage are: 1) teaching ExpertPLM how to capture the expert interest and expertise; 2) learning general CQA domain language knowledge. Next, we will introduce the ExpertPLM from Input Layer, Model Architecture and Pre-training Task three aspects.

Input Embedding As shown in Figure 2, for empowering the BERT to model expert interest, we simply concatenate the words of the expert's historical answered questions into a whole sequence as one expert-level input. Then, we add the special tokens `[CLS]` and `[SEP]` at the begining and end of the input word sequence respectively. Furthermore, for distinguishing different historical answered questions, we add the special token `[HSEP]` between the histories (e.g., `[HSEP]` between the q_1 and q_2).

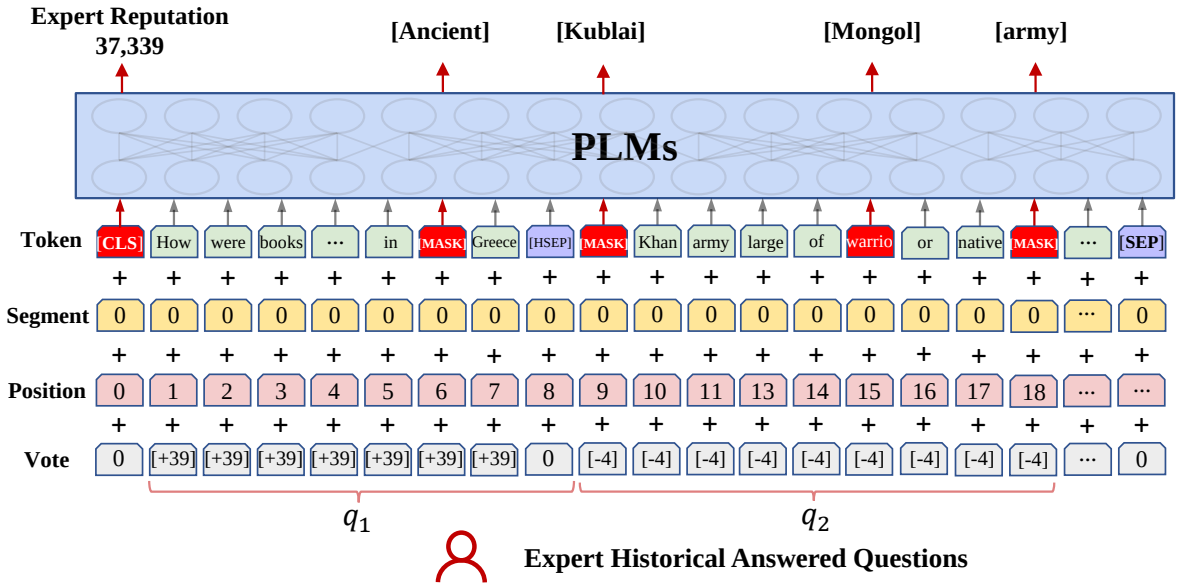

Figure 2: ExpertPLM **Pre-training Framework**. Expert historical answered questions (q_1, q_2, $\cdots$) and vote scores are aligned as the input for indicating the expert interest and expertise. The first token `[CLS]` will be always masked to pretrain the user reputation during pre-training.

Given an expert-level input, considering that the original pre-trained BERT weight (e.g., `bert-base-uncased`) has already carried a great deal of language knowledge, we utilize that to initialize the token, segment and position embeddings. Hence, the input token representation matrix $\mathbf{E}_t$ is constructed by summing its corresponding token, segment and position embedding:

$$\mathbf{E}_t = \mathbf{E}_{token} + \mathbf{E}_{seg} + \mathbf{E}_{pos} \ . \quad (1)$$

