EndCold: An End-to-End Framework for Cold Question Routing in Community Question Answering Services

Jiankai Sun[1*], **Jie Zhao**[2], **Huan Sun**[2] and **Srinivasan Parthasarathy**[2]
[1]ByteDance Inc.
[2]Department of Computer Science and Engineering, The Ohio State University
jiankai.sun@bytedance.com, {zhao.1359, sun.397}.osu.edu, srini@cse.ohio-state.edu

Abstract

Routing newly posted questions (a.k.a cold questions) to potential answerers with the suitable expertise in Community Question Answering sites (CQAs) is an important and challenging task. The existing methods either focus only on embedding the graph structural information and are less effective for newly posted questions, or adopt manually engineered feature vectors that are not as representative as the graph embedding methods. Therefore, we propose to address the challenge of leveraging heterogeneous graph and textual information for cold question routing by designing an end-to-end framework that jointly learns CQA node embeddings and finds best answerers for cold questions. We conducted extensive experiments to confirm the usefulness of incorporating the textual information from question tags and demonstrate that an end-to-end framework can achieve promising performances on routing newly posted questions asked by both existing users and newly registered users.

1 Introduction

Community Question Answering services (CQAs) such as Stack Exchange and Yahoo! Answers are examples of social media sites, with their usage being examples of an important type of computer supported cooperative work in practice. In recent years, the usage of CQAs has seen a dramatic increase in both the frequency of questions posted and general user activity. This, in turn, has given rise to several interesting problems ranging from expertise estimation to question difficulty estimation, and from automated question routing to incentive mechanism design on such CQAs [Yang *et al.*, 2013; Fang *et al.*, 2016; Sun *et al.*, 2018a].

In this work, we focus on the problem of routing newly posted questions to suitable experts before answers are written (item cold-start). Usually, there are two types of questions in CQAs – resolved (questions with answers) and newly posted questions (questions that have not received any answers). The newly posted questions may themselves be posted by new askers (such as newly registered users who have not asked a question earlier) or existing askers (such as users who have asked several questions previously). We refer to these newly posted questions as *cold questions*. The majority of approaches have focused on evaluating content quality after the fact (after questions have been resolved) [Yang *et al.*, 2013]. Yet, as the CQAs continue to grow, routing the cold questions to matching experts before answers have been provided has become a critical problem. For example, in Stack Overflow, about 5.22 million questions have not been answered [1].

Recently, the usage of graphs has seen an increase in solving the question routing problem. For example, *QDEE* [Sun *et al.*, 2018a] was proposed to leverage Expertise Gain Assumption (EGA) to build the competition graph and then applied social agony [Tatti, 2014; 2015] to infer graph hierarchy [Sun *et al.*, 2017] and assign each node a scalar value to represent where it stands in the competition graph. The assigned scalar value of question nodes are question difficulty scores, and the assigned scalar value of user nodes are user expertise scores. *QDEE* then routed cold questions to users with matching expertise based on the corresponding questions' difficulty level. It's worth mentioning that both question difficulty and user expertise follow the characteristic of asymmetric transitivity. For example, given a question q_1 is easier than q_2 and q_2 is easier than q_3, we can infer that q_1 is easier than q_3 conveniently. Then, *ATP* [Sun *et al.*, 2019] was proposed to tackle the challenge of directed graph embedding with asymmetric transitivity preservation and then it leveraged the generated embedding results to appropriately route and assign newly posted questions to users with the suitable expertise and interest in CQAs. However, a multiple pipeline including directed graph embedding generation and experts finding is required where each step has to be optimized separately. Another drawback is that *ATP* can only leverage graph structure information, while other information (such as textual information) that does not follow the asymmetric transitivity can not be incorporated.

In Stack Exchange sites, voting is central for providing quality questions and answers [2]. The more that people vote on a post, the more certain future visitors can be

[*]The work was mainly done when the first author was a Ph.D. student at the Ohio State University.

[1]https://stackexchange.com/sites

[2]https://stackoverflow.com/help/why-vote

confident of the quality of information contained within the post. Voting indicates a CQA community's long-term review for a given user's expertise level under a specific topic. ColdRoute [Sun *et al.*, 2018b] views the problem of identification of best answerers as a regression problem (finding the answerers who have the highest predicted voting scores). The input features of *ColdRoute* are users' past asking and answering activities in CQAs, and *ColdRoute* can leverage Factorization Machines (FMs) [Rendle, 2010; 2012] to model pairwise interactions among different objects (questions, askers, answerers, and question tags). While in theory FMs can model high-order interactions, in practice it only models order-2 feature interactions due to the high complexity [Guo *et al.*, 2017].

