# Dual-View Self-Supervised Pre-Training for Expert Finding

**Mingqiao Zhang**[1] , **Hongtao Liu**[2] , **Yinghui Wang**[3,4] , **Yumeng Wang**[2] and **Qiyao Peng**[*2]
[1]School of Computer Science and Technology, Nanjing University, Nanjing, China
[2]Tianjin University, Tianjin, China
[3]Beijing Institute of Control and Electronic Technology, Beijing, China
[4]National Key Laboratory of Information Systems Engineering, Beijing, China
231220059@smail.nju.edu.cn, {htliu, wangyinghui, ymwang, qypeng}@tju.edu.cn

## Abstract

Expert finding plays a crucial role in Community Question Answering platforms by routing questions to the most suitable answerers. The key challenge lies in learning high-quality representations for questions and experts. Most existing methods rely on limited supervised signals such as expert–question interactions, and typically focus on modeling only one side of the expert–question pair, suffering from data sparsity and incomplete representation. In this paper, we propose a Dual-view Self-Supervised pre-training framework for Expert Finding (SSEF) that simultaneously pre-trains expert and question representations from large-scale unlabeled data. Specifically, on the expert view, we design a self-supervised module with two data-augmentation strategies, namely historical behavior cropping and reordering, and optimize expert representations via contrastive learning over augmented sequences of historically answered questions. On the question view, we apply analogous augmentation strategies to capture intrinsic semantic differences among questions. The two view-specific modules are unified through multi-task learning with shared PLM parameters, enabling the model to capture latent semantic relatedness across views. Extensive experiments on six real-world CQA datasets demonstrate that SSEF consistently outperforms existing methods, and further analysis confirms its effectiveness under zero-shot settings and its transferability to other models.



## 1 Introduction

Expert finding methods route questions to suitable answerers on Community Question Answering (CQA) websites, and have attracted considerable attention in recent years [Hu *et al.*, 2019; Hu *et al.*, 2021; Li *et al.*, 2022; Zhao *et al.*, 2017; Qian *et al.*, 2022; Qian *et al.*, 2023]. The task requires learning well representations of both experts and questions from historical interactions, so that an incoming question can be matched to a capable expert.

Existing methods vary in how they balance the modeling of experts and questions. A first line prioritizes question modeling. Doc2Vec-based and early neural methods [**?**; Shahriari *et al.*, 2015] produce rich question representations, yet represent experts through simple aggregation (e.g., mean or max pooling) over their answered questions, which collapses diverse answering preferences into coarse vectors. A second line of work prioritizes expert modeling. ExpertPLM [Peng and Liu, 2022] and ExpertBert [Liu *et al.*, 2022b] design pre-training tasks tailored to expert behaviors and achieve fine-grained expert representations, but treat questions as information inputs and encode them with relatively shallow strategies, leaving question semantics underexploited. A third line attempts to give equal attention to both sides. NeRank [Li *et al.*, 2019], RMRN [Fu *et al.*, 2020] and PMEF [Peng *et al.*, 2022] learn expert and question representations jointly and obtain strong performance, yet they rely on supervised expert–question interactions for representation learning, and such interactions are extremely sparse on CQA platforms. For example, the interaction densities on the *Es* and *Gis* datasets from StackExchange[1] are only 0.0429% and 0.0435%, which fundamentally limits the quality of learned representations regardless of modeling strategy. In short, existing work suffers from two intertwined limitations: an imbalanced modeling granularity across the expert and question views, and a universal dependence on sparse supervised signals.

Self-supervised pre-training remarkably successful at distilling general representations from unlabeled data in NLP [Devlin *et al.*, 2018; Peng *et al.*, 2025] points to a natural direction. However, adapting it to expert finding is non-trivial. Standard PLMs operate at the sentence and token level, whereas an expert is not a sentence but a sequence of historical answering behaviors, and token-level objectives such as Masked Language Modeling cannot distinguish the inherent preference differences between experts. Besides, expert representations are learned only indirectly, as a by-product of fitting expert-question matching labels. There is no explicit mechanism that pushes the model to distinguish one expert's personalized preference from another's, so experts with overlapping answer vocabularies but different underlying interests tend to collapse into similar representa-

*Corresponding author

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

tions. This leaves two specific questions: (i) How to design a dual-view pre-training architecture that handles expert and question representations simultaneously? and (ii) How to design view-specific self-supervised tasks that capture the unique characteristics of each view?

To this end, we propose a Dual-view **S**elf-**S**upervised pre-training framework for **E**xpert **F**inding (**SSEF**). The core is that we design two view-specific pre-training tasks for expert and question representation learning respectively from large unlabeled data to address the two aforementioned questions.

For question-i, we construct a dual-view pre-training corpus including: (1) *question view*, which contains the title of each question (one question, one input line) from CQA websites; (2) *expert view*, which aggregates the historical questions answered by one expert as one sequence (one expert, one input line). We further design two special tokens (`[ER]` and `[QR]`) at the beginning of two sequences to help the PLMs identify whether the input is an expert or a question. In this way, the model is endowed with the ability to capture expert and question representations simultaneously. For question-ii, one of the crucial modules in self-supervised pre-training is data augmentation. Hence, considering that the expert interest could be learned from the historical behaviors (i.e., the answered questions), we propose two expert-view data augmentation methods (namely behavior crop and reorder) to augment the original *expert-view information*. Then the model can be optimized with expert-view contrastive learning. In this way, our method can pre-train expert representations by distinguishing the preference difference between experts. For questions, we employ two data augmentation strategies on the *question-view information* (namely word crop and reorder) and pre-train question features by distinguishing the intrinsic difference between questions.

The expert-/question-view self-supervised modules are unified by multi-task learning and share the parameters. In the fine-tuning phase, we utilize the pre-trained weight to encode expert/question from their respective views, and then accomplish the downstream expert finding task. In summary, the contributions of our method are:

(1) We design a self-supervised pre-training framework for expert finding, and this pre-training and fine-tuning paradigm could alleviate data sparsity.

(2) The model pre-trains experts and questions simultaneously. Besides We design new data-augmentation approaches for expert-view and question-view self-supervised learning.

(3) To verify the effectiveness of SSEF, we perform extensive experiments on six real-world CQA datasets. Experimental results demonstrate that our method can achieve better performance than existing baselines and validate the effectiveness of our approach.

(4) In zero-shot setting, our method still shows competitive results. Another observation is that our pre-training weight could further improve other methods' performance.