Furthermore, the vote scores an expert has received could indicate his/her expertise to answer different questions. Generally speaking, the higher the vote score the answer received, the more satisfied the community is with the answer, and the answerer has the stronger professional expertise to answer such questions. For example, as shown in Figure 2, the vote score of answer (i.e., -4) about *Mongol* represent the expert may lack the ability to answer *Mongol* related questions, but have much expertise to answer *Ancient Greece* related questions (vote score: $+39$). Hence, we introduce the vote scores corresponding to each historical question for measuring the expert abilities in different question fields effectively. We encode the normalized vote score and integrate that with the input representation $\mathbf{E}_t$ as follows:

$$\mathbf{E}_{in} = \mathbf{E}_t + \mathbf{E}_v \ . \quad (2)$$

It is mentioned that the dimension of vote score embedding coincides with the corresponding historical question. In other words, for the historical question q_1 and the corresponding vote score v_1, the dimension of vote score embedding is mapped to the same dimension as the question q_1 input embedding. In this way, the BERT model could capture the expert interests and expertise for different questions.

BERT Layer BERT model architecture consists of multi-layer bidirectional Transformer encoder layers. Each Transformer encoder layer has the following two major sub-layers, i.e., multi-head self-attention and position-wise feed-forward. Let $\mathbf{E}^l_{in}$ denote the input representation of the $(l+1)$-th Transformer encoder layer. We omit the layer subscript l of each parameter for convenience.

Multi-Head Self-Attention. This sub-layer aims to capture the contextual representations for each word. The self-attention function is defined as:

$$Att(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = Softmax(\mathbf{Q}\mathbf{K}^T/\sqrt{d})\mathbf{V} \ , \quad (3)$$

where $\mathbf{Q}$, $\mathbf{K}$ and $\mathbf{V}$ represent the query, key and value matrix correspondingly. Multi-head self-attention layer $MH(\cdot)$ will project the input to multiple sub-spaces and capture the interaction information, which is denoted as:

$$MH(\mathbf{E}_{in}) = [head_1; \cdots; head_h]\mathbf{W} \ , \quad (4)$$

$$head_i = Att(\mathbf{E}\mathbf{W}^q_i, \mathbf{E}\mathbf{W}^w_i, \mathbf{E}\mathbf{W}^v_i) \ , \quad (5)$$

where $\mathbf{W}^q_i, \mathbf{W}^w_i, \mathbf{W}^v_i \in \mathcal{R}^{d\times\frac{d}{h}}$ and $\mathbf{W} \in \mathcal{R}^{d\times d}$ are parameters. Via the multi-head self-attention,

the input representation $\mathbf{E}$ is transformed to $\mathbf{H} \in \mathcal{R}^{n \times d}$, where n is the token number.

Position-wise feed-forward. For the input $\mathbf{H}$, the calculation is defined as:

$$FFN(\mathbf{H}) = RELU(\mathbf{H}\mathbf{W}_1^f + b_1^f)\mathbf{W}_2^f + b_2^f \,, \quad (6)$$

where $\mathbf{W}_1^f$, $\mathbf{W}_2^f$ and b_1^f, b_2^f are learnable parameters. Furthermore, the residual connection is introduced into each of the two sub-layers, and layer normalization is applied to each sub-layer.

4.2 Pre-training Task

In this section, we will present the reputation-augmented MLM training task in the pre-training stage, which is the core task to enforce the PLMs to model expert abilities and capture the CQA language knowledge.

Firstly, considering that the original MLM task has the power capable ability to train the PLMs, we adopt the MLM task to learn the language knowledge in CQA scenario, which is first randomly masking some words and then using the bidirectional context information to re-construct the input sequence. However, the origin MLM is only for the input corpus and would be incapable to learn the expert-level features (e.g., the ability).

As denoted above, the reputation the expert received (as shown in Figure 1) in CQA could reflect the overall expertise in answering questions. Generally speaking, the higher reputation, the higher expertise, and the answer provided by the expert would be more satisfied users from CQA community. Hence, we design a reputation-augmented MLM task to pre-train the model for empowering the model to capture the overall expertise of experts. Specifically, given the example input illustrated in Figure 2, the output of the special token `[CLS]` could capture the whole input sequence information. Hence, we adopt the token `[CLS]` as a special indicator to predict the expert reputation. We normalize all expert reputations to $0-11$, and transform them as special tokens (e.g., $[0]-[11]$) for convenient prediction. It is noted that the reputation `[CLS]` token is always masked during the pre-training phase.