Therefore, we propose to solve the cold question routing problem by marrying the merits of the above two types of methods. On one hand, we use graph embedding techniques to encode useful high-order graph structure information, and on the other hand, we enable the incorporation of heterogeneous information (e.g. textual) in the node embedding as well. Specifically, *users* (including both askers and answerers), *questions*, and *question tags* are represented as three different types of nodes, which are connected by their historical interactions into an undirected CQA graph. We then employ a Graph Convolutional Network (GCN) [Kipf and Welling, 2016] to generate deeper and neighbor-aware node representations in the undirected graph. Our proposed method allows more flexible node interactions and can enrich the question/user node embeddings with the textual information from question tags. And question tags can bridge the gap between cold questions and exiting nodes in the CQA graph. This overall end-to-end framework can be optimized directly with the cold question routing objective, which enables the learning of task-specific embeddings for heterogeneous nodes in the CQA graph, and a better cold question routing performance.

In summary, we made the following contributions:

- Encode users' past asking and answering activities and textual information by using an undirected heterogeneous graph, of which nodes are questions, tags, and users (askers and answerers) and edges are interactions among them.
- Build an end-to-end framework which can route cold questions in CQAs effectively.
- Demonstrate the effectiveness of our proposed framework on various Stack Exchange sites.

2 Related Work

Recently, some graph embedding methods [Fang *et al.*, 2016; Zhao *et al.*, 2016; 2017; Sun *et al.*, 2019] which can model the users' past asking and answering activities by using a graph are proposed to address the question routing problem. *NeRank* [Zeyu Li, 2019] optimizes the objectives of graph embedding and question routing alternatively, and *NeRank* leverages the partial order constraints in the ranking loss. *NeRank* has to use an extra Long short-term memory (LSTM) to learn the embedding of the text. *EndCold* can learn the embeddings of text, questions, and users simultaneously.

By iteratively introducing questions tags and textual descriptions (question title and body), *ColdRoute* [Sun *et al.*, 2018b] demonstrated that question tags play a more important role than question title and body in terms of experts finding. However, ColdRoute fails to take the higher-order interactions among questions, askers, answerers, and question tags into consideration due to high time complexity [Guo *et al.*, 2017].

2.1 Problem Statement

Assume we are given four relational sets of data in terms of Questions $\mathcal{Q} = \langle q_1, q_2, \ldots, q_n \rangle$, Askers $\mathcal{A} = \langle a_1, a_2, \ldots, a_m \rangle$, Answerers $\mathcal{U} = \langle u_1, u_2, \ldots, u_k \rangle$, and Question Tags $\mathcal{T} = \langle t_1, t_2, \ldots, t_l \rangle$. For each question $q_i \in \mathcal{Q}$, we have a tuple of the form $\langle Asker_i$, $Answerers_i$, $BestAnswerer_i$, $Tags_i$, and $Scores_i \rangle$, where $Asker_i \in \mathcal{A}$, $Answerers_i \subset \mathcal{U}$, $BestAnswerer_i \in \mathcal{U}$, $Tags_i \subset \mathcal{T}$. Each voting $score \in Scores_i$ is an integer, which is calculated based on the difference between $Answerer_i$'s up-votes and down-votes which are assigned to it by users who viewed the question or provided answers for that in the CQA environment. Note that the $BestAnswerer$ for a question may not be specified by $Asker$.

Given the preliminaries (above), in this work, we focus on the problem of routing newly posted questions to suitable experts before answers are written (item cold-start). Each quadruple case $\langle q, u, a, t \rangle$, where $q \in \mathcal{Q}$, $u \in \mathcal{U}$, $a \in \mathcal{A}$, $t \subset \mathcal{T}$ has a voting score $y \in \mathbb{R}$, which is equal to the difference between times of up-voting and down-voting. Given the testing question set $\mathcal{Q}_t$, the predicted ranking list of all potential answerers for a test question q^* is R^{q^*} for all $q^* \in \mathcal{Q}_t$. The answerer who's ranked top 1 among all candidates in R^{q^*} for the corresponding target question q^* will be selected as the best answerer for q^*.

