Preference Relationship-Based CrossCMN Scheme for Answer Ranking in Community QA

Qing Chen[1,2], Jianji Wang[1,*], Xuguang Lan[1], Nanning Zheng[1,*]
[1]*National Engineering Laboratory for Visual Information Processing and Applications*
Institute of Artificial Intelligence and Robotics, Xi'an Jiaotong University, Xi'an, China
[2]*School of Software Engineering, Xi'an Jiaotong University, Xi'an, China*
small_persimmon@stu.xjtu.edu.cn;{wangjianji,xglan,nnzheng}@mail.xjtu.edu.cn

***Abstract*—Community question answering (CQA) systems aim to provide users with high-quality answers. Nevertheless, unreliable answers are often returned to users in CQA systems, and the phenomenon causes that users have to browse multiple answers to find the best one. To improve such problem, we design a novel scheme, named PW-CrossCMN. The scheme ranks the candidate answers by pair-wise approach based on numerous historical documents. In the scheme, we apply the preference relationship into deep learning framework. Specifically, the scheme consists of two phases. In phase 1, the scheme extracts the features via automated feature engineering to construct the preference vectors and then divides the vectors into balanced positive and negative training samples based on the preference relationship. In phase 2, we build the CrossCMN model, which implements the multi-network parallel convolution and the cross forward propagation of full-connected layers, to achieve training and prediction tasks. Moreover, the multi-layer perception (MLP) is introduced to extract combination features in the prediction module. We perform extensive experiments on two typical datasets, and the results show that our scheme has more excellent performance in answer ranking task compared with several state-of-the-art baselines. In addition, we have released the relevant codes.**

***Index Terms*—answer ranking, preference relationship, parallel convolution network, community question answer**

I. Introduction

Community question answering (CQA) systems have become an important platform to provide solutions in our lives. Users solve daily questions by acquiring online knowledge in two ways. The first is to propose questions and then wait or invite the professionals/non-professionals to respond, and the other is to search historical documents, in which numerous questions have been solved, to find the needed solutions. Simultaneously, many effective CQA systems provide services for users. Mainstream CQA systems can be mainly divided into the general and the vertical types [7]. The general CQA systems include all fields of people's lives, such as Quora[1] and Wiki[2]. The vertical CQA systems, such as Stack Overflow[3] and HealthTap[4], generally cater to professional fields to provide users with more reliable and high-quality answers.

To improve the effectiveness and user satisfaction of the CQA system, important results have been achieved in the academia and practical applications via deep learning and natural language processing technologies. In the field of Natural Answer Generation (NAG), knowledge base (KB) retrieval and sequence model (seq2seq) were combined to generate more fluent answers [1]. By making full use of valuable information in the corpus with noise and uneven quality, Liu *et al.* improved the accuracy of simple and complex problems by 6.8% and 8.7%, respectively [2]. In the aspect of question routing, Ji and Wang routed new questions to the appropriate respondents by the ranking learning method [3]. Inspired by Ji and Wang's work, Cheng *et al.* provided the TTM topic model and multi-objective optimization method for question routing in CQA [4]. Macavaney *et al.* proposed a new method based on CNN to represent the questions in the field of Complex Answer Retrieval (CAR) [5]. Kratzwald *et al.* adopted the adaptive scheme to determine the optimal documents count in Adaptive Document Retrieval [6].

Nevertheless, various problems still need to be solved in CQA. One of the most influential problems is the phenomenon of *question starvation* [7], for which no recognized solution has yet been found. According to the report by Calefato *et al.*, unsolved questions in Stack Overflow have reached over 7 million questions (about 50%), of which 70% have been answered [8]. Moreover, for Wikianswers[5], only approximately 25% of the 1.16 million questions on its question and answer (Q&A) site are answered. Researchers have discussed this problem from different views. For example, the NAG method and question routing mentioned above help solve this problem by generating new answers or routing unsolved questions to the proper answerers. However, NAG cannot guarantee the correctness of the answers, and question routing method cannot provide instant answers or even cannot guarantee these questions are finally answered.