In this way, our model ExpertPLM could pre-train expert-level representation containing expert interests and capabilities, which are beneficial to downstream expert finding task.

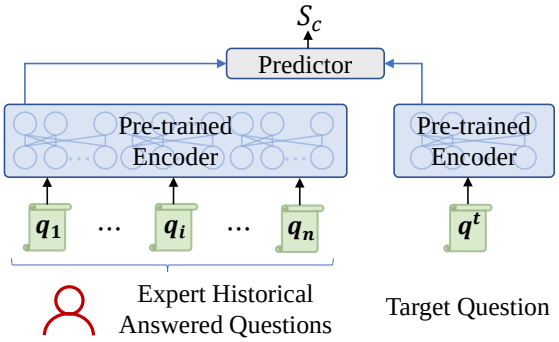

Figure 3: ExpertPLM Fine-tuning.

4.3 Fine-tuning for Expert finding

Though the ExpertPLM has pre-trained the expert-level representation, the downstream task is still slightly different from the pre-training, since it focuses not only on modeling expert but also on modeling interaction between expert and target question. Considering the pre-trained model based MLM can naturally capture the CQA language knowledge, we use the pre-trained model to learn the features of questions. As illustrated in Figure 3, we enter the expert historical answered questions and the target question into two same pre-trained encoders separately. Then, we concatenate two `[CLS]` representations for predicting the matching score S_c between the expert and the target question. We employ negative sampling technology (Huang et al., 2013) and the cross-entropy loss to fine-tune our model as follows:

$$S_c = \frac{\exp(S_c)}{\sum_{j=1}^{K+1}\exp(S_j)} \,, Loss = -\sum_{c=1}^{K+1} \hat{S}_c log(S_c) \,, \quad (7)$$

where $\hat{S}_c$ is the ground truth label and S_c is the normalized probability predicted by the model.

5 Experiments

5.1 Datasets and Experimental Settings

We construct a dataset containing 103,005 expert-level input data for pre-training expert representation, which is from StackExchange[2]. For fine-tuning and verifying the effect of the model in specific domains, we select six different domains, i.e., **English**, **Biology**, **Es**, **Electronics**, **Gis** and **CodeReview**. Each dataset includes a question set, in which, each question is associated with its title, an "accepted answer" among several answers provided by different answerers. And the provider of