3 Methodology

In this section, we talk about the design of our proposed framework (both sequential and end-to-end). The key steps of the sequential and end-to-end model for routing newly posted questions are shown in Figure 1. We refer the sequential model as *Seq* and the end-to-end model as *EndCold*. The key steps of *Seq* are:

1. Build the undirected heterogeneous CQA graph which incorporates users past asking and answering activities and their corresponding textual information in CQAs
2. Apply suitable graph embedding methods to learn representations for each node (including question, user, and tag) in the corresponding input CQA graph
3. Model the question routing problem as a regression problem, and concatenate the embeddings of question, asker, answerer, and tag as feature vectors and use corresponding voting scores as targets to train a suitable regression model f
4. Apply f to predict the voting score a candidate user can achieve on a given cold question q^*

5. Routing q^* to its corresponding predicted best answerer, identified by selecting the user who's predicted to achieve the highest voting score among all candidate answerers

The biggest difference between our sequential model *Seq* and end-to-end framework *EndCold* is that *EndCold* optimizes the graph embedding module (step 2 in *Seq*) and regression module (step 3 in *Seq*) simultaneously as shown in Figure 1.

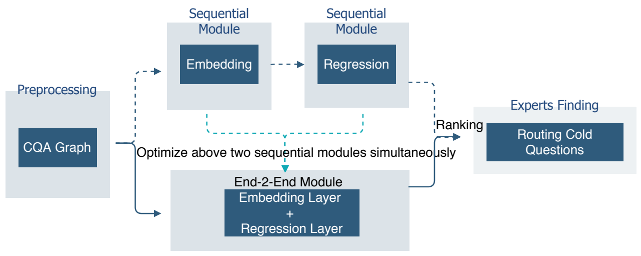


Figure 1: Illustration of the sequential and end-to-end framework for cold question routing in CQAs

3.1 Build the Undirected CQA Graph

Figure 2 illustrates how we build the corresponding heterogeneous CQA graph of which nodes are questions, tags, and users (including askers and answerers) and edges represent the interactions among questions, askers, answerers, and question tags. A (question, asker) edge represents the asking activity. A (question, answerer) edge indicates the answering activity. A (question, tag) edge represents the textual information. We use different ids (node type + integer) to distinguish nodes. For example, user 1 in Figure 2 will be represented as $u1$, and question 1 will be represented as $q1$.

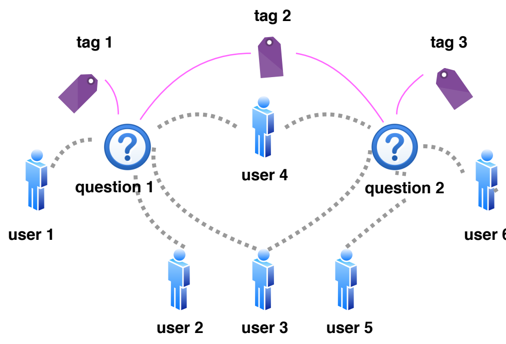


Figure 2: Illustration of the input CQA Graph with question tags: user 1 asked a question 1 which has tags 1 and 2. User 2, 3, and 4 answered question 1. User 4 asked a question 2 with tag 2 and 3. Question 2 was answered by user 3, 5 and 6.

Both users' past asking and answering activities and their corresponding textual information (question tags) are incorporated in the above CQA graph, which can help us bridge the gap between cold question nodes and existing nodes in the CQA graph. Cold questions can be connected with the existing CQA graph via the common question tag nodes they have and existing asker node (only applicable for cold questions asked by existing askers).

3.2 The Sequential Module for Cold Question Routing

In this section, we talk about the sequential model named as *Seq-TA* (By default, *Seq* is referred to *Seq-TA* in our paper) which takes the undirected CQA Graph with question tags as the input and leverages any suitable graph embedding algorithms to generate embeddings for the question, user, and tag nodes. Take our experiments for example, we use node2vec [Grover and Leskovec, 2016] to generate corresponding embeddings[3]. Each answering thread between a question and an answerer can be represented as a quadruple of the target question, its asker, the corresponding answerer, and question tags. The corresponding quadruple case $\langle q, u, a, t\rangle \in \langle \mathcal{Q}, \mathcal{U}, \mathcal{A}, \mathcal{T}\rangle$ can be represented as a concatenated feature vector $(\vec{q}, \vec{u}, \vec{a}, \vec{t})$, where $\vec{q}$, $\vec{u}$, and $\vec{a}$ are the embedding vectors of question q, answerer u, and asker a respectively. $\vec{t}$ is the average embedding of tags in set t. The voting score of u achieved on q is an integer. A regression model f is trained based on the above feature vectors as inputs and corresponding voting scores as the targets.

This design gives us the flexibility to explore the different features' relative importance in cold question routing. For example, information of asker and question tags can be iteratively introduced to our model to explore their relative importance as follows:

- *Seq-A*: explore the importance of question asker by using triples of $\langle \mathcal{Q}, \mathcal{U}, \mathcal{A}\rangle$ on routing cold questions asked by existing askers. Other settings are as same as *Seq-TA*.
- *Seq-T*: explore the importance of question tags by using triples of $\langle \mathcal{Q}, \mathcal{U}, \mathcal{T}\rangle$ on routing cold questions either from existing askers or new askers. Other settings are as same as *Seq-TA*.
- *Seq-Un*: explore the importance of question asker and tags by using tuples of $\langle \mathcal{Q}, \mathcal{U}\rangle$ on routing cold questions. Other settings are as same as *Seq-TA*.

