Recurrent Memory Reasoning Network for Expert Finding in Community Question Answering

Jinlan Fu, Yi Li
School of Computer Science, Shanghai Key Laboratory of Intelligent Information Processing
Fudan University, Shanghai, China
(fujl16,liy)@fudan.edu.cn

Qi Zhang
School of Computer Science, Shanghai Key Laboratory of Intelligent Information Processing,
Research Institute of Intelligent and Complex Systems
Fudan University, Shanghai, China
qz@fudan.edu.cn

Qinzhuo Wu, Renfeng Ma, Xuanjing Huang
School of Computer Science, Shanghai Key Laboratory of Intelligent Information Processing
Fudan University, Shanghai, China
(qzwu17,rfma17,xjhuang)@fudan.edu.cn

Yu-Gang Jiang
School of Computer Science,
Jilian Technology Group (Video++)
Fudan University, Shanghai, China
ygj@fudan.edu.cn

ABSTRACT

Expert finding is a task designed to enable recommendation of the right person who can provide high-quality answers to a requester's question. Most previous works try to involve a content-based recommendation, which only superficially comprehends the relevance between a requester's question and the expertise of candidate experts by exploring the content or topic similarity between the requester's question and the candidate experts' historical answers. However, if a candidate expert has never answered a question similar to the requester's question, then existing methods have difficulty making a correct recommendation. Therefore, exploring the implicit relevance between a requester's question and a candidate expert's historical records by perception and reasoning should be taken into consideration. In this study, we propose a novel *recurrent memory reasoning network* (RMRN) to perform this task. This method focuses on different parts of a question, and accordingly retrieves information from the histories of the candidate expert. Since only a small percentage of historical records are relevant to any requester's question, we introduce a Gumbel-Softmax-based mechanism to select relevant historical records from candidate experts' answering histories. To evaluate the proposed method, we constructed two large-scale datasets drawn from Stack Overflow and Yahoo! Answer. Experimental results on the constructed datasets demonstrate that the proposed method could achieve better performance than existing state-of-the-art methods.

CCS CONCEPTS

• **Information systems → Recommender systems**; • **Computing methodologies → Neural networks**;

KEYWORDS

Expert Finding, Community Question Answering, Memory Network

ACM Reference Format:
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 *The Thirteenth ACM International Conference on Web Search and Data Mining (WSDM'20), February 3–7, 2020, Houston, TX, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3336191.3371817

WSDM '20, February 3–7, 2020, Houston, TX, USA

ACM ISBN 978-1-4503-6822-3/20/02…$15.00
https://doi.org/10.1145/3336191.3371817

1 INTRODUCTION

Community question answering (CQA) websites, such as Stack Overflow[1], Yahoo! Answers[2], and Quora[3], widely use expert finding tasks in real-world contexts. The need for expert finding arises in many real CQA applications, such as the identification of best answers [24] and question routing [12]. Apart from CQA sites, expert finding is also used in many real applications, including finding appropriate reviewers for a scientific paper [15] and finding the right supervisor for university graduate students[1]. Expert finding in CQA can improve the quality of answers and solve other crucial CQA problems, such as low user response to a question, long wait times for the requester, and poor-quality answers.

Expert finding problems have been approached from different aspects. Various content-based methods [5, 10, 13, 16, 20] have been proposed to perform this task, which learn a user model from past question-answering histories. Riahi et al. [20] proposed the use of a structural topic model to learn a user model. Liu et al.[13] introduced a latent-Dirichlet-allocation-based topic model to learn the question and user topic. Huna et al.[10] proposed a new reputation calculation mechanism based on the quality of answers provided to filter highly active spammers from expert lists. Support-vector-machine-based[16] and gradient-boosted-decision-tree-based[5] methods have also been used for expert finding. Numerous network-based methods have been introduced [14, 19, 29–31] to evaluate the

[1] https://www.stackoverflow.com
[2] https://answers.yahoo.com
[3] https://www.quora.com/

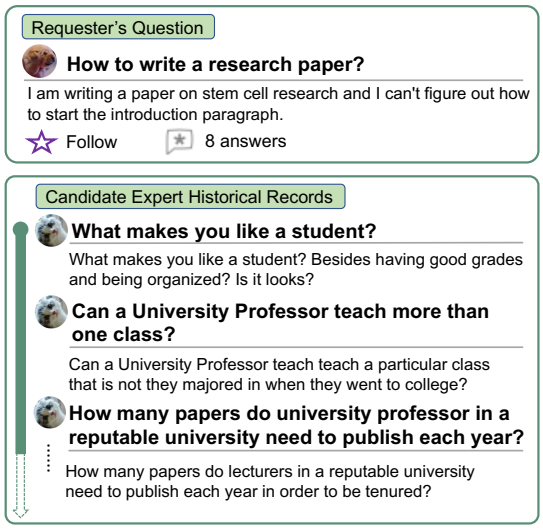

Figure 1: Example from Yahoo! Answer. The top box is a requester's question and the bottom box shows the partial history records of a candidate expert. The bold text indicates question titles, and the plain text the question bodies. Because the candidate expert has never before answered questions similar to the requester's question, most existing methods have difficulty making the correct recommendation. However, we can infer that the candidate expert is a professor who has published several papers (according to the third answered question), and should therefore be able to correctly answer the requester's question about writing a research paper.

authoritativeness of users and to perform expert recommendations. Some studies[8, 27] have used collaborative filtering for making expert recommendations in CQA contexts.

The above methods have tended to only superficially comprehend both the requester's question and the candidate experts' levels of expertise. In some cases, a problem arises when the requester's question is unlike any question previously addressed by the candidate expert. Figure 1 shows an example taken from Yahoo! Answer, in which the requester's question was "How to write a research paper?". The bottom box shows some questions previously answered by a candidate expert. It is difficult to recommend this candidate expert as one who can answer the requester's question when using existing methods, since the content and topic covered by the requester's question and the candidate expert are quite different. However, given the candidate expert's previously answered questions, "*Record 1: What are the characteristics of a student?*", "*Record 2: Can a university professor teach more than one class?*", and "*Record 3: How many papers does a university professor in a reputable university need to publish each year?*", we can infer that this candidate expert is a professor (according to records 1, 2, and 3) who should publish several papers every year (according to record 3). Therefore, the candidate expert has sufficient expertise in writing research papers and has the ability to accurately answer the requester's question. Without performing reasoning, it is impossible to correctly evaluate the various candidate experts.