## 2 Related Works

### 2.1 Expert Finding

To find capable experts for providing satisfactory answers to questions on CQA websites, the expert finding has received much attention from both research and industry community [Zhu *et al.*, 2011; Liu *et al.*, 2015; Roy *et al.*, 2018; Yuan *et al.*, 2020b; Wang *et al.*, 2024; Amendola *et al.*, 2024; Etemadi *et al.*, 2024]. Previous studies about expert finding could be categorized into *traditional methods* and *deep learning-based methods*. In most traditional methods, feature-engineering [Cao *et al.*, 2012] or topic-modeling technology [Guo *et al.*, 2008; Riahi *et al.*, 2012; Wei *et al.*, 2016] is employed to model the questions and experts, then routed target questions to the suitable experts. Deep learning-based methods employed the neural network to model experts and questions, then matched experts and target questions [Li *et al.*, 2019; Fu *et al.*, 2020; Ghasemi *et al.*, 2021; Liu *et al.*, 2022b]. For example, TCQR [Zhang *et al.*, 2020] employed a temporal context-aware attention model to learn the answerer representation, which could model temporal dynamics by multi-resolution settings.

Different from existing methods of learning expert and question representations that usually rely on supervised data, we design a self-supervised pre-training paradigm, which could integrate self-supervised signals from large unlabeled data and pre-train expert and question representations.

### 2.2 Self-Supervised Learning

Self-Supervised Learning (SSL) [Liu *et al.*, 2020; Gui *et al.*, 2024; Ren *et al.*, 2025] have shown outstanding performance on various machine learning tasks, especially in Computer Vision [He *et al.*, 2020; Newell and Deng, 2020], Natural Language Processing [Devlin *et al.*, 2018], etc. For example, BERT [Devlin *et al.*, 2018] used masked language models to capture deep bidirectional representations, which has been utilized in various domains [Xu *et al.*, 2023]. However, 1) they almost are designed on corpus-level or sentence-level for learning general language knowledge; 2) data augmentation tasks in these methods (e.g., token shuffle/masking) are almost token-level, which would be not suitable for modeling expert. Hence, designing appropriate data augmentation tasks for expert finding is necessary.

It is noted that ExpertPLM [Peng and Liu, 2022] and ExpertBert [Liu *et al.*, 2022b] are also expert finding models with self-supervised paradigms. However, they 1) only focus on expert modeling and omit the question modeling simultaneously; 2) usually mask words in the expert-view input information for expert interest modeling, which is not customized for expert finding and could not distinguish the inherent interest difference between experts.

## 3 Problem Statement

In this section, we formalize the expert finding. We represent column vectors and matrices by bold lower case letters (e.g., $\mathbf{e}$) and bold upper case letters (e.g., $\mathbf{X}$), respectively. Then, we formulate the problem definition of expert finding in our paper. Suppose that there is a target question $q^t$ and a candidate expert set $\mathcal{C} = \{c_1, \cdots, c_N\}$ respectively, where $N$ is the number of experts. We need to compute the matching score $s_c$ measuring the relevance between the expert in $\mathcal{C}$ and the target question $q^t$. Then, candidate experts are ranked and recommended to the target question $q^t$ based on the $s_c$.

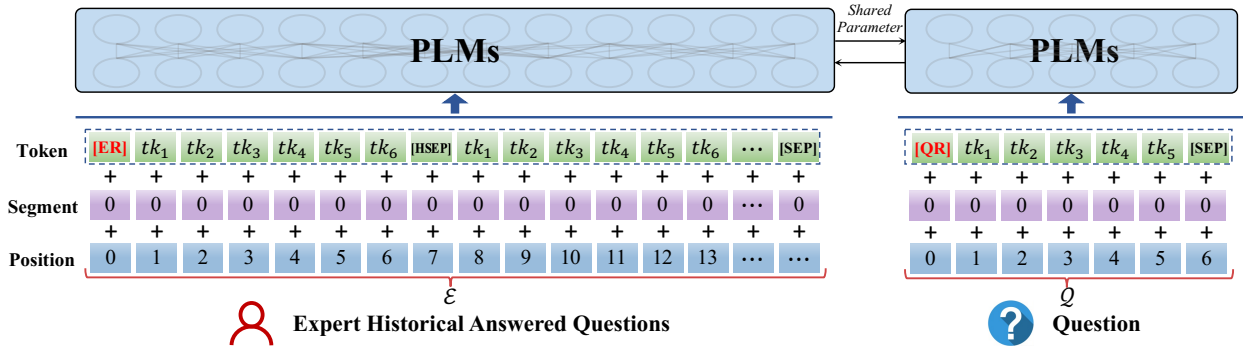


Figure 1: SSEF Pre-training Framework. We feed the expert-view information (e.g., $q_1$, $q_2$, $\cdots$) and the question-view information (e.g., $q$) into BERT for pre-training expert/question.

Furthermore, the candidate expert $c_i \in \mathcal{C}$ is associated with a set of her/his historical answered questions, which can be denoted as $\mathcal{E}_i = \{q_1, \cdots, q_m\}$, where $m$ is the number of historical questions. Any question is represented by its related title, which is denoted as a word sequence $\mathcal{Q}$. Note that the expert who provides the "accepted answer" to the question will be regarded as the ground truth.

## 4 Proposed Method

In this section, we will introduce the training procedure of our dual-view method *SSEF*, which is composed of the following stages:

(1) In pre-training, the expert-/question-view pre-training inputs are first well constructed respectively. Then we propose expert-/question-view data augmentation methods to enhance original expert/question information and perform self-supervised tasks to fully model expert/question from unlabeled input data respectively. (2) In fine-tuning, we initialize the parameters of the expert or question representation model with the pre-trained parameters and then utilize traditional expert finding supervised signals to fine-tune the model.

### 4.1 Dual-View Pre-Training Framework

In this part, we will present the expert/question pre-training framework, including the well-designed input layer and a brief backbone architecture introduction.

For expert-view input, we concatenate the words of the expert's historically answered questions into a whole sequence as shown in Figure 1. We add `[HSEP]` tokens between histories to distinguish different historical questions, and the special token `[SEP]` at the end of the input word sequence for recognizing the end of the sequence. Moreover, we propose to design a special token (`[ER]`) at the beginning of the input sequence, for indicating the expert information and help obtaining representations. Hence, the expert-view information is denoted as:

$$\mathcal{E} = [[\texttt{ER}], \underbrace{[tk_1], [tk_2]}_{q_1}, [\texttt{HSEP}], \underbrace{[tk_1], [tk_2], [tk_3]}_{q_2}, [\texttt{SEP}]]. \quad (1)$$

Likewise, we propose a special token `[QR]` at the beginning of the question-view information to facilitate obtaining the question representation, which is defined as follows:

$$\mathcal{Q} = [[\texttt{QR}], \underbrace{[tk_1], [tk_2], [tk_3], [tk_4], [tk_5]}_{q}, [\texttt{SEP}]]. \quad (2)$$

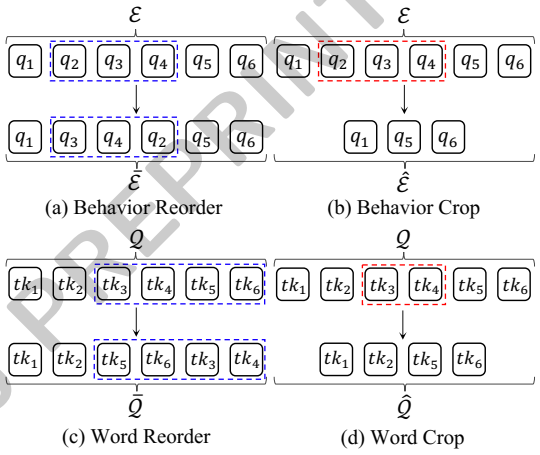


Figure 2: Data Augmentation Strategies.