In fact, modern CQA systems have accumulated massive historical Q&A documents, based on which numerous unsolved questions can be answered by searching for similar questions. In this paper, a novel scheme, named PW-CrossCMN, is proposed to optimize the ranking of candidate answers depending on the historical documents. This scheme can improve the *question starvation* phenomenon effectively by searching for similar questions in historical documents and

* The corresponding author.
[1]https://www.quora.com/
[2]https://www.wiki.com/
[3]https://stackoverflow.com/
[4]https://www.healthtap.com/
[5]https://www.answers.com/

2374-8486/19/$31.00 ©2019 IEEE
DOI 10.1109/ICDM.2019.00018

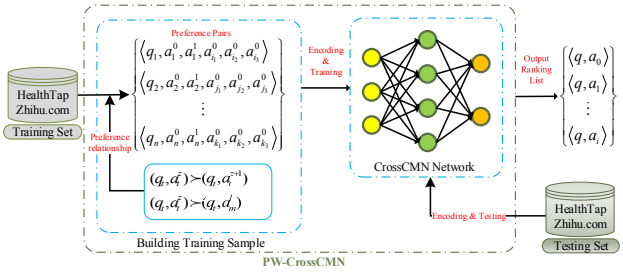


Fig. 1: Framework of our scheme. Firstly, construct training samples based on preference relationship and Q&A texts. Then build and train CrossCMN network model. Finally, the test set is predicted and sorted by CrossCMN.

return the reliable answers for the user. The workflow of our scheme is shown in Fig. 1.

Two typical datasets, Zhihu and Healthtap, are used in our experiments to measure the effectiveness of our scheme. Zhihu is a general CQA system, and Healthtap is a vertical CQA system in the medical field.

The main contributions of this paper are as follows:

- **Propose a novel scheme named PW-CrossCMN to rank candidate answers**: The preference vectors are constructed with the improved preference relationship by the pair-wire approach, and then the vectors are trained in the proposed CrossCMN model. Here, we first train the preference vectors by parallel convolution network.
- **Construct a new preference relationship of training samples**: The preference relationship is considered for all candidate answers of a question. Then, according to the preference relationship, the dataset is divided into balanced positive and negative training samples by automated feature engineering.
- **Build the CrossCMN model to realize training and prediction**: We design a deep learning model which is more suitable for answer ranking in CQA system. In our model, the parallel convolutional neural network is used to extract the local information. In the prediction module, the multi-layer perceptron is used to extract the combination feature.
- **Achieve the best performance and release the codes**: The experimental results show that our method outperforms state-of-the-art baselines. The average improvement rate from P@1 to P@5 on two different datasets is 6.29. The relevant codes[6] of our scheme and its simplified versions are released.

The rest of this paper is structured as follows. Section II introduces the related work of answer ranking in CQA. The PW-CrossCMN model is proposed in Section III. Then the experimental results and analysis are given in Section IV. Finally, this work is concluded in Section V.

[6]https://github.com/small-persimmon/Answer_Ranking

II. RELATED WORK

A. Learning To Rank

Learning-to-Rank (LTR) is a widely used technology in the field of information retrieval. LTR includes various classical machine learning models such as AdaBoost, ListNet, and RankSVM. Originally, LTR was used to solve the ranking of web pages on the Internet, and now many approaches of LRT have been applied in CQA systems. Dalip *et al.* and Shah *et al.* successfully applied LTR technology to evaluate and predict the performance of answer ranking in CQA [9], [10].

LTR has three main document ranking approaches, namely, point-wise, pair-wise and list-wise. The point-wise approach mainly deals with single documents, which are scored according to the classification or regression function learned from training data. Then the document scores are taken as the standard of documents ranking [10]. The pair-wise approach considers the context of documents and takes the comparison results as the standard to rank answers [7], [11]. Different from the first two ranking approaches, the list-wise approach does not convert the ranking problem into classification or regression form, but directly optimizes the ranking results of documents according to the evaluation criteria [12]. In our scheme, the pair-wise strategy is used.

B. Preference Relationship

Constructing the preference relationship among objects is highly important in pair-wise approaches of LTR. Carvalho *et al.* significantly improved the performance of their method by studying the effects of outlying pairs and employing meta-learning thinking [16]. In the context of pair-wise ranking, Attentive Pooling (AP) enabled the pooling layer to be aware and it made the information from the two input items can directly influence the computation of each other's representations [19]. Wu *et al.* identified that user search intentions can be reflected by user behavior. Then, a new intention-based language model was proposed through these intention signals based on the idea [25]. Inspired by user research and observation, Nie *et al.* constructed positive, neutral, and negative training samples based on a novel preference relationship, and it considerably simplified the labeling process [7].