[2]https://archive.org/details/stackexchange

Dateset	English				Gis				CodeReview			
Method \ Metric	MRR	P@1	P@3	NDCG@20	MRR	P@1	P@3	NDCG@20	MRR	P@1	P@3	NDCG@20
Doc2Vec	0.2326	0.1523	0.2946	0.3522	0.2912	0.1608	0.2873	0.3519	0.3015	0.1731	0.3199	0.4246
CNTN	0.2968	0.1837	0.3936	0.4225	0.4012	0.2312	0.4015	0.4201	0.3553	0.2013	0.5077	0.5035
NeRank	0.4895	0.2716	0.6143	0.5641	0.4697	0.3032	0.5577	0.5836	0.4947	0.3089	0.6055	0.6154
TCQR	0.3425	0.1927	0.4987	0.4822	0.4553	0.2634	0.5489	0.5637	0.4253	0.2112	0.5382	0.5839
RMRN	0.4677	0.2522	0.6162	0.5675	0.4897	0.3239	0.5777	0.5832	0.4311	0.2517	0.5580	0.5892
UserEmb	0.3173	0.1956	0.4236	0.4551	0.3223	0.2433	0.4477	0.4562	0.3915	0.2031	0.5126	0.5246
PMEF	0.4947	0.2865	0.6314	0.5875	0.4861	0.3311	0.5888	0.5911	0.5020	0.3139	0.6161	0.6170
ExpertPLM	**0.5250**	**0.3168**	**0.6512**	**0.6117**	**0.5085**	**0.3456**	**0.6058**	**0.6178**	**0.5115**	**0.3263**	**0.6313**	**0.6242**
Dateset	Es				Biology				Electronics			
Method \ Metric	MRR	P@1	P@3	NDCG@20	MRR	P@1	P@3	NDCG@20	MRR	P@1	P@3	NDCG@20
Doc2Vec	0.3587	0.2557	0.3613	0.3821	0.3561	0.2146	0.3699	0.4793	0.3871	0.2693	0.4138	0.4863
CNTN	0.3897	0.2720	0.4078	0.4693	0.3358	0.2017	0.3369	0.3691	0.4952	0.2740	0.6538	0.6146
NeRank	0.5647	0.3932	0.6413	0.6617	0.4371	0.2886	0.5361	0.4991	0.5105	0.3459	0.5976	0.6221
TCQR	0.4716	0.3059	0.5941	0.6053	0.3962	0.2406	0.4422	0.4716	0.5016	0.3233	0.5953	0.6004
RMRN	0.5516	0.3761	0.6304	0.6453	0.4662	0.3153	0.5562	0.5578	0.5622	0.4038	0.6759	0.6729
UserEmb	0.4703	0.3032	0.5735	0.4869	0.3497	0.2263	0.3675	0.4672	0.4175	0.2954	0.4541	0.5170
PMEF	0.5735	0.4269	0.6520	0.6661	0.4719	0.3216	0.5693	0.5688	0.5872	0.4238	0.6950	0.6829
ExpertPLM	**0.5840**	**0.4380**	**0.6588**	**0.6793**	**0.4956**	**0.3353**	**0.5868**	**0.5929**	**0.6003**	**0.4356**	**0.7025**	**0.7046**

Table 1: Expert finding results of different methods. The best performance of the baselines is underlined. We perform t-test and the results show that **ExpertPLM** outperforms other baselines at significance level p-value<0.05.

Datasets	# questions	# answerers	# answers
Es	36,271	3,260	50,801
Gis	50,718	3,168	70,034
Biology	8,704	630	11,411
English	46,692	4,781	104,453
Electronics	56,614	3,084	102,214
CodeReview	36,947	2,242	57,622

Table 2: Statistical details of the datasets.

Datasets	Biology		English	
	Vote score	Reputation	Vote score	Reputation
Max	+287	65,677	+828	140,445
Min	-8	1	-69	1
Avg	3.49	81.46	3.38	75.25

Table 3: Biology and English datasets. Statistical details of Vote score and Reputation. It is noted that, in CQA websites, the initial value of Reputation is 100.

the "accepted answer" is the ground truth expert. We follow the preprocessing method in previous work (Peng et al., 2022a). 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 the ratios 80%, 10%, 10% respectively in chronological order.

We adopt the pre-trained weight **bert-base-uncased** as the base model. The ExpertPLM pre-training weight contains 110M parameters. To alleviate the overfitting problem, we utilize dropout technology (Srivastava et al., 2014) and set the dropout ratio as 0.2. We adopt Adam (Kingma and Ba, 2015) optimization strategy to optimize our model and set the learning rate to $5e$-5 in further pre-training and $5e$-2 in fine-tuning. We independently repeat each experiment 5 times and report the average results. All experiments are implemented using Pytorch frame and using two 24GB-memory RTX 3090 GPU servers with Intel(R) Xeon(R)@2.20GHz CPU. Our code, pre-trained weight and the validation data are anonymously available on the Dropbox [3].