In this paper, we propose a novel *recurrent memory reasoning network* (RMRN) to achieve this task, which is motivated by *memory, attention, and control (MAC)* [9] algorithm with respect to visual question-answering tasks. Like MAC, our model is a recurrent structure of several cascading reasoning units known as *reasoning memory cells* (RMCs). In each reasoning process, RMCs use an attention mechanism to attend to different aspects of the question, and then accordingly retrieve relevant information from candidate experts' historical records. The retrieved information represents different aspects of the expertise the answerers must have. Unlike MAC, which only utilizes retrieved information to update the memory and then erases that information, we utilize a retrieval memory to store all of the information retrieved during the entire reasoning process. There are many historical records for candidate experts that are irrelevant to the requester's current question. For example, historical records about "*English*" would be mostly irrelevant to a question about "*Java*". Therefore, if we take into consideration the complete historical records of a user, then the recommendation performance may be negatively impacted. To address this issue, we introduce a Gumbel-Softmax-based mechanism to extract some relevant historical records of the candidate expert. There are many out-of-vocabulary (OOV) words in CQAs, such as code segments (*"torch.nn"* and *"torch.nn.functional"*), nonstandard spellings, and oral language. To capture morphological information, we introduce enhanced word embedding augmented with multi-grained word representations, including character, subword, and word levels. To demonstrate the effectiveness of our model, we constructed two large-scale datasets drawn from *Stack Overflow* and *Yahoo! Answer*. The experimental results on our datasets showed that our method achieves better performance than existing methods.

The main contributions of our work is summarized as follows.

(1) We proposed a recurrent memory reasoning network (RMRN)- a recurrent neural network that facilitates explicit reasoning in expert finding. It only attends to partial historical records that are relevant to requester's question, which are selected by Gumbel-Softmax-based mechanism.
(2) To alleviate the OOV introduced by oral language or special expressions such as code segments, we introduced enhanced word embedding augmented with multi-grained word representations, including character, subword, and word levels.
(3) We constructed two large-scale datasets drawn from *Stack Overflow* and *Yahoo! Answer* to evaluate the performances of various existing methods. [4]

2 RELATED WORK

Expert finding aims to identify users who have sufficient expertise in a given topic/query. The most popular approaches to expert finding in CQA systems can be categorized as either topic-based or network-based.

Topic-based methods [7, 13, 20, 26, 28] measure the question-user relationship in the topic space, and can alleviate the problem of lexical gaps. Guo et al.[7] proposed a topic-sensitive probabilistic model for learning question representation. Xu et al. [26] proposed

[4]We will release the constructed datasets.

a probabilistic dual-role model that separately models users' roles as requesters and answerers based on a probabilistic latent semantic analysis model. Riahi et al. [20] utilized the segmented topic model [6] to more realistically model user expertise. Yang et al. [28] considered the expertise of users to be strongly correlated with the topics and voting scores of their previous Q&A posts. Liu et al.[13] proposed the ZhihuRank algorithm, which is based on link structure and topical similarity.

Network-based methods [19, 29–31]evaluate the authoritativeness of users in a user-user network comprising their asking-answering relations. Liu et al. Zhu et al. [31] measured the category relevance of questions and rank user authority on an extended category link graph. Zhao et al. [30] proposed a graph-regularized matrix completion algorithm for predicting ratings based on the social network of users. Zhao et al. [29] proposed a ranking metric network learning framework that exploits users' relative quality rank with respect to given questions as well as their social relations. Qian et al. [19] proposed a weakly supervised factor graph known as WeakFG to model a user's expertise and willingness by considering that users with the same social identities exhibit the same social behaviors. The authors of some studies have used collaborative filtering for making expert recommendations in CQA [8, 27]. Yang and Manandhar [27] employed probabilistic matrix factorization to learn a user-tag matrix, and then used this matrix to recommend experts when given a new question. Huang et al.[8] proposed a tree-guided learning model for recommending experts across multiple collaborative networks.

3 RECURRENT MEMORY REASONING NETWORK

Figure 2 (a) shows the overall architecture of the model. The recurrent memory reasoning network (RMRN) is an end-to-end architecture that repeats an explicit reasoning process by stacking small blocks, knows as reasoning memory cells. The RMRN contains three modules: 1) *Input Module*. We proposed the use of a multi-grain word representation, including character-, subword-, and word-level representations to alleviate the OOV problem in social media. 2) *Reasoning Memory Cells*. Reasoning memory cells (RMCs) comprise the core of the RMRN, with each RMC performing one reasoning step. 3) *Prediction Module*. The prediction module gives a prediction on whether or not to recommend the current candidate expert to answer the given requester's question.

3.1 Input Module

Word-level representation captures the dependency relationships between words, but is unable to handle the unknown words problem. Character-level representation extracts morphological information, and its effectiveness is verified for rare and unknown word representations. To alleviate the OOV problem in social media, we augmented word representation with subword-level representation and ELMo[18] that involves character-level representation.

Assume that a sentence $S_w = [w_1, w_2, w_3, ..., w_n]$ is encoded by a multi-grained word representation into an embedding sequence $S_e = [e_1, e_2, e_3, ..., e_n]$, n is the sentence length. The embedding of e_j is concatenated by three parts:

$$e_i = [e_j^w; e_j^s; e_j^c] \tag{1}$$

where $e_j^w \in \mathbb{R}^d$ is pre-trained word embedding, $e_j^s \in \mathbb{R}^{d_s}$ is subword-level embedding and $e_j^c \in \mathbb{R}^{d_c}$ is contextualized word embedding encoded by pre-trained ELMo encoders.