Regarding the regression module, we examined two types of SVM based regressors implemented by scikit-learn. One is a Epsilon-Support Vector Regression (SVM)[4] with the polynomial kernel (degree is set as 2). Another is the LinearSVR[5] with the kernel type as a linear function. A neural network based regressor which is implemented based on Keras [6] has also been taken into examination.

3.3 The End-to-End Module for Cold Question Routing

The end-to-end module consists of two layers: embedding layer and regression layer. The embedding layer is built based

[3]Other state-of-the-art undirected graph embedding methods can be applied to this module too

[4]http://scikit-learn.org/stable/modules

[5]http://scikit-learn.org/stable/modules

[6]https://keras.io

on the Graph Convolutional Networks (GCNs) [Kipf and Welling, 2016]. The core idea behind GCNs is to learn how to iteratively aggregate feature information from local graph neighborhoods using neural networks [Ying *et al.*, 2018]. A single "convolution" operation transforms and aggregates feature information from a node's one-hop graph neighborhood, and by stacking multiple such convolutions information can be propagated across far reaches of a graph [Ying *et al.*, 2018]. For example, in our undirected CQA graph, a question tag node can propagate its semantic information to other nodes and accumulate information from the asker and answerer nodes. The embedding of the question tag can be enriched through above propagation and accumulation, which is beneficial for cold question routing since question tags are the only information we can leverage for questions asked by new askers. The embedding layer is followed by a multi-layer perceptron (MLP) to do the regression. In our experiments, we use 2 layers in GCNs and 3 hidden layers in MLP. We evaluate the mean squared error for all the training examples.

3.4 Experts Finding for Cold Questions

The final step of both *Seq* and *EndCold* is to identify the best answerers for cold questions, which works as follows:

1. Given a cold question q^* and a set of potential answerers C_q^*, predict each candidate u's voting score for q^*, where $u \in C_q^*$
2. Select the user who achieves the highest voting score as the best answerer for q^*

Given a newly posted question q^* asked by an asker a using tags t and a potential answerer u, the prediction function f_{Seq} in *Seq* can treat the new question as a missing value by $f_{Seq}(\vec{0}, \vec{u}, \vec{a}, \vec{t})$. The user u who achieves the highest value of $f_{Seq}(\vec{0}, \vec{u}, \vec{a}, \vec{t})$ will be selected as the best answerer for question q^*. It's possible that the newly posted question q^* can be asked by a new asker who has no information to learn the corresponding embedding vector. Then the prediction function f_{Seq} can treat the new asker as a missing value by $f_{Seq}(\vec{0}, \vec{u}, \vec{0}, \vec{t})$.

Instead of using missing value $\vec{0}$ to represent the feature vector of q^* in *Seq*, *EndCold* can leverage GCNs to propagate information from existing tag nodes and asker nodes (if applicable) to q^*. q^* can accumulate these information and get its corresponding embedding vector as $\vec{q^*}$. The regression function $f_{EndCold}$ in *EndCold* can predict the voting score u can achieve on q^* as $f_{EndCold}(\vec{q^*}, \vec{u}, \vec{a}, \vec{t})$.

It is possible that the potential answerer u is a newly registered user who has not provided any answer before in CQAs (user cold-start). In this scenario, the prediction function can be simplified as $f(\vec{q^*}, \vec{0}, \vec{a}, \vec{t})$. All new registered users will receive the same predicted voting score for the same target question. More efforts will be spent to make accurate predictions for the user cold-start problem in our future work.

4 Experiments and Analyses

We conducted our cold question routing experiments following the same settings as ColdRoute on 8 large Stack Exchange sites [Sun *et al.*, 2018b]. We compare our proposed models with state-of-the-art methods based on two popular evaluation criteria (**Precision**@3 [Zhu *et al.*, 2014; Zhao *et al.*, 2016; Fang *et al.*, 2016; Zhao *et al.*, 2017; Sun *et al.*, 2018b; 2019] and **Accuracy** [Zhao *et al.*, 2016; Fang *et al.*, 2016; Zhao *et al.*, 2017; Sun *et al.*, 2018b; 2019])[7].