We briefly list statistical information of vote score and reputation in two datasets as examples, which is shown in Table 3. Since the vote score and the reputation exhibit similar characteristics, we only describe the pre-processing process for vote score, and the reputation is pre-processed in a similar way. First, we perform an overall translation of the vote score to eliminate negative numbers. Then, we perform a logarithmic operation on the vote score (i.e., $ln(\cdot)$) to mitigate the effects of excessive variance. To facilitate model calculation and make the number of scores contained in each score segment is approximately similar, we normalize the vote score to integer between 1 and 10, which is calculated as follows:

$$v_{min} = min(V^u), v_{max} = max(V^u)\ , \qquad (8)$$

where v_{min} and v_{max} represent the minimum score

[3] https://github.com/pengqy/EMNLP2022_ExpertPLM

and the maximum score in vote score sequence V^u.

$$v_i^* = round(\frac{9}{v_{max} - v_{min}} * (v_i - v_{min}) + 1) \quad (9)$$

where v_i^* is the normalized score of the i-th vote score v_i in V^u, $round$ is the rounding operator.

5.2 Baselines and Evaluation metrics

We compare our method ExpertPLM with recent competitive methods including: **(1) Doc2Vec** selects experts who have previously answered questions relevant to the target question. **(2) CNTN** (Qiu and Huang, 2015) employs the CNN to model questions and computes ranking scores between questions and experts. **(3) NeRank** (Li et al., 2019) learns question, raiser and expert representations via a HIN embedding algorithm and utilizes the CNN to match them. **(4) TCQR** (Zhang et al., 2020) utilizes a temporal context-aware model in multiple temporal granularities to learn the temporal-aware expert representations. **(5) RMRN** (Fu et al., 2020) equips with a recurrent memory reasoning network to explore the implicit relevance between expert and question. **(6) UserEmb** (Ghasemi et al., 2021) utilizes a node2vec to capture social features and uses a word2vec to capture semantic features, then integrates them to improve the expert finding. **(7) PMEF** (Peng et al., 2022a) designs a personalized expert finding method under a multi-view paradigm, which could comprehensively model expert and question. The evaluation metrics include Mean Reciprocal Rank (MRR) (Craswell, 2009), P@1 (i.e., Precision@1), P@3 (i.e., Precision@3) and Normalized Discounted Cumulative Gain (NDCG@20) (Järvelin and Kekäläinen, 2002) to verify the expert ranking quality.

5.3 Performance Comparison

We report experimental results of ExpertPLM and other comparative methods in Table 1. There are some findings in these results. Some earlier methods (e.g., Doc2Vec, CNTN) obtain poor results on almost datasets, the reason may be that they usually employ max or mean operation on histories to model expert, which omits different history importance. On the contrary, recent methods (e.g., RMRN, PMEF, etc.) achieve better results on different datasets, which is due to these methods focusing on modeling the different interest for different questions.

As we can see, our model *ExpertPLM* outperforms other comparative methods and achieves great improvements. Our method introduces the expert-level representation pre-training mechanism to pre-train the expert interests and expertise for different questions on different CQA domains. Via pre-training, the expert representation can be roughly captured by the model, which could be beneficial to the downstream expert finding task. Meanwhile, this paradigm captures general CQA language knowledge during pre-training, which could enhance the modeling of questions and experts in the downstream task and yield better performance.

5.4 Ablation Study

To highlight the effectiveness of our designing reputation-augmented MLM pre-training task, we design three model variants: **(a) Only Cm**, adopt corpus-level MLM to pre-train over CQA corpus (i.e., one question title one input line) and then fine-tune instead of the expert-level pre-training; **(b) Only Em**, adopt expert-level input for MLM pre-training but remove the vote score information and the reputation task during expert-level pre-training; **(c) w/o Rep**, adopt expert-level MLM task for pre-training but remove the reputation task during expert-level pre-training.

As shown in Table 4, we can have the following observations: (1) **Only Em** outperforms **Only Cm**. This is because **Only Em** employs expert-level input and pre-trains specifically for experts, and hence it could learn the more precise representation of experts compared with the corpus-level pretraining in **Only Cm**. (2) **w/o Rep** outperforms **Only Em**. Compared with **Only Em**, the **w/o Rep** introduces the vote score information additionally to pre-train the expert in different abilities to answer different questions, which is the core of downstream task. (3) Our complete model **ExpertPLM** obtains the best results. The reason is it can pre-train the expert-level representations including interest and expertise. Further, the pre-trained model also captures the CQA language knowledge, which could yield better performance. In all, the results of ablation studies meet our motivation and validate the effectiveness of our proposed pre-training task.