In our experiments, we used bi-directional gated recurrent units (GRU) [4] to encode the sentence-level representations. Assume that the requester's question is $q = \{qw_1, \cdots, qw_j, \cdots, qw_{n_q}\}$, where n_q is the question length, and qw_j is the embedding of the $j-$th word in the requester's question. The word embedding can be an enhanced word embedding e, or any level of word representation (such as character-level e^c, subword-level e^s, and word-level e^w). We obtained the contextual requester's question representation as follows:

$$q_j = BiGRU(qw_j)$$
$$Q = [q_1, q_2, \cdots, q_j \cdots, q_{n_q}],$$

where q_j is the $j-$th word representation of the question.

As for the representation of the candidate expert historical records, we assume that the historical recods are $H = [H_1, H_2, \cdots, H_{n_H}]$, and each record consists of word sequence $H_j = \{hw_1^j, hw_2^j, \cdots, hw_{n_h}^j\}$, where n_H and n_h are the number of historical records considered and the maximum sentence length of each historical record, respectively. The word embedding can be an enhanced word embedding e or any level of word representation (such as character-level e^c, subword-level e^s, and word-level e^w). We keep the sentence-level representation of each record.

$$y_j = BiGRU(H_j)$$
$$H = [y_1, y_2, \cdots, y_j, \cdots, y_{n_H}],$$

where y_j is the $j-$th historical record representation and H is the representation of the candidate expert historical records.

3.2 Reasoning Memory Cell

Figure 2 (b) shows the structure of the RMCs. The initial memory state m_0 is the question representation q_{n_q}, and the initial control state c_0 is the random vector of dimension d. RMCs have three units: the control, read, and write units. We briefly describe their internal mechanisms below. 1) *Control unit*: This unit points to future work or the next aspect of user expertise that should be considered. In n iterations (n RMCs), the RMRN uses an attention mechanism to iteratively focuses on different parts of a requester's question. 2) *Read unit*: This unit has access to the candidate expert's historical records, and retrieves the information that is relevant to the current control state. Because records that are irrelevant to the requester's question may negatively impact the performance, we introduced a Gumbel-Softmax based mechanism, which selects only those records that are relevant. 3) *Write unit*: This unit updates the memory by erasing some old information and writing some newly retrieved information.

3.2.1 The Control Unit. We expect the control unit to capture different aspects of the expertise required by answerers to answer the requester's question. The control unit interacts with the read and write units. We utilize the previous control state $\mathbf{c}_{i-1}$ and the question's contextual words $Q = [q_1, q_2, \cdots, q_{n_q}]$ to explore the next aspect of expertise should be considered.

To facilitate the RMCs' ability to readily explore different aspects of the question, like the MAC[9], we utilize different parameters W_i

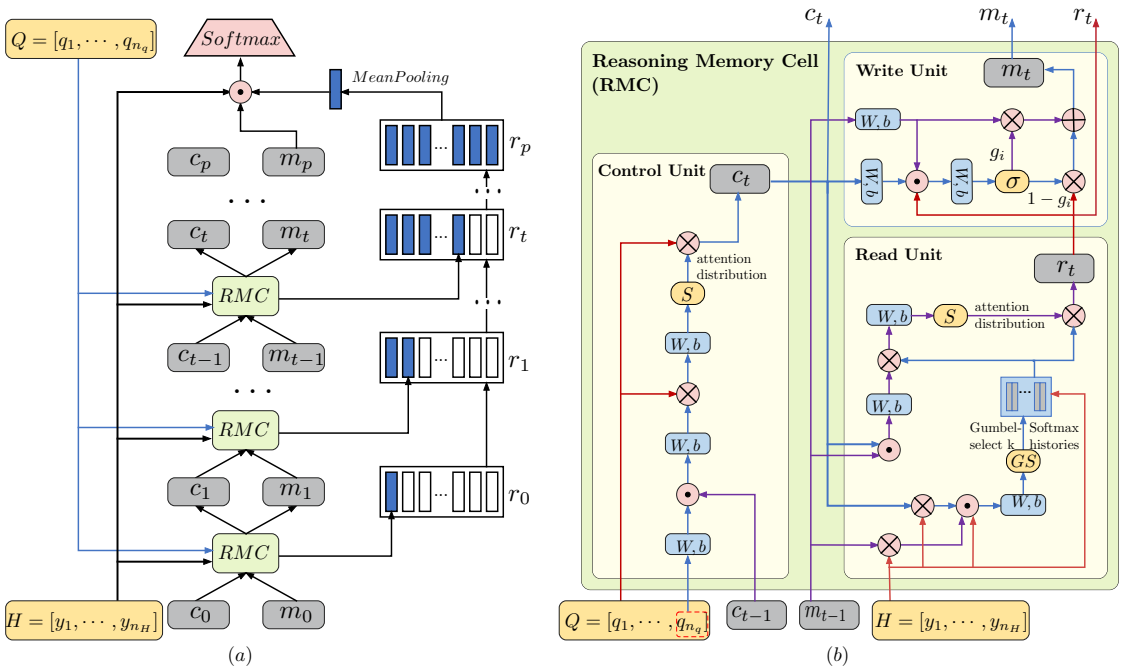

Figure 2: Illustration of a recurrent memory reasoning network (a) and its component reasoning memory cell (b). H is the candidate expert's historical records representation, Q is the requester's question representation, and c_0 and m_0 are the initial control and memory unit state, respectively.

and b_i to linearly transform the question representation q_{n_q} (we utilized the last hidden state of BiGRU as the sentence representation) at each iteration.

$$q_i = W_i q_{n_q} + b_i, \tag{2}$$

where $W_j \in \mathbb{R}^{d\times d}$ and $b_j \in \mathbb{R}^d$ are parameters, $i = 1, 2, \cdots, p$ is the iteration steps. In each iteration, the linearly transformed parameters W_i and b_i are different. Then, we concatenate the new question representation q_i and previous control state c_{i-1} and perform a linear transformation again to yield a d-dimensional vector cq_i:

$$cq_i = W_{cq}[q_i, c_{i-1}] + b_{cq}, \tag{3}$$

where $W_{cq} \in \mathbb{R}^{2d\times d}$ and $b_{cq} \in \mathbb{R}^d$ are parameters.