4.1 Performance of Different Sequential Models

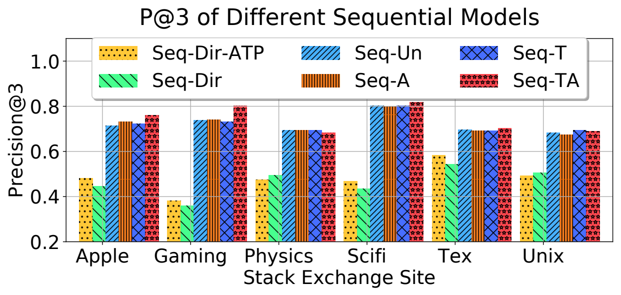
In this section, we compare the routing performance of different sequential models with different embedding methods (such as *node2vec* and *ATP*) on various input graphs (such as directed competition graph used by *QDEE* and undirected CQA graph with question tags proposed in this work) with our proposed models. The key steps of each method in this section follow the same sequential framework as we discussed in Methodology section. The differences fall in how to build the input graph and get the embedding vector for each node in the corresponding input graph. The details of all comparison partners are as follows:

- **Seq-Dir-ATP**: The input graph of *Seq-Dir-ATP* is the same directed competition graph used in *QDEE*. Embeddings of nodes (questions and users) are generated by *ATP*. Given a newly posted question q^*, *Seq-Dir-ATP* applies Expertise Gain Assumption (*EGA*) [Sun *et al.*, 2018a] to use the source and target embedding of the question which has the highest difficulty level among all questions asked by the same asker to approximate q^*'s source and target embedding respectively. Given a candidate user u, the number of votes u can achieve on q^* can be predicted by feeding the concatenated source and target feature vectors of q^* and u to a regression model f, which can be represented as $f(\vec{q_s^*}, \vec{q_t^*}, \vec{u_s}, \vec{u_t})$, where $\vec{\cdot_s}$ and $\vec{\cdot_t}$ represent corresponding node's source and target feature vector respectively. The dimension of both source and target feature vector is tuned to set as 128. Hence the dimension of concatenated feature for f is 512.
- **Seq-Dir**: It uses the same directed competition graph as *Seq-Dir-ATP*. However, *Seq-Dir* leverages node2vec [Grover and Leskovec, 2016] to generate embeddings for question and user nodes. Like *Seq-Dir-ATP*, *EGA* is applied to generate embeddings for newly posted questions. Feature vector representations of q^* and u are concatenated as inputs of the regression model. Parameters are tuned to set as $p = q = 1, d = 128$[8]. Hence the dimension of concatenated feature for f is 256.

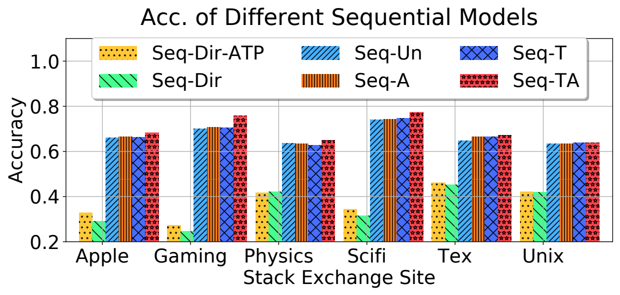
It's worth mentioning *Seq-Dir-ATP* and *Seq-Dir* cannot handle the scenario of cold questions asked by new askers since *EGA* can only be applied to questions asked by existing askers to infer the embedding vector of the newly posted question. Hence, we only show their corresponding performance on routing cold questions asked by existing askers.

[7]Other evaluation criteria such as Precision@1 and MRR [Zhu *et al.*, 2014] used in ColdRoute [Sun *et al.*, 2018b] follow the same performance pattern as Precision@3 and Accuracy. To save space, we don't report their results here.

[8]https://github.com/snap-stanford/snap/



(a) Precision@3



(b) Accuracy

Figure 3: Performance of Different Sequential Models on Routing Cold Questions Posted by Existing Askers

	Site	Apple	AskUbuntu	Gaming	Physics	Scifi	Serverfault	Tex	Unix
P@3	ColdRoute-T	0.6581	0.6274	0.7796	**0.7194**	0.7742	0.6074	0.6343	0.6869
	Seq (pointwise)	0.7607	0.668	0.8019	0.6837	0.8172	0.6	**0.7029**	0.6902
	Seq (pairwise)	**0.7692**	0.7345	0.8147	0.6786	**0.8208**	0.6117	0.6286	**0.7003**
	EndCold	0.7607	**0.743**	**0.8339**	0.7041	**0.8208**	**0.6222**	0.6914	0.6902
Acc	ColdRoute-T	0.6324	0.6054	0.7387	0.6354	0.7369	0.5807	0.5802	0.6404
	Seq (pointwise)	0.6836	0.6572	0.759	0.65	0.7734	0.5917	**0.6719**	0.6381
	Seq (pairwise)	**0.7079**	0.7092	0.7656	0.6404	0.7619	0.6005	0.5888	**0.648**
	EndCold	0.7022	**0.7189**	**0.778**	**0.6733**	**0.7787**	**0.607**	0.6495	0.6401