C. Deep Learning in CQA

Applications of deep learning have provided many important results in CQA answer ranking. Lai *et al.* elaborated three deep learning architectures in answer ranking, namely, Siamese Architecture, Attention Architecture, and Compare-Aggregate Architecture [17]. Tan *et al.* proposed the QA-LSTM/CNN model by combining bidirectional long short-term memory (bi-LSTM) with CNN, and they used bi-LSTM to construct Q&A text embedding [18]. Santos *et al.* constructed an AP-BILSTM model to realize the feature weighting of the answers and questions and effectively improved the model performance [19]. Bian *et al.* implemented the Dynamic-Clip Attention model that aimed to filter the noise in the attention matrix and mined the semantic relevance of

word-level vectors [20]. Afterwards, on the basis of Compare-Aggregate Architecture and according to the point-wise approaches of LTR, Shen *et al.*, Tran *et al.* and Tay *et al.* proposed the IWAN, CARNN and MCAN models respectively. These models all introduced the attention mechanism and LSTM, where CARNN was an improvement of IWAN [21]–[23].

III. PW-CrossCMN SCHEME

We propose a novel scheme, named PW-CrossCMN, to optimize the answer ranking of CQA systems, and we elaborate our scheme in three aspects as follows:

A. Construct Preference Relationship

For a question, the order of answers is usually consistent with the number of the "votes" in CQA. This means that answers with more votes gain a higher ranking. Compared with the neutral preference relationship of Nie *et al.* [7], different degrees of preference are observed between answers according to the number of votes in this study. Therefore, we construct the preference vectors by the new preference relationship. The details of preference relationship are as follows:

- **Definition 1:** For a question, an answer has a higher preference than the one behind it according to the user votes. Generally, for a question, the votes of each answer are different. According to the number of votes, the first answer ranks higher than the second answer, and thus gains preference. Similarly, the question prefers the second answer than the third answer. The preference relationship can thus be constructed between answers according to the votes.
- **Definition 2:** A question prefers its answers over those of others. In general, the content of the answers is related to the question, and users can judge their required answers according to the content. The experimental results of Nie *et al.* show that users are more inclined to the answer of the question itself. The question-specific answers are more suitable when the questions are often complex and sophisticated. According to this definition, the preference relationship is constructed between answers of similar questions to the original.

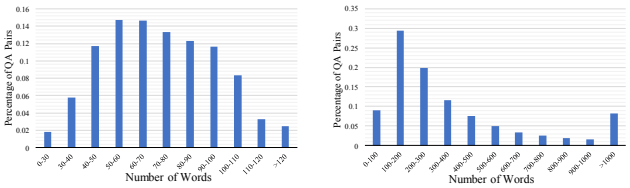
Based on the two definitions, we define a_i^j to be the jth candidate answer of q_i, and a_i^{j-1} is the previous answer to a_i^j. a_i^0 is the first and best answer to q_i. The preference levels of $a_i^0, a_i^1, \cdots a_i^n$ decreases gradually according to the ranking. a_k^t is the tth answer to q_k, which q_k is a similar question of q_i. We derive the following equations:

$$\begin{cases} (q_i, a_i^{j-1}) \succ (q_i, a_i^j), j \neq 0 \\ (q_i, a_i^j) \quad \succ (q_i, a_k^t), i \neq k, \end{cases} \tag{1}$$

where $\succ$ represents the preference relationship.

We set $x = x^{(1)} - x^{(2)}$, where $x^{(1)}$ and $x^{(2)}$ are the d-dimensional feature vectors and y is the result of preference relationship x, which satisfies the following relations:

$$y = \begin{cases} +1, \ x^{(1)} \succ x^{(2)} \\ -1, \ x^{(2)} \succ x^{(1)}. \end{cases} \tag{2}$$



(a) Words in HealthTap (b) Words in Zhihu

Fig. 2: (a) and (b) are the word distribution of Q&A texts in HealthTap and Zhihu datasets, respectively.

In view of this, we build the preference training set with preferable labels $P = \{(x_i, y_i)\}_{i=1}^{N}$.

B. Feature Extraction

Before building the preference relationship, the topic and sent2Vec features are extracted by automated feature engineering. Simultaneously, the user information as the feature needs to be drawn to construct the final feature vector.

Topic Feature: Text topic model is mainly used to mine the topics and effectively classify documents. To build the model, we adopt the Bayesian-based Latent Dirichlet Allocation (LDA), which generates the topic distribution of each document and the word distribution of each topic through the perplexity metric. The LDA topic models of Zhihu and HealthTap datasets are built and the t-dimensional topic vectors are constructed as T.

Sent2Vec Feature: The quality of topic feature extracted by the LAD model is descended when the text is short [4]. The words distribution of Q&A text in HealTap and Zhihu datasets are shown in Figs. 2(a) and 2(b), respectively. Clearly, most of the Q&A text is short in length. To obtain richer feature information, we thus construct the sentence-level features based on the thought of Doc2Vec [13]. The s-dimensional sentence vector of Zhihu and HealthTap datasets are generated as S, respectively.