To prevent the MRC from diverging during the reasoning process and to maintain its focus on the words in the question, we compute the following c_i question-guided word attention:

$$\alpha_i = softmax(W_\alpha(cq_i \otimes Q) + b_\alpha),$$
$$c_i = \sum_{j=1}^{n} \alpha i \cdot Q,$$

where $\otimes$ is the element-wise product, $W_\alpha \in \mathbb{R}^{d\times 1}$ is the parameter, α_i is the attention distribution, and c_i is the current control state.

3.2.2 The Read Unit. The read unit has access to the user history H so it can retrieve information that is related to the current reasoning task.

We compute the interaction between the history records H and the previous memory m_{i-1}, and the interaction between the history records H and the current control state c_i:

$$mr_i = H \otimes m_{i-1},$$
$$cr_i = H \otimes c_i,$$

where $\otimes$ denotes element-wise product.

To enable the selection of some important histories that are relevant to the current reasoning step, we introduced a Gumbel-Softmax [11]- based mechanism. The Gumbel-Softmax enables the training of discrete random variables to be trained within the neural network. Categorical probabilities are defined in terms of unnormalized log probabilities

$$g_t = \frac{exp(\frac{log(\pi_t+\gamma_t)}{\tau})}{\sum_{j=1}^{n_H} \exp(\frac{log(\pi_j)+\gamma_t}{\tau})},$$
$$\pi_i = W_e[mr_i, cr_i, H] + b_e,$$

where γ_t is the gumbel noise, τ is the temperature parameter, and n_H is the number of historical records. $W_e \in \mathbb{R}^{3d\times 1}$ is the parameter, $\pi \in \mathbb{R}^{n_H\times 1}$ is the input of Gumbel-Softmax.

In the forward-pass of the straight-through gumbel-softmax, the output vector is converted to a one-hot vector via argmax:

$$s_t = \begin{cases} 1, & i = argmax_j(g_j) \\ 0, & otherwise \end{cases}$$

The backward pass maintains the flow of continuous gradients, which allows the model to be trained end to end. Although *Reinforcement Learning*[25] is an alternatives, it is known to suffer from high variance and slow convergence [11].

Since there might be insufficient information if we keep only a single historical record, we run the Gumbel-Softmax k times, with each run pointing to a unique historical record. The overall output is a list of vectors:

$$\tilde{H}^i = [\tilde{y}_1^i, \tilde{y}_2^i, \cdots, \tilde{y}_k^i],$$

where $\tilde{H}^i \in \mathbb{R}^{k \times d}$ is the selected k historical records at the i iteration step.

We then re-weight the selected relevant histories using a soft attention mechanism. We merge the information of m_{i-1} and c_i, then utilize it to attend to the selected histories $\tilde{H}^i$. The soft attention computation is as follows:

$$\begin{aligned} mc_i &= W_{mc}[m_{i-1}, c_i] + b_{mc}, \\ mch_i &= W_{mch}(mc_i \otimes \tilde{H}^i) + b_{mch}, \\ \beta_i &= softmax(mch_i), \\ r_i &= \sum_{j=1}^{k} \beta_{i,j} \tilde{H}_j^i, \end{aligned}$$

where $W_{mc} \in \mathbb{R}^{2d \times d}$, $W_{mch} \in \mathbb{R}^{d \times 1}$ are parameters, β_i is the soft attention distribution, and r_i is the retrieved information.

Because the retrieved information is relevant to different aspects of the expertise required by the answerers, we want to build a user model according to the sub-expertise that the answerers needed, and, intuitively, we need all the retrieved information. We utilized a retrieval memory R to store the retrieved information during the entire reasoning process. At each reasoning step, we write r_i into the retrieval memory.

$$R_i = [r_1, r_2, \cdots, r_i]$$

$R_i \in \mathbb{R}^{i \times d}$ is the retrieved memory at i-th reasoning.

3.2.3 The Write Unit. The write unit is responsible for creating a new memory state m_i that holds all the global important information obtained from the interaction between the requester's question and the candidate expert's historical records. It receives the previous memory state m_{i-1}, the control state c_i, and the newly retrieved information r_i, and utilizes a concatenation operation to merge them:

$$\begin{aligned} \overline{c}_i &= W_{c3} c_i + b_{c3}, \\ \overline{m}_{i-1} &= W_{m3} m_{i-1} + b_{m3}, \\ mcr_i &= W_{mcr}[\overline{m}_{i-1}, \overline{c}_i, r_i] + b_{mcr}. \end{aligned}$$

Then, we computed a gate to make a trade-off regarding the amount of new information the newly created memory m_i considers from the previous memory m_{i-1} and the newly retrieved information r_i.

$$\begin{aligned} g_i &= \sigma(mcr_i), \\ m_i &= g_i m_{i-1} + (1 - g_i) r_i, \end{aligned}$$

where m_i is the newly created memory, and g_i is the gate value. This memory gating process prevents the model from forgetting the old memory information. If the write unit decides to keep all the old memory information, then the newly retrieved information provided by the read unit are simply ignored, and the old memory information is passed along to the next step.

3.3 Prediction Module

After p iterations, the retrieval memory has obtained all the expertise needed by a user needs to answer the requester's question. We build a user model based on the information in the retrieval memory, using a mean pooling operation.

$$\hat{r} = W_R mean(R) + b_R,$$

where $R \in \mathbb{R}^{p \times d}$ is the retrieval memory.

As input, the output prediction module takes the question representation q_{n_q}, the mean pooling of all the retrieved information $\hat{r}$, and the memory state m_p that was passed from the last RMC. We then concatenate the q_{n_q}, $\hat{r}$, and m_p.

$$f = W_f[m_p, q_{n_q}, \hat{r}] + b_f.$$

The final prediction is built by the following equation:

$$P(y = c|f; \theta_s) = \frac{exp(\theta_s^c f)}{\sum_j exp(\theta_s^j f)},$$

where θ_s^c is the weight vector of the c-th class and $j \in \{0, 1\}$.