The special tokens `[ER]` and `[QR]` are not just for capturing the representation of experts and questions. More importantly, they could help the model identify whether the input data is an expert or a question, which can be employed throughout the pre-training and fine-tuning.

Then, the input representation matrix $\mathbf{X}$ is constructed by summing its corresponding token, segment, and position embedding:

$$\mathbf{X} = \mathbf{X}_{token} + \mathbf{X}_{seg} + \mathbf{X}_{pos}\ . \quad (3)$$

It is noted that we don't employ the Next Sentence Prediction task during pre-training, and hence, we set all the segment IDs to 0.

In our method, we adopt the widely used BERT [Devlin *et al.*, 2018] as our base backbone for further pre-training.

### 4.2 Data Augmentation

In this part, we will show the well-designed data augmentation methods for experts and questions before pre-training.

*4.2.1 Expert-view Augmentation* For generating self-supervised signals to pre-train expert representation from historical questions, we design two augmentation approaches (shown in Figure 2 (a)(b)), which could construct different views of the same sequence but still maintain the main preference hidden in historical interactions.

**(1) Behavior Reorder** In fact, we could obtain satisfied recommendation lists without strictly following the chronological order [Yuan *et al.*, 2019; Yuan *et al.*, 2020a]. Similarly, on CQA websites, it is likely that the expert answer questions in any order. For example, the sequence of historical questions is [$q_1-$"Single word for plaintiff or defendant", $q_2-$"Geometric or Geometrical?", $q_3-$"Replacement for this means that ..."]. This is almost equivalent to [$q_2, q_1, q_3$], since there is no strict order between questions.

Hence, we can derive a self-supervised augmentation operator with the assumption of flexible order. Specifically, given an expert historical question sequence $\mathcal{E}_i = \{q_1, \cdots, q_m\}$, we randomly shuffle a continuous sub-sequence with augmentation ratio $\alpha_b$ (the length is $\alpha_b * |\mathcal{E}_i|$), denoted as:

$$\bar{\mathcal{E}}_i = \texttt{reorder}(\mathcal{E}_i). \tag{4}$$

In this way, the SSEF could model the expert by relying less on the order of interaction sequences.

**(2) Behavior Crop** In fact, randomly cropping a continuous sub-sequence in $\mathcal{E}$ might not affect the expert modeling greatly. This augmentation is designed to identify the influential historical questions for modeling expert, and make that less sensitive to the complete historical sequence. More concretely, given an expert historical question sequence $\mathcal{E}_i = \{q_1, \cdots, q_m\}$, we randomly crop a continuous sub-sequence (with the length $\beta_b * |\mathcal{E}_i|$), which is denoted as:

$$\hat{\mathcal{E}}_i = \texttt{crop}(\mathcal{E}_i). \tag{5}$$

In this way, the SSEF could capture a local view of the expert histories and model more generalized expert representation without comprehensive information of experts.

*4.2.2 Question-view Augmentation.* We employ two augmentation approaches on the question-view information to construct different views of that. As shown in Figure 2 (c)(d), given the $i$-th question word sequence $\mathcal{Q}_i$, we employ **word reorder** [Yan *et al.*, 2021] or **word crop** augmentation strategies to change a continuous sub-sequence in one question, and the augmentation ratio is $\alpha_w$ or $\beta_w$ (e.g., the changing length is $\alpha_w * |\mathcal{Q}_i|$). The operators are defined as follows:

$$\bar{\mathcal{Q}}_i = \texttt{reorder}(\mathcal{Q}_i)\ , \hat{\mathcal{Q}}_i = \texttt{crop}(\mathcal{Q}_i)\ . \tag{6}$$

It is noted that we only shuffle position IDs while keeping the order of the token IDs in the word reorder. This procedure encourages the model to identify the unique information that is shared across different views of the same question input, which helps learn more generalized question embeddings.

## 4.3 Self-Supervised Pre-Training Task

Having established the augmented views of the expert or question, we design two self-supervised tasks for expert and question respectively during the pre-training stage.

**Expert Contrastive Task.** This task teaches the model to discriminate the representation of a particular expert from other experts. Given $j$-th expert sequence, we have obtained the expert representation $\mathbf{e}_j$ (generated from `[ER]`). The augmented ones (randomly selected from behavior crop or reorder) are considered as positives (with the representation as $\bar{\mathbf{e}}_j$ or $\hat{\mathbf{e}}_j$), while other in-batch (with $B$ is batch size) experts are considered as negatives. The expert-view self-supervised loss can be formally presented as follows:

$$\mathcal{L}_{ec} = -\sum_{j=1}^{B} \log \frac{\exp(\text{sim}(\mathbf{e}_j \cdot \hat{\mathbf{e}}_j)/\tau)}{\sum_{j'=1}^{B} \exp(\text{sim}(\mathbf{e}_j \cdot \hat{\mathbf{e}}_{j'})/\tau)}\ , \tag{7}$$

$$\text{sim}(\mathbf{a} \cdot \mathbf{b}) = \cos(\mathbf{a} \cdot \mathbf{b}) = \frac{\mathbf{a}^T \mathbf{b}}{\|\mathbf{a}\|_2 \cdot \|\mathbf{b}\|_2}\ , \tag{8}$$

where $\tau$ is a temperature parameter. Since batches are constructed randomly, the in-batch negative augmented instances $\hat{\mathbf{e}}_{j'}$ would contain some experts from multiple domains.

**Question Contrastive Task.** The object is to discriminate the representation of a particular question from other questions, which is similar to the expert contrastive task. Given $i$-th question, the question-view self-supervised loss can be denoted as:

$$\mathcal{L}_{qc} = -\sum_{i=1}^{B} \log \frac{\exp(\text{sim}(\mathbf{q}_i \cdot \hat{\mathbf{q}}_i)/\tau)}{\sum_{i'=1}^{B} \exp(\text{sim}(\mathbf{q}_i \cdot \hat{\mathbf{q}}_{i'})/\tau)}\ , \tag{9}$$

where $\hat{\mathbf{q}}_i$ is $i$-th question augmented representation and $\hat{\mathbf{q}}_{i'}$ is the negative augmented instance.