Table 1: Performance on newly posted questions asked by existing askers in 8 different Stack Exchange sites

The experiment results are summarized in Figure 3, where we can make the following observations:

- The best overall performance is achieved by *Seq-AT*, which concatenates the embedding of all four nodes in the $(\vec{q}, \vec{u}, \vec{a}, \vec{t})$ quadruple as input features and feeds them to the regression model. The fact that *Seq-AT* consistently outperforms its feature-ablated variants *Seq-A*, *Seq-T* and *Seq-Un* gives us some guideline that it is necessary to incorporate both user past activities (asking and answering) and textual information when designing question routing algorithms.
- For all the CQA sites and under all metrics, the four methods using the undirected CQA graph with question tags as input perform significantly better than *Seq-Dir* and *Seq-Dir-ATP*) which use the direct competition graph. It demonstrates the advantage of using our proposed undirected CQA Graph with question tags in comparison with using the directed competition graph leveraged by *QDEE* [Sun *et al.*, 2018a] and *ATP* [Sun *et al.*, 2019] on the task of cold question routing.
- Note that despite *Seq-Un* does not explicitly use the embedding of question tag nodes, it still significantly outperforms *Seq-Dir*. This is because the graph embedding algorithm (node2vec [Grover and Leskovec, 2016]) in our experiments) allows neighboring nodes to share information, so the question node embeddings are regulated by their assigned tags as well.

4.2 Pointwise V.S. Pairwise

In this section, We would like to compare the performance difference between pointwise and pairwise based sequential models on cold question routing in CQAs, with hoping that the corresponding results can guide our future design of the question routing system in CQAs. The difference between pointwise and pairwise based sequential models are listed as follows:

- **Pointwise**: its goal is to learn a regression function $f : \langle q, u, a, t\rangle \rightarrow \mathbb{R}$. The user $u \in \mathcal{U}$ who achieves the highest value of $f(q, u, a, t)$ will be selected as the best answerer for question q.
- **Pairwise**: We introduce the relative quality rank to model users' past answering activities in CQAs, which is in the form of (q, a, t, u_+, u_-), meaning that the answer provided by answerer u_+, obtains more voting scores than the answer provided by the answerer u_- for question q asked by asker a with tags set t. Let $\mathcal{C} = \{(q, a, t, u_+, u_-)\}$ denote the set of ranking pair constraints derived from the community votes. More formally, we aim to learn a ranking function f that for any $(q, a, t, u_+, u_-) \in \mathcal{C}$, the inequality holds: $f(q, a, t, u_+) > f(q, a, t, u_-) + \epsilon$. The user $u_+ \in \mathcal{U}$ who satisfies the most number of above constraints will be selected as the best answerer for question q. Given the representations of question and answerers, and the relative quality rank in CQA sites, the loss function $\mathcal{L}_r$ in Pairwise is designed as follows:

	Site	Apple	AskUbuntu	Gaming	Physics	Scifi	Serverfault	Tex	Unix
P@3	ColdRoute-T	0.5171	0.61	0.7	**0.7249**	0.7081	0.6074	0.6894	0.6525
	ColdRoute-TA	0.5589	0.6013	0.6688	0.7205	0.6957	0.5617	0.6439	0.6419
	Seq (pointwise)	0.6502	0.6863	0.7625	0.7205	**0.764**	0.6267	0.7121	0.6499
	Seq (pairwise)	0.6616	0.6885	0.7625	0.6681	0.7453	0.61	0.7045	0.671
	EndCold	**0.7034**	**0.7102**	**0.8**	**0.7249**	0.7516	**0.63**	**0.7349**	**0.6976**
Acc	ColdRoute-T	0.5247	0.5809	0.6591	0.6838	0.6841	0.5807	0.5747	0.6034
	ColdRoute-TA	0.5555	0.5797	0.6327	0.6810	0.6761	0.5460	0.5700	0.5995
	Seq (pointwise)	0.6179	0.6219	0.7123	0.6853	**0.7356**	**0.5955**	0.6069	0.6163
	Seq (pairwise)	0.6326	0.6455	0.708	0.6604	0.7354	0.578	0.587	0.6261
	EndCold	**0.6743**	**0.6881**	**0.7447**	**0.6887**	0.731	0.5915	**0.6227**	**0.6393**

Table 2: Performance on newly posted questions asked by new askers in 8 different Stack Exchange sites

$$\mathcal{L}_r = \sum_{(q,a,t,u_+,u_-)\in\mathcal{C}} = \max(0, \epsilon + f^-(q,a,t,u_-) - f^+(q,a,t,u_+))) \quad (1)$$

where $f^+(\cdot)$ denotes the voting score achieved by high-quality answerers and $f^-(\cdot)$ represents the voting score of low-quality answerers for question routing. The hyper-parameter ϵ ($\epsilon > 0$) controls the margin in the loss function and $\mathcal{C}$ is the set of pairwise relative ranking constraints.