In our work, the training objective is to minimize cross entropy of the predicted and the true label distributions, which is defined as:

$$J(H, Q; P, \hat{P}) = -\sum_{c=1}^{C} \hat{P}_c log(P_c),$$

where C is the class number, $\hat{P}$ is the one-hot representation of the ground truth label, and P is the predicted probability of the labels.

4 EXPERIMENT

Most existing works have considered the answerer who have already answered the target question as their candidate experts. While this is useful in selecting the best answer to a question, it is not helpful in directing a new question toward the potential best experts. Therefore, following the work of [20], we consider all the users in the corpus as candidate experts for a given question.

4.1 Datasets Construction

Regarding the labeled datasets, a public datasets of the competition ByteCup[5] was organized by Toutiao, but it contains no historical user question-answering records. To evaluate the effectiveness of our proposed model, we constructed two large-scale datasets deawn from Stack Overflow and Yahoo! Answer. We regard the answerer that received the "best answer" as the expert of the given question.

4.1.1 Stack Overflow. Stack Overflow is a popular question-answering service that focuses on a wide range of topics in computer programming. In our experiments, we used the December 2018 data dump [6]. First, we selected a subset of the whole dataset ranging from January 1 to December 31, 2011. To keep the tag distribution the same as that in original data collection, we chose some tags that had been used in the work [20]. We selected a subset of Stack Overflow data to meet the following two conditions: 1) questions tagged with at least one of the selected tags; 2) questions are a resolved questions that have a best answer and at least two answers. We removed stop words, leaving only the questions with

[5] https://biendata.com/competition/bytecup2016/

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

at least two words. Table 1 shows the dataset statistics. Note that only 1,717 users obtained at least 40 ($N \geq 40$) best answers. Those users constituted just 2.7% of all users, but answered 30% of all the answered questions. Like the work [20], in this work we considered users who had authored at least N ($N = 40$) best answers. With respect to the user history records, we selected 50 question-answering records of users who had last provided answers during 2011. Each records consisting of question title, question body, and answer content. The requester's questions consist of question title and question body. In total, in our Stack Overflow dataset for this experiment, we had 128,217 questions and 1,717 users, and each user had 50 question-answering records. We randomly split the whole dataset into the training, developing, and testing data with a proportion of 3:1:1, respectively.

Table 1: Data statistics. N is the number of best answers in Stack Overflow and those with the highest rating in Yahoo! Answers.

Dataset	Data type	Questions	Answerers
Stack Overflow	All	424,223	63,450
	$N \geq 40$	128,217	1,717
Yahoo! Answers	All	235,052	78,067
	$N \geq 20$	100,336	1,512

4.1.2 Yahoo! Answers. Next, we crawled a data set from Yahoo! Answers, the statistics of which are shown in table 1. We removed any questions with a length of less than two words as well as all the stop words, which left the constructed dataset with 78,067 users and 235,052 resolved questions that had at least two answers and a highest rating greater than 2. Like the Stack Overflow dataset, we only considered the experts who had answered at least 20 questions. For the user history, we selected 50 question-answering records that had been recently answered by the users, each records consisting of question title and answer content. The requester's question only include question title.. As a result, in our experiment, the Yahoo! Answers dataset contained 100,336 questions and 1,512 users, and each user had 50 question-answering records. Finally, we separated the whole dataset into training, developing, and testing data with the proportion of 3:1:1, respectively.

4.2 Candidate Expert Retrieval

Ideally, for each requester's question, all the experts in the corpus should be candidates. However, this is time consuming, since the number of experts is tremendous. To solve this issue, we retrieved a list of candidates beforehand. Some coarse, fast selection methods can be used, such as TF-IDF or BM25 [21]. In this paper, we utilized BM25 to retrieve the expert candidates.

The BM25 coarse selection method was utilized on the developing and testing sets for both *Stack Overflow* and *Yahoo! Answer*. The recall is 23.36% and 26.96%, respectively. As for the training set, we first used the requester's question to retrieve the top 30 results from the whole expert set using BM25. Then we randomly select nine experts from those results to construct the negative pairs.

4.3 Evaluation Metrics and Parameter Setting

Expert finding requires a ranked list of experts who can give high-quality answers to requester's questions, so we need high recall more than high precision. Therefore, as in [20], we utilized Success-at-N (S@N) as our evaluation metric in this paper. When the best answerer (or highest rating answerer) appeared as the top N predicted expert, the prediction is successful, and the value of S@N is the reciprocal rank of that candidate expert, otherwise the value of S@N is 0.

Table 2: Hyperparameters for our model in the experiment.

Hyperparameter	Value
reduced ELMo embedding dim.(d_c)	20
pre-trained ELMo embedding dim.	256
word embedding dim. (d)	50
BPE merge operation num.	$1k$
subword embedding dim. (d_s)	20
subword CNN kernel num.	2
subword CNN kernel size	[2, 3]
embedding dropout	0.4
Adam learning rate	0.001
model length on Stack. (p)	5
histories selected on Stack. (k)	5
model length on Yahoo. (p)	4
histories selected on Yahoo. (k)	3

In our experiments, the word embeddings were initialized using 50-dimensional, pre-trained GloVe word vectors [17], and the embeddings were fixed during training. For the Stack Overflow dataset, we set the max requester's question length to 70, the max number of histories to 50, and the max history length to 100. For the Yahoo! Answer dataset, we set the max requester's question length to 24, the max number of histories to 50, and the max history length to 100. We used Adam optimization. For other hyperparameters of our proposed model, we took those hyperparameters that achieved the best performance in the development set. Table 2 shows the final hyperparameters used.

4.4 Baselines

We compared our proposed approach against several competitive methods that utilized question-answering history information to improve the overall performance.

(1) **ZhihuRank** (Liu et al. 2015) [13] : A submodule of *zhihuRank* that contains the link analysis module and topical similarities module. Unlike the original model, our *zhihuRank* only included the topical similarities module.
(2) **LDA** (Blei et al. 2003) [2]: We used LDA to extract the topic vectors of questions and candidate experts, and a cosine similarity was computed to get the similarity.
(3) **BM25** (Robertson et al. 2009) [21]: BM25 was one of our comparative models, and it was also utilized to retrieve the candidate experts.
(4) **BiGRU-Attn**: We utilized BiGRU to encode the question and candidate expert historical records, respectively. Then the attention mechanism was applied to incorporate the correlation between the user and the question.