Furthermore, for helping the model capture the CQA language knowledge, inspired by BERT [Devlin *et al.*, 2018], we simply mask 15% of the input tokens randomly (default in the BERT) and then predict those masked tokens. The formal definition is:

$$\mathcal{L}_{wm} = -\sum_{i \in \mathcal{N}} \log p(n = n_i|\theta), \quad n_i \in [1, 2, \cdots, |\mathcal{V}|] \tag{10}$$

where $\mathcal{N}$ is the masked token set, $\mathcal{V}$ is the vocabulary.

**Model Learning.** At the pre-training stage, we leverage a multi-task training strategy to jointly pre-train the model:

$$\mathcal{L}_{pt} = \mathcal{L}_{ec} + \mathcal{L}_{qc} + \mathcal{L}_{wm}\ . \tag{11}$$

## 4.4 Fine-Tuning for Expert Finding

We initialize the expert and question encoders with pre-trained parameters. Expert sequences begin with an `[ER]` token and question sequences with a `[QR]` token. Their representations are concatenated to predict the matching score $s_c$. Using negative sampling [Huang *et al.*, 2013] with $K$ negative instances, we fine-tune with cross-entropy loss:

$$s_c = \frac{\exp(s_c)}{\sum_{j=1}^{K+1} \exp(s_j)}\ , \mathcal{L}_{ft} = -\sum_{c=1}^{K+1} \hat{s_c} \log(s_c)\ , \tag{12}$$

where $\hat{s_c}$ is the ground truth label and $s_c$ is the normalized predicted probability.

| Dataset | CodeGolf | | | | 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 |
| NeRank | 0.4496±0.0014 | 0.2616±0.0073 | 0.5143±0.0010 | 0.5341±0.0045 | 0.4697±0.0015 | 0.3032±0.0014 | 0.5577±0.0018 | 0.5836±0.0080 | 0.4947±0.0016 | 0.3089±0.0037 | 0.6055±0.0041 | 0.6154±0.0012 |
| TCQR | 0.3225±0.0073 | 0.2027±0.0032 | 0.5087±0.0014 | 0.5322±0.0013 | 0.4553±0.0094 | 0.2634±0.0013 | 0.5489±0.0015 | 0.5637±0.0071 | 0.4253±0.0159 | 0.2112±0.0017 | 0.5382±0.0017 | 0.5839±0.0047 |
| RMRN | 0.4526±0.0025 | 0.2321±0.0010 | 0.5052±0.0020 | 0.5375±0.0021 | 0.4897±0.0016 | 0.3239±0.0026 | 0.5777±0.0035 | 0.5832±0.0031 | 0.4311±0.0019 | 0.2517±0.0027 | 0.5580±0.0011 | 0.5892±0.0038 |
| UserEmb | 0.3073±0.0161 | 0.1857±0.0138 | 0.4132±0.0165 | 0.4530±0.0134 | 0.3223±0.0148 | 0.2433±0.0075 | 0.4477±0.0095 | 0.4562±0.0102 | 0.3915±0.0079 | 0.2031±0.0096 | 0.5126±0.0081 | 0.5246±0.0101 |
| PMEF | 0.4232±0.0014 | 0.2617±0.0021 | 0.5066±0.0019 | 0.5343±0.0037 | 0.4861±0.0045 | 0.3311±0.0017 | 0.5888±0.0024 | 0.5911±0.0027 | 0.5020±0.0031 | 0.3139±0.0051 | 0.6161±0.0038 | 0.6170±0.0012 |
| ENEF | 0.3159±0.0049 | 0.1927±0.0040 | 0.3073±0.0038 | 0.4326±0.0041 | 0.4936±0.0045 | 0.3337±0.0043 | 0.5732±0.0030 | 0.6023±0.0025 | 0.4906±0.0032 | 0.3103±0.0031 | 0.6012±0.0030 | 0.6120±0.0048 |
| ExpertBert | 0.4431±0.0032 | 0.3032±0.0024 | 0.5195±0.0029 | 0.5816±0.0025 | 0.5170±0.0032 | 0.3537±0.0026 | 0.6066±0.0024 | 0.6265±0.0013 | 0.5123±0.0028 | 0.3287±0.0024 | 0.6265±0.0028 | 0.6248±0.0032 |
| ExpertPLM | 0.4533±0.0025 | 0.3013±0.0023 | 0.5196±0.0033 | 0.5815±0.0027 | 0.5085±0.0021 | 0.3456±0.0021 | 0.6058±0.0022 | 0.6178±0.0028 | 0.5115±0.0031 | 0.3263±0.0026 | 0.6313±0.0021 | 0.6242±0.0025 |
| CGEF | 0.4621±0.0033 | 0.3147±0.0027 | 0.5318±0.0033 | 0.5864±0.0032 | 0.5098±0.0020 | 0.3444±0.0030 | 0.5563±0.0029 | 0.6074±0.0027 | 0.4981±0.0024 | 0.3215±0.0028 | 0.5862±0.0022 | 0.6114±0.0026 |
| **SSEF** | **0.4724±0.0014** | **0.3235±0.0018** | **0.5423±0.0025** | **0.5912±0.0031** | **0.5225±0.0020** | **0.3674±0.0027** | **0.6118±0.0034** | **0.6275±0.0015** | **0.5225±0.0012** | **0.3390±0.0026** | **0.6334±0.0030** | **0.6391±0.0016** |