5.5 Pre-trained Weight Analysis

In this section, we conduct two experiments to further explore the influences of the pre-trained weight in the following two aspects through comparing with the origin Bert weight.

Dateset	Es				Gis				CodeReview			
Method \ Metric	MRR	P@1	P@3	NDCG@20	MRR	P@1	P@3	NDCG@20	MRR	P@1	P@3	NDCG@20
Only Cm	0.5671	0.4084	0.6398	0.6567	0.4827	0.3239	0.5866	0.5967	0.4965	0.3071	0.6017	0.6034
Only Em	0.5763	0.4252	0.6476	0.6635	0.4933	0.3298	0.6011	0.6011	0.5011	0.3103	0.6157	0.6118
w/o Rep	0.5768	0.4311	0.6522	0.6719	0.5018	0.3376	0.6054	0.6129	0.5089	0.3189	0.6205	0.6196
ExpertPLM	**0.5840**	**0.4380**	**0.6588**	**0.6793**	**0.5085**	**0.3456**	**0.6058**	**0.6178**	**0.5115**	**0.3263**	**0.6313**	**0.6242**

Table 4: The variants of ExpertPLM pre-training experiment results.

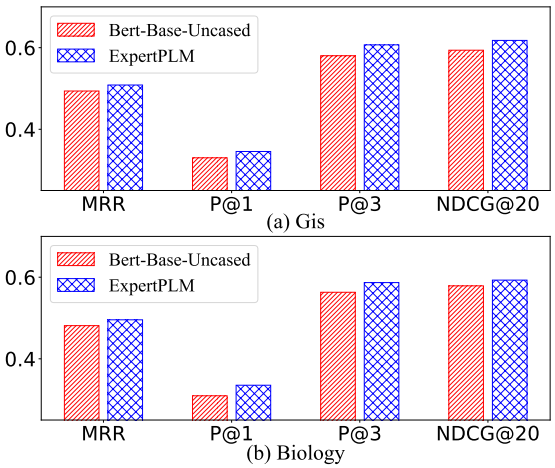

Figure 4: Effects of ExpertPLM Pre-training Weight.

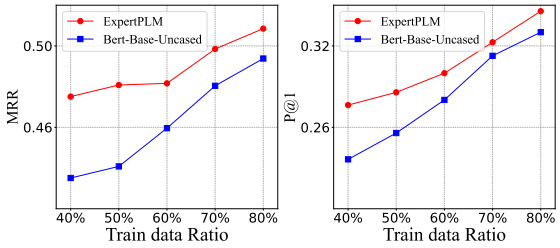

Figure 5: Impacts of Train Data Ratio (Gis).

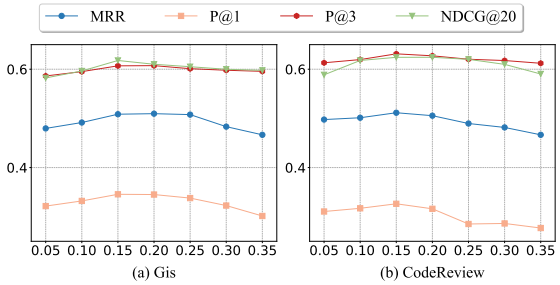

Figure 6: Impacts of Different Mask Ratios.

Effect of Pre-trained Weight In our method, we employ the ExpertPLM pre-training weight to accomplish expert finding via a fine-tuning way. Hence, we will explore the effectiveness of the pre-training weight in this section. We replace the weight directly with the original `bert-base-uncased` in the fine-tuning stage, i.e., we employ two bert-base-uncased weights to learn the expert and question representations respectively, then compute the matching score.