Table 3: Performances of different approaches on our datasets. The first part is the results of the traditional methods. The second part is the results of the existing deep learning methods. The third part is the main results of our method's variant models. RMRN is our base model and only word-level word representations.

Method	Stack Overflow					Yahoo! Answers				
	S@1	S@2	S@3	S@4	S@5	S@1	S@2	S@3	S@4	S@5
ZhihuRank–(Liu et al. 2015) [13]	0.0133	0.0206	0.0248	0.0278	0.0299	0.0144	0.0240	0.0294	0.0334	0.0360
LDA(Blei et al. 2003) [2]	0.0141	0.0212	0.0256	0.0285	0.0306	0.0168	0.0267	0.0323	0.0372	0.0407
BM25 (Robertson et al. 2009) [21]	0.0409	0.0537	0.0613	0.0659	0.0689	0.0484	0.0645	0.0724	0.0777	0.0813
BiGRU-ATTN	0.0675	0.0831	0.0909	0.0959	0.0996	0.0811	0.1011	0.1103	0.1155	0.1196
D-Attn (Seo et al. 2017) [22]	0.0694	0.0851	0.0929	0.0980	0.1014	0.0821	0.1017	0.1113	0.1171	0.1210
ANR (Chin et al. 2018) [3]	0.0704	0.0878	0.0965	0.1019	0.1052	0.0827	0.1039	0.1132	0.1187	0.1226
N2NMem (Sukhbaatar et al. 2015) [23]	0.0710	0.0884	0.0966	0.1022	0.1056	0.0839	0.1039	0.1135	0.1193	0.1229
MAC (Hudson and Manning 2018) [9]	0.0719	0.0884	0.0967	0.1020	0.1054	0.0842	0.1040	0.1140	0.1199	0.1233
RMRN	0.0728	0.0885	0.0965	0.1019	0.1055	0.0867	0.1072	0.1168	0.1226	0.1267
RMRN+Sub	0.0746	0.0915	0.0998	0.1050	0.1083	0.0885	0.11.05	0.1214	0.1254	0.1288
RMRN+ELMo	0.0752	0.0924	0.1007	0.1060	0.1093	0.0891	0.1107	0.1215	0.1260	0.1295
RMRN+Enhanced	**0.0759**	**0.0933**	**0.1017**	**0.1071**	**0.1104**	**0.0903**	**0.1122**	**0.1233**	**0.1278**	**0.1314**

(5) **D-Attn** (Seo et al. 2017) [22]: The Dual Attention-based Model (*D-Attn*) was proposed to learn the user and item representations based on their reviews.
(6) **ANR** (Chin et al. 2018) [3]: *ANR*) was proposed to extract an aspect-based representation for both the user and item based on reviews.
(7) **N2NMem** (Sukhbaatar et al. 2015) [23]: N2NMem is an end-to-end memory network that can be trained via back propagation. N2NMem has a reasoning ability.
(8) **MAC** (Hudson and Manning 2018) [9]: A careful design memory framework for visual question answering tasks.

RMRN is our based model and only word-level word representations. We conducted several variations of our model based on the different word representations. The word representations of RMRN+Sub, RMRN+ELMo and RMRN+Enhanced are a concatenation of word and subword-level embeddings, a concatenation of word and character-level embeddings, and a concatenation of character, subword, and word-level embeddings, respectively.

4.5 Results and Discussion

Table 3 shows a comparison of the results obtained by our proposed method and existing methods on the constructed evaluation datasets. As shown in the table, our proposed model (RMRN) performed better than the other methods in all metrics.

The reasoning mechanism methods (RMRN, MAC, and N2NMem) performed better performance than the other previous methods (ANR, D-Attn, zhihuRank-, etc.), which indicates that iterative reasoning facilitates an understanding of the implicit relevance of both the requester's question and the candidate experts' expertise. Our proposed method RMRN outperformed MAC and N2NMem, which are both competitive memory-based reasoning methods. We consider the reasons for this to be as follow: 1) MAC and N2NMem are designed specifically for question-answering task, and the predicted answer only depends on the last reasoning state. However, in expert finding, all of the reasoning states should be taken into consideration, because every reasoning state measures the expertise of the candidate expert regarding the current aspect of the requester's question. We wrote all the retrieved information into a retrieval memory, on which we based our construction of a user model. 2) The performance may be negatively impacted, when utilizing a complete set of historical records. Unlike MAC and N2NMem, our RMRN uses only the portion of the historical records that are relevant to the current aspect of requester's question at each reasoning process. Therefore, the careful design of the RMRN can effectively handle the expert finding task.

The neural-network-based methods performed much better than the traditional methods (zhihuRank, LDA, and BM25). This may be true for two reasons. First, word embedding offers a more expressive and efficient representation by maintaining the contextual similarity of words. Second, BiGRU and CNN can capture semantic information from complicated sentences and promote further semantic interaction between a requester's question and the user's historical records.

Multi-grain word embedding helps to capture different levels of word representation and thus improves the performance. Compared to our RMRN, which utilized only word embedding, the performance of RMRN+Subword, RMRN+ELMo, and RMRN-Enhanced are better. What's more, RMRN-Enhanced, which includes character-, subword-, and word-level representations, achieved the best result, because character- and subword-level word representations capture morphological information on a different level, and word-level word representations contain the semantic information of words, and their combination enhances the RMRN performance.

To further understand the contribution of the model components, we also conducted some ablation studies on our model.