| Dataset | 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 |
| NeRank | 0.5647±0.0049 | 0.3932±0.0014 | 0.6413±0.0012 | 0.6617±0.0028 | 0.4371±0.0024 | 0.2886±0.0023 | 0.5361±0.0017 | 0.4991±0.0015 | 0.5105±0.0044 | 0.3459±0.0015 | 0.5976±0.0047 | 0.6221±0.0037 |
| TCQR | 0.4716±0.0022 | 0.3059±0.0019 | 0.5941±0.0015 | 0.6053±0.0023 | 0.3962±0.0024 | 0.2406±0.0042 | 0.4422±0.0048 | 0.4716±0.0011 | 0.5016±0.0035 | 0.3233±0.0020 | 0.5953±0.0034 | 0.6004±0.0036 |
| RMRN | 0.5516±0.0024 | 0.3761±0.0038 | 0.6304±0.0047 | 0.6453±0.0027 | 0.4662±0.0010 | 0.3153±0.0020 | 0.5562±0.0037 | 0.5578±0.0031 | 0.5622±0.0027 | 0.4038±0.0019 | 0.6759±0.0023 | 0.6729±0.0031 |
| UserEmb | 0.4703±0.0016 | 0.3032±0.0015 | 0.5735±0.0034 | 0.4869±0.0016 | 0.3497±0.0032 | 0.2263±0.0032 | 0.3675±0.0048 | 0.4672±0.0026 | 0.4175±0.0012 | 0.2954±0.0039 | 0.4541±0.0044 | 0.5170±0.0017 |
| PMEF | 0.5735±0.0034 | 0.4269±0.0015 | 0.6520±0.0045 | 0.6661±0.0028 | 0.4719±0.0050 | 0.3216±0.0049 | 0.5693±0.0033 | 0.5688±0.0046 | 0.5872±0.0022 | 0.4238±0.0014 | 0.6950±0.0012 | 0.6829±0.0024 |
| ENEF | 0.5302±0.0028 | 0.3732±0.0015 | 0.6479±0.0024 | 0.6338±0.0016 | 0.4641±0.0034 | 0.3138±0.0027 | 0.5341±0.0039 | 0.5593±0.0050 | 0.5523±0.0033 | 0.4196±0.0020 | 0.6823±0.0033 | 0.6911±0.0032 |
| ExpertBert | 0.5799±0.0024 | 0.4278±0.0022 | 0.6537±0.0034 | 0.6581±0.0038 | 0.4970±0.0050 | 0.3323±0.0020 | 0.5766±0.0024 | 0.5865±0.0045 | 0.5923±0.0019 | 0.4257±0.0039 | 0.6968±0.0034 | 0.6933±0.0027 |
| ExpertPLM | 0.5840±0.0050 | 0.4380±0.0044 | 0.6588±0.0025 | 0.6793±0.0030 | 0.4956±0.0023 | 0.3353±0.0014 | 0.5868±0.0012 | 0.5929±0.0030 | 0.6003±0.0025 | 0.4356±0.0027 | 0.7025±0.0014 | 0.7046±0.0033 |
| CGEF | 0.5872±0.0026 | 0.4408±0.0010 | 0.6625±0.0023 | 0.6841±0.0041 | 0.4901±0.0035 | 0.3346±0.0039 | 0.5654±0.0039 | **0.6176±0.0046** | 0.6068±0.0028 | 0.4435±0.0015 | 0.7071±0.0045 | 0.7080±0.0044 |
| **SSEF** | **0.5930±0.0016** | **0.4433±0.0017** | **0.6678±0.0023** | **0.6894±0.0024** | **0.5056±0.0025** | **0.3466±0.0032** | **0.5918±0.0049** | 0.6101±0.0018 | **0.6123±0.0024** | **0.4526±0.0041** | **0.7123±0.0014** | **0.7099±0.0013** |

Table 1: Expert finding results of different methods. The best performance of the baselines is underlined.

| Dataset | Biology | | | | CodeGolf | | | | Electronics | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Method \ Metric | MRR | P@1 | P@3 | NDCG@20 | MRR | P@1 | P@3 | NDCG@20 | MRR | P@1 | P@3 | NDCG@20 |
| Only QR | 0.4837±0.0063 | 0.3332±0.0065 | 0.5708±0.0084 | 0.5874±0.0066 | 0.4465±0.0024 | 0.2961±0.0062 | 0.5239±0.0092 | 0.5633±0.0080 | 0.5992±0.0011 | 0.4296±0.0070 | 0.7008±0.0080 | 0.6934±0.0080 |
| Only ER | 0.4936±0.0075 | 0.3384±0.0060 | 0.5759±0.0086 | 0.6031±0.0065 | 0.4527±0.0065 | 0.3018±0.0064 | 0.5326±0.0079 | 0.5807±0.0047 | 0.6029±0.0032 | 0.4325±0.0066 | 0.7072±0.0069 | 0.6953±0.0037 |
| Original BERT | 0.4738±0.0061 | 0.3285±0.0015 | 0.5704±0.0094 | 0.5779±0.0076 | 0.4391±0.0040 | 0.2883±0.0036 | 0.5127±0.0064 | 0.5538±0.0038 | 0.5861±0.0072 | 0.4207±0.0053 | 0.6932±0.0029 | 0.6815±0.0029 |
| **SSEF** | **0.5056±0.0025** | **0.3466±0.0032** | **0.5918±0.0049** | **0.6101±0.0018** | **0.4724±0.0014** | **0.3235±0.0018** | **0.5423±0.0025** | **0.5912±0.0031** | **0.6123±0.0024** | **0.4526±0.0041** | **0.7123±0.0014** | **0.7099±0.0013** |

Table 2: The variants of SSEF pre-training experiment results.

# 5 Experiments

## 5.1 Experimental Settings

**Datasets and Metrics.** For pre-training, we construct a corpus from StackExchange[2] comprising 103005 expert and 286835 question sequences, with 670224 interactions across 13262 tags. The pre-training corpus strictly excludes all fine-tuning validation and test data. For fine-tuning, we evaluate on six domains: *Biology*, *CodeGolf*, *Gis*, *Es*, *Electronics*, and *CodeReview*. Following [Peng *et al.*, 2022], the provider of the "accepted answer" serves as the ground truth expert, while platforms lacking explicit accepted answers are excluded [Li *et al.*, 2019]. As summarized in Table 3, we chronologically split each dataset into 80/10/10% for training, validation, and testing. Performance is evaluated using Mean Reciprocal Rank (MRR), Precision@$K$ ($K = 1, 3$), and NDCG@20.

**Baselines.** We compare SSEF with eight competitive baselines: (1) *Traditional Methods*: NeRank [Li *et al.*, 2019], UserEmb [Ghasemi *et al.*, 2021], ENEF [Liu *et al.*, 2022a] and CGEF [Peng *et al.*, 2024]; (2) *PLM-based Methods*: TCQR [Zhang *et al.*, 2020], RMRN [Fu *et al.*, 2020], PMEF [Peng *et al.*, 2022], ExpertBert [Liu *et al.*, 2022b], and ExpertPLM [Peng and Liu, 2022]. Furthermore, we test the model transferable performance on PMEF [Peng *et al.*, 2022] and MATER [Zahedi *et al.*, 2024] in Section 5.5.

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

| Datasets | # questions | # answerers | # interactions | density(%) |
|---|---|---|---|---|
| Es | 36,271 | 3,260 | 50,801 | 0.0429 |
| Gis | 50,718 | 3,168 | 70,034 | 0.0435 |
| Biology | 8,704 | 630 | 11,411 | 0.2081 |
| CodeGolf | 5,101 | 2,874 | 62,456 | 0.4260 |
| Electronics | 56,614 | 3,084 | 102,214 | 0.0585 |
| CodeReview | 36,947 | 2,242 | 57,622 | 0.0695 |

Table 3: Statistical details of each dataset in fine-tuning.

## 5.2 Performance Comparison

We compare SSEF with baseline methods on six CQA datasets. The results are reported in Table 1, and we can find:

Some earlier methods (e.g., Doc2Vec, CNTN) obtain poorer 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. NeRank utilizes the heterogeneous network embedding to unify raisers, answerers, and questions to learn embedding, which has obtained superior performance. Furthermore, the recent method PMEF has obtained superior performance compared with other baselines. One possible reason is that PMEF models the expert and question under a multi-granularity paradigm, which could be more comprehensive.