The results are illustrated in Figure 4. We find that the ExpertPLM is useful for the downstream expert finding task. As denoted above, accurately learning expert representation is a critical task for expert finding as it can encode expert interest and expertise for answering different questions. Compared with ExpertPLM, Bert-Base-Uncased could not capture such expert characteristics, which reduces the performance of the downstream expert finding task. This observation validates the effectiveness of our ExpertPLM pre-training weight.

Effect of Train Data Ratio in Fine-tuning In this part, we adjust the training data ratio in the fine-tuning stage to explore the effect of different data ratio on model training. We employ $[40\%, 50\%, 60\%, 70\%, 80\%]$ all data in Gis dataset as training data to fine-tune our model, meanwhile, the ratios of validation data and testing remain the same (i.e., 10%) with the main experiments.

As shown in Figure 5. We can find that there are growing gaps between the results of ExpertPLM and Bert-Base-Uncased with the reduction of training data, which indicates the advantage of the pre-trained expert model is larger when the training data is more scarce. This may be because the ExpertPLM can exploit expert histories and vote scores to capture the expert interest and expertise during pre-training phase, which could reduce the dependency on training data during the fine-tuning stage. And the Bert-Base-Uncased could be incapable of capturing expert-level representation, which could be affected by the ratio of training data in the fine-tuning.

5.6 Effect of Mask Ratio in Pre-training

The mask ratio is an important hyperparameter of ExpertPLM during pre-training and we have varied the ratio in $[0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35]$ for

exploring the parameter sensitivity of the Masked Language Model in pre-training.

The results are shown in Figure 6. We can find that all metric results increase at first as the ratio of masked token increases, and reach the maximum value (i.e., the best model performance), and then degrades. When the masked ratio is small, the BERT model could not capture adequate CQA language knowledge, which could reduce the performance of downstream tasks. In the contrary, when the mask ratio is large, the `[MASK]` symbol appears in pre-training stage more frequently, which could intensify the mismatch between pre-training and fine-tuning. Hence, we set up the mask ratio to 0.15 during the pre-training stage.

6 Conclusion

In this paper, we propose ExpertPLM, a pre-training language model for the expert finding task in CQA. The core of our method is that we design an expert-specific pre-training framework based on a masked language model, towards precisely modeling experts (i.e., interest and expertise) based on the historical answered questions and vote scores. Meanwhile, the pre-trained language model could capture the CQA language knowledge, which is beneficial to the downstream task. We conduct detailed experiments on real world CQA datasets, and the results fully validate the effectiveness of our proposed pretraining method, In the future, we would like to explore a larger scale comprehensive expert pre-training model and extend the pre-trained model to more downstream tasks.

7 Limitation

Although our model has achieved excellent performance, there may be some limitations in this study that could be addressed in future research. First, the existing pre-training dataset is still a little small, which would lead to inadequate pre-training. In the future, we will construct a larger pre-training dataset for larger-scale CQA pre-training. Second, some users have more historical answered questions, which will cause that the input sequence length is greater than 512. In the future, we will explore CQA pre-training based on long-sequence modelling. Third, during pre-training, we only design the expertise learning task for users. In the future, we will explore to introduce more user modeling tasks (e.g., interest modeling) during pre-training.

References

Shuo Chang and Aditya Pal. 2013. Routing questions for collaborative answering in community question answering. In *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, ASONAM '13, page 494–501, New York, NY, USA. Association for Computing Machinery.

Nick Craswell. 2009. Mean reciprocal rank. In *Encyclopedia of Database Systems*, page 1703. Springer US.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the International Conference on North American Chapter of the Association for Computational Linguistics*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Jinlan Fu, Yi Li, Qi Zhang, Qinzhuo Wu, Renfeng Ma, Xuanjing Huang, and Yu-Gang Jiang. 2020. Recurrent memory reasoning network for expert finding in community question answering. In *Proceedings of the International Conference on Web Search and Data Mining*, pages 187–195, New York, NY, USA. Association for Computing Machinery.

Negin Ghasemi, Ramin Fatourechi, and Saeedeh Momtazi. 2021. User embedding for expert finding in community question answering. *Proceedings of the ACM Transactions on Knowledge Discovery from Data*, 15(4):1–16.

Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using click through data. In *Proceedings of the Conference on Information and Knowledge Management*, CIKM '13, page 2333–2338, New York, NY, USA. Association for Computing Machinery.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations*, pages 1–15. Ithaca, NY: arXiv.org.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Zeyu Li, Jyun-Yu Jiang, Yizhou Sun, and Wei Wang. 2019. Personalized question routing via heterogeneous network embedding. In *Proceedings of the International Conference on Artificial Intelligence*, volume 33, pages 192–199. AAAI Press.

Hongtao Liu, Zhepeng Lv, Qing Yang, Dongliang Xu, and Qiyao Peng. Efficient non-sampling expert finding. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management, Atlanta, GA, USA, October 17-21, 2022*, pages 4239–4243. ACM.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the Internation Conference on Neural Information Processing Systems*, pages 3111–3119, Red Hook, NY, USA. Curran Associates Inc.

Qiyao Peng, Hongtao Liu, Yinghui Wang, Hongyan Xu, Pengfei Jiao, Minglai Shao, and Wenjun Wang. 2022a. Towards a multi-view attentive matching for personalized expert finding. In *Proceedings of the ACM Web Conference 2022*, pages 2131–2140.

Qiyao Peng, Wenjun Wang, Hongtao Liu, Yinghui Wang, Hongyan Xu, and Minglai Shao. 2022b. Towards comprehensive expert finding with a hierarchical matching network. *Knowledge-Based Systems*, 257:109933.

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

Xipeng Qiu and Xuanjing Huang. 2015. Convolutional neural tensor network architecture for community-based question answering. In *Proceedings of the International Joint Conference on Artificial Intelligence*. AAAI Press.

Zhaopeng Qiu, Xian Wu, Jingyue Gao, and Wei Fan. 2021. U-bert: Pre-training user representations for improved recommendation. In *Proc. of the AAAI Conference on Artificial Intelligence. Menlo Park, CA, AAAI*, pages 1–8.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of Machine Learning Research*, 15(1):1929–1958.

Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 1441–1450.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the International Conference of Neural Information Processing Systems*, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Chuhan Wu, Fangzhao Wu, Tao Qi, and Yongfeng Huang. 2021. Empowering news recommendation with pre-trained language models. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1652–1656.

Chuhan Wu, Fangzhao Wu, Tao Qi, Jianxun Lian, Yongfeng Huang, and Xing Xie. 2020. Ptum: Pre-training user model from unlabeled user behaviors via self-supervision. *arXiv preprint arXiv:2010.01494*.

Liu Yang, Minghui Qiu, Swapna Gottipati, Feida Zhu, Jing Jiang, Huiping Sun, and Zhong Chen. 2013. Cqarank: jointly model topics and expertise in community question answering. In *Proceedings of the ACM International Conference on Information & Knowledge Management*, pages 99–108, New York, NY, USA. Association for Computing Machinery.

Sha Yuan, Yu Zhang, Jie Tang, Wendy Hall, and Juan Bautista Cabotà. 2020. Expert finding in community question answering: a review. *Artificial Intelligence Review*, 53(2):843–874.

Xuchao Zhang, Wei Cheng, Bo Zong, Yuncong Chen, Jianwu Xu, Ding Li, and Haifeng Chen. 2020. Temporal context-aware representation learning for question routing. In *Proceedings of the International Conference on Web Search and Data Mining*, pages 753–761, New York, NY, USA. Association for Computing Machinery.

Zhou Zhao, Hanqing Lu, Vincent Zheng, Deng Cai, Xiaofei He, and Yueting Zhuang. 2017. Community-based question answering via asymmetric multi-faceted ranking network learning. In *Proceedings of the International Conference on Artificial Intelligence*.

Zhou Zhao, Lijun Zhang, Xiaofei He, and Wilfred Ng. 2014. Expert finding for question answering via graph regularized matrix completion. *IEEE Transactions on Knowledge and Data Engineering*, 27(4):993–1004.