In our experiments, the pairwise based model *Seq (pairwise)* leveraged SVM^{rank} [Joachims, 2006] to learn the partial order of two potential answerers given a newly posted question. As shown in Table 1 (cold questions asked by existing askers) and 2 (cold questions asked by new askers), *Seq (pairwise)* does not show significant improvement over the pointwise model *Seq (pointwise)* consistently. The reason is that *Seq (pairwise)* suffers from the data sparseness problem. The average number of answers per question among the 8 Stack Exchange sites is 1.79, and 57.19% of questions have only one answer, which indicate that *Seq (pairwise)* cannot extract enough valid question-answerer pairs for training. With CQAs growing and information of answers becoming dense, *Seq (pairwise)* can have more valid relative ranking constraints in $\mathcal{C}$ to train and then be more robust and effective.

4.3 Sequential Model V.S. End-to-End Model

In this section, we aim to compare the difference between our sequential model and end-to-end model. As shown in Table 1 and 2, the overall performance of end-to-end model *EndCold* is better than the performance of sequential models (both pointwise and pairwise). For example, *EndCold* improves upon the evaluation metrics (Accuracy and Precision@3) over *Seq* (pointwise) by 3.37% and 3.33% respectively on routing cold questions asked by new askers. Above results demonstrate the effectiveness of our end-to-end framework in comparing with the sequential models.

4.4 Comparison with the State-of-the-Art on Cold Question Routing

In this section, we compared our proposed model with *ColdRoute* [Sun *et al.*, 2018b], a state-of-the-art model on routing cold questions posted by both new and existing askers to suitable experts. *ColdRoute-A*, *ColdRoute-T*, and *ColdRoute-TA* are three variants of *ColdRoute* and they were designed to explore the importance of asker information, question tags, and both respectively. We also compare our model with some semantic matching based approaches [Srba and Bielikova, 2016; Li and King, 2010; Li *et al.*, 2011; Dong *et al.*, 2015; Yang and Manandhar, 2014; Szpektor *et al.*, 2013; Yang *et al.*, 2013; Figueroa and Neumann, 2013; Zhou *et al.*, 2013; Guo *et al.*, 2008; Ji *et al.*, 2012; Le and Mikolov, 2014]. Due to space limitation, we mainly report the best comparison partners (*ColdRoute-T* and *ColdRoute-TA*) here. As shown in Table 1 and 2, our proposed models perform better than *ColdRoute-T* on routing cold questions asked by both existing askers and new askers, which demonstrates the effectiveness of our proposed framework.

5 Conclusion and Future Work

In this paper, we proposed an end-to-end framework for cold question routing in CQAs. The input of the framework is an undirected heterogeneous graph with encoding users' past asking and answering activities and textual information of questions. Our proposed model can leverage the higher-order graph structure and content information to embed nodes in the input graph, and do the question routing simultaneously. Extensive experiments show that our model performs better than the state-of-the-art on routing cold questions (asked by both existing askers and new askers). In order to increase the expertise of the entire community, we plan to address the problem of routing cold questions (item cold-start) to newly registered users (user cold-start) in CQAs in our future work.

Acknowledgments

This work is supported by NSF grants EAR-1520870, CCF-1645599, CCF-1629548, IIS-1550302, IIS-1815674, CNS-1513120, the Army Research Office under cooperative agreements W911NF-17-1-0412 and a grant from the Ohio Supercomputer Center (PAS0166). All content represents the opinion of the authors, which is not necessarily shared or endorsed by their sponsors.

References

[Dong *et al.*, 2015] H. Dong, J. Wang, H. Lin, B. Xu, and Z. Yang. Predicting best answerers for new questions: An approach leveraging distributed representations of words in community question answering. In *FCST*, 2015.

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

[Figueroa and Neumann, 2013] Alejandro Figueroa and Günter Neumann. Learning to rank effective paraphrases from query logs for community question answering. In *AAAI*, 2013.

[Grover and Leskovec, 2016] Aditya Grover and Jure Leskovec. Node2vec: Scalable feature learning for networks. In *KDD*, 2016.