Compared with most baselines, SSEF performs mostly better than them on six datasets, which could verify the effective-

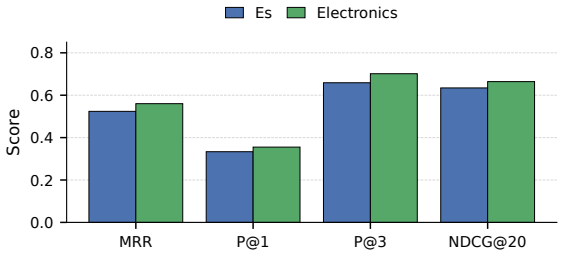


Figure 3: Zero-shot Performance. Fine-tuning on Biology and testing on Es and Electronics.

ness of our proposed method. Different from those methods, we carefully construct expert-/question-view input information and corresponding training tasks, which could model expert/question generally in a self-supervised way and alleviate the supervised signal sparsity issue. It is noted that the very recent works ExpertBert and ExpertPLM also achieve superior performance. Our method outperforms them. The reason may be that they only focus on expert modeling and omit the fact that supervised data is sparse in expert finding, while SSEF carefully designs data augmentation strategies and training tasks for self-supervised pre-training, which could learn more precise representations of experts/questions.

## 5.3 Zero-Shot Performance

Considering our method is a pre-training method, we test the model performance with zero-shot setting. We only fine-tune the pre-trained model on the smallest dataset Biology, and test the performance on the Es and Electronics. The experimental results are shown in Figure 3. It is surprising that the model can still achieve good performance (especially in P@3) by simply transferring from Biology to Es and Electronics without any fine-tuning. This could be attributed to the model has already captured a large amount of expert and question related knowledge during pre-training. And merely fine-tuning on smaller datasets (e.g., biology) is required to stimulate model understanding of the expert finding task, leading to zero-shot performance on other large datasets.

## 5.4 Ablation Study

### Effect of Pre-Training Tasks

To evaluate the effectiveness of our designed inputs and tasks, we compare SSEF against two ablated variants and a standard baseline: **(a) Only QR:** Employs only question-view information and tasks for pre-training. **(b) Only ER:** Employs only expert-view information and tasks for pre-training. **(c) Original BERT (Baseline):** Directly adopts original BERT weights and tasks for further pre-training on the CQA corpus (one question per line), followed by fine-tuning.

As shown in Table 2, we can have the following observations: 1) Original BERT obtains the worst performance. This is equivalent to removing the question-/expert-view tasks from SSEF during pre-training. In this way, the model could lose the ability to model experts/questions. 2) It is surprising that the simple Only QR still outperforms the most of baselines and Original BERT. This is because Only QR could capture CQA language knowledge and benefit question-view

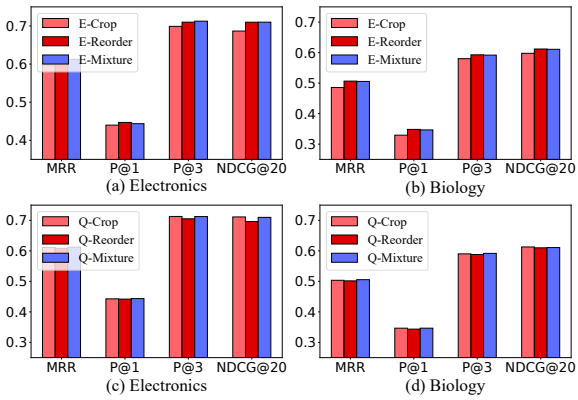


Figure 4: Performance comparison w.r.t various data augmentation strategies on experts or questions.

representation learning during pre-training, which could help downstream expert finding tasks. 3) Only ER outperforms Only QR. This is because Only ER could pre-train expert-view representations that help model experts in downstream tasks. Furthermore, despite the absence of question-view sequences and tasks, Only ER allows the model to simply pre-train question representations due to the question sequences existing in the expert-view sequence. 4) Our complete model SSEF obtains the best results. The reason is that the SSEF could simultaneously pre-train the representations of expert/question with self-supervised tasks, which allows the model to capture unique representations for expert/question before performing the downstream task.

### Effect of Different Data Augmentations

We analyze SSEF with different augmentation strategies. **E-Crop** or **E-Reorder** represents an enhanced view of the expert generated with only crop or reorder operation. **E-Mixture** represents randomly selecting one operation (crop or reorder) for generating the enhanced view. The question side setting is similar. As shown in Figure 4, we observe: 1) All strategies outperform the baseline PMEF, indicating they all assist in pre-training. 2) The composition of strategies might not exceed a single strategy, likely because it is hard for the pre-training model to distinguish which augmentation method transformed the input data. 3) E-Reorder usually performs better than E-Crop, whereas the question side is the opposite. This may be because cropping directly destroys the original expert behavior sequence, providing limited knowledge for downstream tasks. Finally, we employ the mixture strategy to construct different augmented views.

### Effect of Different Augmentation Ratio

We explore different augmentation ratios (expert parameters $\alpha_b/\beta_b$ and question parameters $\alpha_w/\beta_w$ varied in $\{0.1, 0.15, 0.2, 0.25, 0.3\}$) on CodeReview and Biology datasets. As shown in Figure 5, we observe: (1) The performance increases first and then decreases. This is because increasing the ratio improves robustness, while excessive augmentation seriously damages original information. (2) The turning points of the expert and question curves are different. The reason may be that, compared with the average words

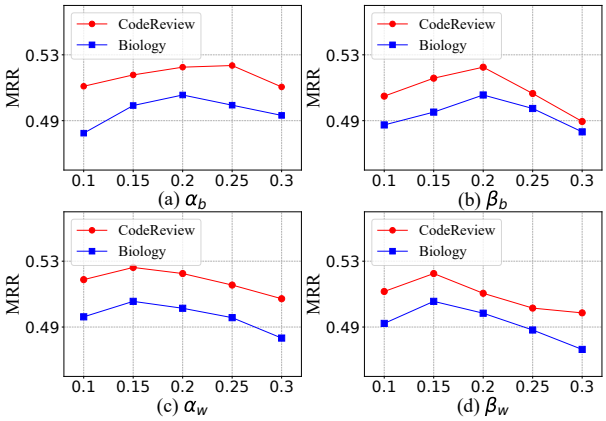


Figure 5: Model Performance with different augmentation ratios.