Effect of Control Unit: To understand the contribution to our model made by the control unit, which is defined in section 3.2.1, we performed several ablation studies in the control unit, the results of which are shown in the first part of Table 4. The term *que+mem as control* replaces the control state c_i as the concatenation of question q_i and memory state m_i, as with the query state defined in [23]. The *shared weights for que* utilized shared parameters to linearly transform the question q_{n_H} at each reasoning process. Compared to

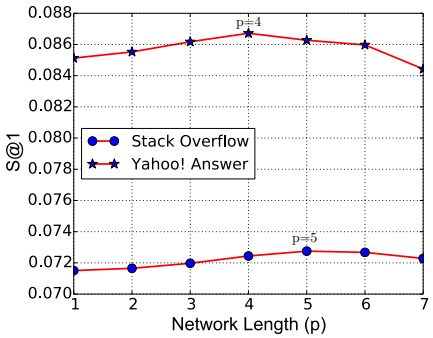

Figure 3: Performance with different numbers of RMC.

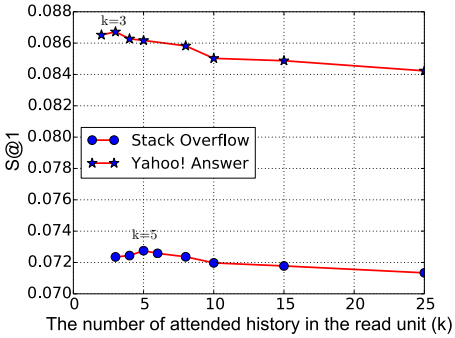

Figure 4: Performance with different numbers of selected historical records in the read unit.

Table 4: S@1 comparison with several ablation models on the control unit, write unit, and output module.

	Ablation Model	Stack Overflow	Yahoo! Answer
Control Unit	que+mem as control	0.0720	0.0855
	shared weights for que	0.0717	0.0860
Read Unit	soft attention for read	0.0718	0.0850
Write Unit	ignored previous mem	0.0721	0.0861
	gate computed in MAC	0.0722	0.0858
Output Module	predict on que+mem	0.0715	0.0857
	predict on que+avgAllRetri	0.0713	0.0863
	predict on que+mem+Retri	0.0724	0.0864
Standard	**RMRN**	**0.0728**	**0.0867**

our RMRN with *que+mem as control* and *shared weights for que*, the RMRN achieved a better result. This demonstrates the helpfulness of having the control unit focus on different parts of a question, and that different transformation parameters in the question representation can help to readily concern a question's different parts.

Effect of Read Unit: The read unit utilizes the Gumbel-Softmax mechanism to select historical records that are relevant to the current aspect of requester's question. We compared our standard model with the model *soft attention for read* that used a soft attention mechanism to attend to all the candidate-expert historical records, the results of which are shown in the second part of Table 4. The performance of our standard model, which selects only some historical records that are relevant to the requester's question, is much better than that using soft attention, which attends to all the candidate-expert histories.

Effect of Write Unit: The standard RMRN write unit uses a gate to combine the previous memory state m_{i-1} with the newly retrieved information r_i, and the gate computed based on m_{i-1}, r_i, and the control state c_i. We conducted some ablations on the write unit, and the results of which are shown in the third part of Table 4. The *ignored previous mem* ignores the previous memory state, and directly assigns the r_i to m_i. A comparison of *ignored previous mem* with our standard RMRN model, the result shows that the information obtained in the previous reasoning state is helpful. The *gate computed in MAC* utilized only the control state c_i to compute the gate value, like MAC. The result shows that the calculation of the gate by the variant is slightly worse than that by our standard model.

Effect of Output Unit: The standard RMRN output module makes a final prediction based on the concatenation of the question feature q_{n_q}, memory feature m_p, and the mean pooling of all the retrieved information $\hat{r}$. We conducted several ablations on the output module to explore the contribution of various information, the results of which are shown in fourth part of Table 4. As inputs to the classifier, the *predict on que+mem*, *predict on que+mem+Retri*, and *predict on que+avgAllRetri* models utilize a concatenation of q_{n_q} and m_p, a concatenation of q_{n_q}, m_p, and r_p, and a concatenation of q_{n_q} and $\hat{r}$, respectively. Compared with the *predict on que+mem* model, the *predict on que+mem+Retri* which utilizes the last retrieved information r_p, achieved an improvement in S@1. In addition, our standard RMRN model, which utilized the $\hat{r}$, achieved the best performance. Therefore, we can conclude that the retrieved information in the read unit is important to achieving improved performance. The performance of the *predict on que+avgAllRetri*, which ignores the m_p, was lower than that of our standard RMRN model. The m_p makes a trade-off between the importance of previous and new information, and $\hat{r}$ utilizes all the retrieved information, since all information plays an important role in improving the performance.

4.6 Parameter Sensitivity

In this section, we analyze the influence of the critical parameters contained in our model. Specifically, we are concerned about the impact of the model length and the number of histories selected in the read unit.

First, we tested the performance using RMRN's of various lengths ranging from one to seven, the results of which are shown in Figure 3. Our model achieved the best performances on the Yahoo! Answer and Stack Overflow datasets when the model lengths were four and five, respectively. Since increasing the length of the network increases the complexity and number of parameters in the model, the results are not significantly improved by increasing the model length. The model depth indicates the number of inferences made, and the more complex is the dataset, the more inferences are needed.

Therefore, the complexity of the Stack Overflow dataset makes its job more difficult than that of Yahoo! Answer.

Second, we evaluated our model with selections of different numbers of candidate-expert historical records, the results of which are shown in Figure 4. Our model achieved the best performance on the Yahoo! Answer and Stack Overflow datasets when using three and five candidate-expert historical records, respectively. If we increase the number of selected historical records, the performances of the Stack Overflow and Yahoo! Answer datasets drop significantly. This may be due to the introduction of some noisy historical records when the value of k becomes too large. Although the maximum number of historical records is 50 in the Stack Overflow and Yahoo! Answer datasets, the optimal numbers of historical records selected were small, i.e., three for the Yahoo Answer dataset and five for the Stack Overflow dataset. It may be that the historical records selected are only relevant to one aspect of the requester's question at each stage of the reasoning process.