[Guo *et al.*, 2008] Jinwen Guo, Shengliang Xu, Shenghua Bao, and Yong Yu. Tapping on the potential of qa community by recommending answer providers. In *CIKM*, 2008.

[Guo *et al.*, 2017] Huifeng Guo, Ruiming TANG, Yunming Ye, Zhenguo Li, and Xiuqiang He. Deepfm: A factorization-machine based neural network for ctr prediction. In *IJCAI*, 2017.

[Ji *et al.*, 2012] Zongcheng Ji, Fei Xu, Bin Wang, and Ben He. Question-answer topic model for question retrieval in community question answering. In *CIKM*, 2012.

[Joachims, 2006] Thorsten Joachims. Training linear svms in linear time. In *KDD*, 2006.

[Kipf and Welling, 2016] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[Le and Mikolov, 2014] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *ICML*, 2014.

[Li and King, 2010] Baichuan Li and Irwin King. Routing questions to appropriate answerers in community question answering services. In *CIKM*, 2010.

[Li *et al.*, 2011] Baichuan Li, Irwin King, and Michael R. Lyu. Question routing in community question answering: Putting category in its place. In *CIKM*, 2011.

[Rendle, 2010] Steffen Rendle. Factorization machines. In *ICDM*, 2010.

[Rendle, 2012] Steffen Rendle. Factorization machines with libfm. In *ACM Trans. Intell. Syst. Technol.*, 2012.

[Srba and Bielikova, 2016] Ivan Srba and Maria Bielikova. A comprehensive survey and classification of approaches for community question answering. *ACM Trans. Web*, 2016.

[Sun *et al.*, 2017] Jiankai Sun, Deepak Ajwani, Patrick K. Nicholson, Alessandra Sala, and Srinivasan Parthasarathy. Breaking cycles in noisy hierarchies. In *WebSci*, 2017.

[Sun *et al.*, 2018a] Jiankai Sun, Sobhan Moosavi, Rajiv Ramnath, and Srinivasan Parthasarathy. QDEE: Question Difficulty and Expertise Estimation in Community Question Answering Sites. In *ICWSM*, 2018.

[Sun *et al.*, 2018b] Jiankai Sun, Abhinav Vishnu, Aniket Chakrabarti, Charles Siegel, and Srinivasan Parthasarathy. Coldroute: effective routing of cold questions in stack exchange sites. In *ECML PKDD*, 2018.

[Sun *et al.*, 2019] Jiankai Sun, Bortik Bandyopadhyay, Armin Bashizade, Jiongqian Liang, P. Sadayappan, and Srinivasan Parthasarathy. Atp: Directed graph embedding with asymmetric transitivity preservation. In *AAAI*, 2019.

[Szpektor *et al.*, 2013] Idan Szpektor, Yoelle Maarek, and Dan Pelleg. When relevance is not enough: Promoting diversity and freshness in personalized question recommendation. In *WWW*, 2013.

[Tatti, 2014] Nikolaj Tatti. Faster way to agony discovering hierarchies in directed graphs. In *ECML PKDD*, 2014.

[Tatti, 2015] Nikolaj Tatti. Hierarchies in directed networks. In *ICDM*, 2015.

[Yang and Manandhar, 2014] B. Yang and S. Manandhar. Exploring user expertise and descriptive ability in community question answering. In *ASONAM*, 2014.

[Yang *et al.*, 2013] Liu Yang, Minghui Qiu, Swapna Gottipati, Feida Zhu, Jing Jiang, Huiping Sun, and Zhong Chen. CQArank: Jointly model topics and expertise in community question answering. In *CIKM*, 2013.

[Ying *et al.*, 2018] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. Graph convolutional neural networks for web-scale recommender systems. In *KDD*, 2018.

[Zeyu Li, 2019] Yizhou Sun Wei Wang Zeyu Li, Jyun-Yu Jiang. Personalized question routing via heterogeneous network embedding. In *AAAI*, 2019.

[Zhao *et al.*, 2016] Zhou Zhao, Qifan Yang, Deng Cai, Xiaofei He, and Yueting Zhuang. Expert finding for community-based question answering via ranking metric network learning. In *IJCAI*, 2016.

[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 *AAAI*, 2017.

[Zhou *et al.*, 2013] Guangyou Zhou, Yang Liu, Fang Liu, Daojian Zeng, and Jun Zhao. Improving question retrieval in community question answering using world knowledge. In *IJCAI*, 2013.

[Zhu *et al.*, 2014] H. Zhu, E. Chen, H. Xiong, H. Cao, and J. Tian. Ranking user authority with relevant knowledge categories for expert finding. In *WWW*, 2014.