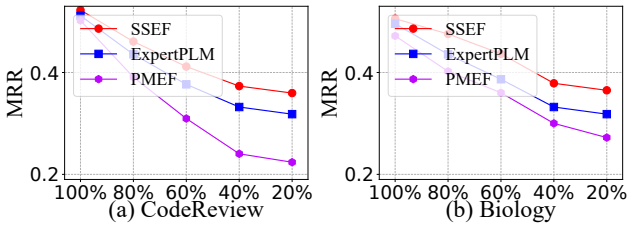


Figure 6: Performance comparison on SSEF, ExpertPLM and PMEF w.r.t. different amounts of training data.

per question, each expert has more behaviors and higher tolerance for data augmentation (e.g., 25.7 behaviors vs. 10.6 words in CodeReview). Thus, we set the question parameters $(\alpha_w, \beta_w)$ to 0.15 and expert parameters $(\alpha_b, \beta_b)$ to 0.2.

## 5.5 Benefit from Pre-training

In this section, we further conduct detailed experiments to analyze the effect of our self-supervised pre-training model.

**Effect of Training Data Ratio in Fine-Tuning**

To investigate whether SSEF alleviates data sparsity, we train on varying proportions ($[100\%, 80\%, 60\%, 40\%, 20\%]$) of the CodeReview and Biology datasets, keeping validation and testing sets fixed at $10\%$. As shown in Figure 6, the performance gap between SSEF and PMEF widens as training data decreases. This suggests that SSEF's pre-training learns general representations, effectively reducing dependency on fine-tuning data. Furthermore, this gap is more pronounced on CodeReview than Biology, aligning with CodeReview's inherently higher sparsity. Overall, results confirm SSEF's potential to mitigate sparse supervised signals.

**Applying the Pre-Training Strategy on Other Models**

Our designed pre-training strategy could learn more general expert and question representations, which have the potential to be applicable for other approaches, e.g., PMEF [Peng *et al.*, 2022], MATER [Zahedi *et al.*, 2024]. We directly add the frozen pre-training expert/question representations to the PMEF and MATER, named PMEF+ and MATER+ respectively for experiments. The results on the public dataset Biology are reported in Figure 7.We can observe that the embeddings derived from our proposed method could consistently improve the performance of PMEF and MATER, which further shows the effectiveness of the proposed method.

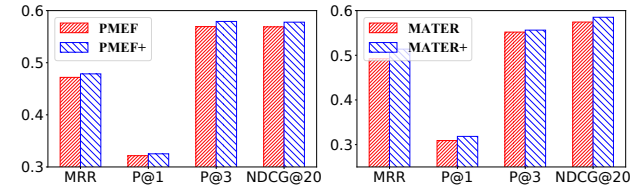


Figure 7: Performance comparison w.r.t. different base models enhanced by our proposed method SSEF.

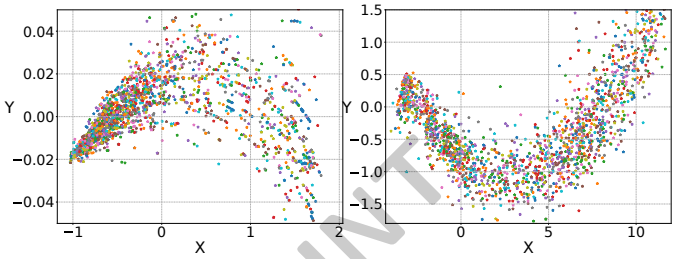


Figure 8: Expert representation 2D visualization on CodeReview dataset: PMEF (left); SSEF (right).

**Understanding the Pre-Trained Representations.**

Recently, uniformity has been recognized as a key property for measuring the quality of representations [Wang and Isola, 2020]. To understand the quality of pre-trained representations, we project the expert representation into 2D space with PCA for visualization. From Figure 8, compared with SSEF, we could observe that most of the expert representations learned by PMEF are mapped in a sharp corner space (the small left part). This is because SSEF pre-trains the expert representation under a self-supervised paradigm. The model could more clearly distinguish experts and prevent them from clustering in the representation space. To some extent, more uniform representations could provide more prior knowledge for the downstream task and promote the performance.

# 6 Conclusion

In this paper, we propose SSEF, a dual-view self-supervised pre-training language model for expert finding. Different from existing supervised-based expert finding methods, the core of our method is to design a dual-view pre-training model for learning transferable expert and question representations in a self-supervised way. In particular, we propose a model-agnostic framework SSEF to supplement the supervised expert finding task. We devise two types of data augmentation on question-view and expert-view information to construct auxiliary question/expert-view contrastive tasks, which could learn more universal expert/question representations for expert finding. Experiments on six real-world CQA datasets validate the effectiveness and transferability of our pre-training paradigm.

## References

[Amendola *et al.*, 2024] Maddalena Amendola, Carlos Castillo, Andrea Passarella, and Raffaele Perego. Understanding and addressing gender bias in expert finding task. *arXiv preprint arXiv:2407.05335*, 2024.

[Cao *et al.*, 2012] Xin Cao, Gao Cong, Bin Cui, Christian S Jensen, and Quan Yuan. Approaches to exploring category information for question retrieval in community question-answer archives. *ACM Transactions on Information Systems (TOIS)*, 30(2):1–38, 2012.

[Devlin *et al.*, 2018] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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*, 2018.

[Etemadi *et al.*, 2024] Roohollah Etemadi, Morteza Zihayat, Kuan Feng, Jason Adelman, Fattane Zarrinkalam, and Ebrahim Bagheri. It takes a team to triumph: Collaborative expert finding in community qa networks. In *Proceedings of the 2024 Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*, pages 164–174, 2024.

[Fu *et al.*, 2020] Jinlan Fu, Yi Li, Qi Zhang, Qinzhuo Wu, Renfeng Ma, Xuanjing Huang, and Yu-Gang Jiang. Recurrent memory reasoning network for expert finding in community question answering. In *WSDM*, pages 187–195, 2020.

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

[Gui *et al.*, 2024] Jie Gui, Tuo Chen, Jing Zhang, Qiong Cao, Zhenan Sun, Hao Luo, and Dacheng Tao. A survey on self-supervised learning: Algorithms, applications, and future trends. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):9052–9071, 2024.

[Guo *et al.*, 2008] Jinwen Guo, Shengliang Xu, Shenghua Bao, and Yong Yu. Tapping on the potential of q&a community by recommending answer providers. In *Proceedings of the ACM Conference on Information and Knowledge Management*, pages 921–930, 2008.

[He *et al.*, 2020] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. In *International Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 9726–9735. Computer Vision Foundation / IEEE, 2020.

[Hu *et al.*, 2019] Jun Hu, Shengsheng Qian, Quan Fang, and Changsheng Xu. Hierarchical graph semantic pooling network for multi-modal community question answer matching. In *Proceedings of the 27th International Conference on Multimedia, 2019, Nice, France, October 21-25, 2019*, pages 1157–1165. ACM, 2019.

[Hu *et al.*, 2021] Jun Hu, Shengsheng Qian, Quan Fang, and Changsheng Xu. Heterogeneous community question answering via social-aware multi-modal co-attention convolutional matching. *IEEE Transactions on Multimedia.*, 23:2321–2334, 2021.