5 CONCLUSION

In this work, we proposed a novel recurrent memory reasoning network that performs expert finding in community question answering by utilizing a reasoning mechanism to explore the implicit relevance between a requester's question and candidate experts' historical records. To alleviate the OOV problem in CQA, we explored the use of a multi-grained word representation, including character-, subword-, and word-level representations. To select relevant historical records from candidate-expert answering histories, we introduced a Gumbel-Softmax-based mechanism in the read unit. In addition, we built two large-scale datasets drawn from Stack Overflow and Yahoo! Answer. Comparing with the baseline models, the performance of our proposed method is much better.

ACKNOWLEDGMENTS

The authors wish to thank the reviewers for their helpful comments. This work was partially funded by China National Key R&D Program (No. 2018YFB1005104, 2018YFC0831105, 2017YFB1002104), National Natural Science Foundation of China (No. 61976056, 61532011, 61751201), Shanghai Municipal Science and Technology Major Project (No.2018SHZDZX01), Science and Technology Commission of Shanghai Municipality Grant (No.18DZ1201000, 16JC1420401, 17JC1420200).

REFERENCES

[1] Fawaz Alarfaj, Udo Kruschwitz, David Hunter, and Chris Fox. 2012. Finding the right supervisor: expert-finding in a university domain. In *Proceedings of the 2012 Conference of the NAACL: human language technologies: student research workshop*. Association for Computational Linguistics, 1–6.

[2] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research* 3, Jan (2003), 993–1022.

[3] Jin Yao Chin, Kaiqi Zhao, Shafiq Joty, and Gao Cong. 2018. ANR: Aspect-based Neural Recommender. In *Proceedings of the 27th ACM international CIKM*. ACM, 147–156.

[4] Kyunghyun Cho, Bart Merrienboer, Caglar Gulcehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In *EMNLP*.

[5] Gideon Dror, Yehuda Koren, Yoelle Maarek, and Idan Szpektor. 2011. I want to answer; who has a question?: Yahoo! answers recommender system. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1109–1117.

[6] Lan Du, Wray Buntine, and Huidong Jin. 2010. A segmented topic model based on the two-parameter Poisson-Dirichlet process. *Machine learning* 81, 1 (2010), 5–19.

[7] Jinwen Guo, Shengliang Xu, Shenghua Bao, and Yong Yu. 2008. Tapping on the potential of q&a community by recommending answer providers. In *Proceedings of the 17th ACM conference on Information and knowledge management*. ACM, 921–930.

[8] Chaoran Huang, Lina Yao, Xianzhi Wang, Boualem Benatallah, Shuai Zhang, and Manqing Dong. 2018. Expert recommendation via tensor factorization with regularizing hierarchical topical relationships. In *International Conference on Service-Oriented Computing*. Springer, 373–387.

[9] Drew A Hudson and Christopher D Manning. 2018. Compositional attention networks for machine reasoning. *arXiv preprint arXiv:1803.03067* (2018).

[10] Adrian Huna, Ivan Srba, and Maria Bielikova. 2016. Exploiting content quality and question difficulty in CQA reputation systems. In *International Conference and School on Network Science*. Springer, 68–81.

[11] Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144* (2016).

[12] Baichuan Li and Irwin King. 2010. Routing questions to appropriate answerers in community question answering services. In *Proceedings of the 19th ACM international CIKM*. ACM, 1585–1588.

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

[14] Zhu Liu, Kan Li, and Dacheng Qu. 2017. Knowledge Graph Based Question Routing for Community Question Answering. In *International Conference on Neural Information Processing*. Springer, 721–730.

[15] David Mimno and Andrew McCallum. 2007. Expertise modeling for matching papers with reviewers. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 500–509.

[16] Aditya Pal, Rosta Farzan, Joseph A Konstan, and Robert E Kraut. 2011. Early detection of potential experts in question answering communities. In *International Conference on User Modeling, Adaptation, and Personalization*. Springer, 231–242.

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

[18] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the NAACL: Human Language Technologies, Volume 1 (Long Papers)*, Vol. 1. 2227–2237.

[19] Yujie Qian, Jie Tang, and Kan Wu. 2018. Weakly Learning to Match Experts in Online Community.. In *IJCAI*. 3841–3847.

[20] Fatemeh Riahi, Zainab Zolaktaf, Mahdi Shafiei, and Evangelos Milios. 2012. Finding expert users in community question answering. In *Proceedings of the 21st International Conference on World Wide Web*. ACM, 791–798.

[21] Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends® in Information Retrieval* 3, 4 (2009), 333–389.

[22] Sungyong Seo, Jing Huang, Hao Yang, and Yan Liu. 2017. Interpretable convolutional neural networks with dual local and global attention for review rating prediction. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 297–305.

[23] Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. 2015. End-to-end memory networks. In *Advances in neural information processing systems*. 2440–2448.

[24] Qiongjie Tian, Peng Zhang, and Baoxin Li. 2013. Towards Predicting the Best Answers in Community-based Question-Answering Services.. In *ICWSM*.

[25] Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning* 8, 3-4 (1992), 229–256.

[26] Fei Xu, Zongcheng Ji, and Bin Wang. 2012. Dual role model for question recommendation in community question answering. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*. ACM, 771–780.

[27] Baoguo Yang and Suresh Manandhar. 2014. Tag-based expert recommendation in community question answering. In *Advances in Social Networks Analysis and Mining (ASONAM), 2014 IEEE/ACM International Conference on*. IEEE, 960–963.

[28] 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 22nd ACM international conference on Information & Knowledge Management*. ACM, 99–108.

[29] Zhou Zhao, Qifan Yang, Deng Cai, Xiaofei He, and Yueting Zhuang. 2016. Expert Finding for Community-Based Question Answering via Ranking Metric Network Learning.. In *IJCAI*. 3000–3006.

[30] Zhou Zhao, Lijun Zhang, Xiaofei He, and Wilfred Ng. 2015. Expert finding for question answering via graph regularized matrix completion. *IEEE Transactions on Knowledge and Data Engineering* 27, 4 (2015), 993–1004.

[31] Hengshu Zhu, Enhong Chen, Hui Xiong, Huanhuan Cao, and Jilei Tian. 2014. Ranking user authority with relevant knowledge categories for expert finding. *World Wide Web* 17, 5 (2014), 1081–1107.