[Huang *et al.*, 2013] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. Learning deep structured semantic models for web search using click through data. In *Proceedings of the Conference on Information and Knowledge Management*, pages 2333–2338, 2013.

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

[Li *et al.*, 2022] Yuhan Li, Wei Shen, Jianbo Gao, and Yadong 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*, pages 2145–2151. ijcai.org, 2022.

[Liu *et al.*, 2015] Xuebo Liu, Shuang Ye, Xin Li, Yonghao Luo, and Yanghui Rao. Zhihurank: A topic-sensitive expert finding algorithm in community question answering websites. In *Proceedings of the International Conference on Web-Based Learning*, pages 165–173. Springer, 2015.

[Liu *et al.*, 2020] Xiao Liu, Fanjin Zhang, Zhenyu Hou, Zhaoyu Wang, Li Mian, Jing Zhang, and Jie Tang. Self-supervised learning: Generative or contrastive. *CoRR*, abs/2006.08218, 2020.

[Liu *et al.*, 2022a] 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*, pages 4239–4243, Atlanta, GA, USA, 2022. ACM.

[Liu *et al.*, 2022b] Hongtao Liu, Zhepeng Lv, Qing Yang, Dongliang Xu, and Qiyao Peng. Expertbert: Pretraining expert finding. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management*, CIKM '22, page 4244–4248, New York, NY, USA, 2022.

[Newell and Deng, 2020] Alejandro Newell and Jia Deng. How useful is self-supervised pretraining for visual tasks? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7345–7354, 2020.

[Peng and Liu, 2022] Qiyao Peng and Hongtao Liu. Expertplm: Pre-training expert representation for expert finding. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 1043–1052, Abu Dhabi, 2022. Association for Computational Linguistics.

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

[Peng *et al.*, 2024] Qiyao Peng, Wenjun Wang, Hongtao Liu, Cuiying Huo, and Minglai Shao. Graph collaborative expert finding with contrastive learning. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pages 2288–2296, 2024.

[Peng *et al.*, 2025] Qiyao Peng, Hongtao Liu, Hua Huang, Jian Yang, Qing Yang, and Minglai Shao. A survey on LLM-powered agents for recommender systems. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, Suzhou, China, November 2025. Association for Computational Linguistics.

[Qian *et al.*, 2022] Lingfei Qian, Jian Wang, Hongfei Lin, Bo Xu, and Liang Yang. Heterogeneous information network embedding based on multiperspective metapath for question routing. *Knowledge-Based Systems*, 240:107842, 2022.

[Qian *et al.*, 2023] Lingfei Qian, Jian Wang, Hongfei Lin, and Liang Yang. Multi-perspective respondent representations for answer ranking in community question answering. *Information Sciences*, 624:37–48, 2023.

[Ren *et al.*, 2025] Xubin Ren, Wei Wei, Lianghao Xia, and Chao Huang. A comprehensive survey on self-supervised learning for recommendation. *ACM Computing Surveys*, 58(1):1–38, 2025.

[Riahi *et al.*, 2012] Fatemeh Riahi, Zainab Zolaktaf, Mahdi Shafiei, and Evangelos Milios. Finding expert users in community question answering. In *Proceedings of the International Conference on World Wide Web*, pages 791–798, 2012.

[Roy *et al.*, 2018] Pradeep Kumar Roy, Jyoti Prakash Singh, and Amitava Nag. Finding active expert users for question routing in community question answering sites. In *Machine Learning and Data Mining in Pattern Recognition: 14th International Conference, MLDM 2018*, pages 440–451, New York, NY, USA, 2018. Springer.

[Shahriari *et al.*, 2015] Mohsen Shahriari, Sathvik Parekodi, and Ralf 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*, pages 1–8, 2015.

[Wang and Isola, 2020] Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *Proceedings of the 37th International Conference on Machine Learning, 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 9929–9939. PMLR, 2020.

[Wang *et al.*, 2024] Yinghui Wang, Qiyao Peng, Hongtao Liu, Hongyan Xu, Minglai Shao, and Wenjun Wang. Deep expertise and interest personalized transformer for expert finding. *Information Processing & Management*, 61(5):103773, 2024.

[Wei *et al.*, 2016] Wei Wei, Gao Cong, Chunyan Miao, Feida Zhu, and Guohui Li. Learning to find topic experts in twitter via different relations. *IEEE Transactions on Knowledge and Data Engineering*, 28(7):1764–1778, 2016.

[Xu *et al.*, 2023] Hongyan Xu, Qiyao Peng, Hongtao Liu, Yueheng Sun, and Wenjun Wang. Group-based personalized news recommendation with long-and short-term fine-grained matching. *ACM Transactions on Information Systems*, 42(1):1–27, 2023.

[Yan *et al.*, 2021] Yuanmeng Yan, Rumei Li, Sirui Wang, Fuzheng Zhang, Wei Wu, and Weiran Xu. ConSERT: A contrastive framework for self-supervised sentence representation transfer. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5065–5075, August 2021.

[Yuan *et al.*, 2019] Fajie Yuan, Alexandros Karatzoglou, Ioannis Arapakis, Joemon M Jose, and Xiangnan He. A simple convolutional generative network for next item recommendation. In *Proceedings of the twelfth ACM international conference on web search and data mining*, pages 582–590, 2019.

[Yuan *et al.*, 2020a] Fajie Yuan, Xiangnan He, Alexandros Karatzoglou, and Liguang Zhang. Parameter-efficient transfer from sequential behaviors for user modeling and recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 1469–1478, 2020.

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

[Zahedi *et al.*, 2024] Mohammad Sadegh Zahedi, Maseud Rahgozar, and Reza Aghaeizadeh Zoroofi. Mater: Bi-level matching-aggregation model for time-aware expert recommendation. *Expert Systems with Applications*, 237:121576, 2024.

[Zhang *et al.*, 2020] Xuchao Zhang, Wei Cheng, Bo Zong, Yuncong Chen, Jianwu Xu, Ding Li, and Haifeng Chen. Temporal context-aware representation learning for question routing. In *WSDM*, pages 753–761, 2020.

[Zhao *et al.*, 2017] Zhou Zhao, Hanqing Lu, Vincent W. Zheng, Deng Cai, Xiaofei He, and Yueting Zhuang. Community-based question answering via asymmetric multi-faceted ranking network learning. In *Proceedings of the Thirty-First Conference on Artificial Intelligence, February*, pages 3532–3539. AAAI Press, 2017.

[Zhu *et al.*, 2011] Hengshu Zhu, Huanhuan Cao, Hui Xiong, Enhong Chen, and Jilei Tian. Towards expert finding by leveraging relevant categories in authority ranking. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 2221–2224, 2011.