UserInfo Feature: We extract the number of answers, question labels and other information related to users as UserInfo feature. To concatenate the UserInfo feature with the topic and the sent2Vec features, the corresponding information of UserInfo feature is converted into digital format. To make it easier to train the model, the UersInfo features to u-dimensional vector is normalized as U.

F is the ultimate feature vector denoted by Eq. (3), as follows:

$$F = Concat(Concat(T,\ S),\ U), \tag{3}$$

where T is the topic vectors, $T \in \mathbb{R}^t$, S is the sentence vectors, $S \in \mathbb{R}^s$ and U is the user feature vectors, $U \in \mathbb{R}^u$. F consists of three types of feature (T, S, and U). $F \in \mathbb{R}^f$ where $f = t + s + u$. $Concat(\cdot)$ represents the concatenate relationship between different types of feature.

C. Design PW-CrossCMN Scheme

The training sample is a preference vector generated according to the preference relationship. Due to the vector containing very little context information, the CrossCMN model

is proposed to replace timing models that rely heavily on context information, such as RNN and LSTM. Our model uses parallel convolutional networks to extract the local information of the preference vectors. We draw into the MLP module that is processed by batch normalization to extract the combined features of the preference vectors.

PW-CrossCMN consists of two phases. The main task is to generate the preference vector in phase 1. The CrossCMN model is built in phase 2. The scheme is described in detail below:

Phase 1: Construct the Preference Vector

Input Layer: After preprocessing the two datasets, the Q&A text is adapted into Eq. (4). q_n represents the nth question in the Q&A dataset, and $a_n^0, a_n^1, \cdots, a_n^i$ represent candidate answers for q_n. The number of candidate answers for each question is set in the experiment, and the candidate answers have been ranked according to the number of votes and the preference relationship. The question q_n and each of its answers a_n^i are input as the format (q_n, a_n^i).

$$\left\{ \begin{array}{c} (q_1, a_1^0) \\ (q_1, a_1^1) \\ \vdots \\ (q_1, a_1^i) \end{array} \right\}, \left\{ \begin{array}{c} (q_2, a_2^0) \\ (q_2, a_2^1) \\ \vdots \\ (q_2, a_2^i) \end{array} \right\}, \cdots, \left\{ \begin{array}{c} (q_n, a_n^0) \\ (q_n, a_n^1) \\ \vdots \\ (q_n, a_n^i) \end{array} \right\} \quad (4)$$

Features Extraction Module: Subsection B of Section III shows that our preference vector consists of the Topic, Sent2Vec and UserInfo features. Based on (q_n, a_n^i), two types of features are extracted. One is the 50-dimensional T_i generated by the topic model of LDA. The other is to generate 50-dimensional S_i according to the Sent2Vec embedding model. The word vector of Sent2Vec is generated by the word2vec [26]. Then, T_i and S_i are concatenated to generate E_i, as follows:

$$E_i{=}Concat(T_i, S_i), \quad (5)$$

where $E_i \in \mathbb{R}^e$ and $e = t + s$. We extract the user information corresponding to each answer and convert it into the feature vector as U_i. We then concatenate E_i and U_i to construct the preference vectors F_i, as follows:

$$F_i{=}Concat(E_i, U_i), \quad (6)$$

where $F_i \in \mathbb{R}^f$.

Generate the Preference Vector: Similarly, the feature vector F_j of (q_n, a_n^j) is generated in the Features Extraction Module in Fig. 3. The final preference vector is constructed by the preference relationship between F_i with F_j, as follows:

$$V = F_i - F_j, \quad (7)$$

where $V \in \mathbb{R}^f$. According to the preference relationships, if $F^{(i)} \succ F^{(j)}$, then the label of V is $+1$. Otherwise, if $F^{(i)} \prec F^{(j)}$, the label is -1.

Phase 2: Build the Cross_CMN Model

Data Conversion: The preference vectors generated by Phase 1 is 1-D spatial data. Thus, data formats must be converted as inputs of the convolution module in Phase 2. By expanding the data dimensions, the 1-D spatial data is transformed into 3-D spatial data. Similar to image processing, the data corresponds to the image height, width, and channels of the image. It is worth emphasizing that the height and channels are fixed at 1, and the width is the dimension of the preference vectors. Data conversion is equivalent to making the convolution networks indirectly implement the 1-D convolution of the preference vector.

Parallel Convolutional Neural Network Module: The Parallel Convolutional Module consists of four parallel convolution networks, namely, CNN-1, CNN-2, CNN-3, and CNN-4. The overall structure of these convolution networks is composed of three convolution blocks, each followed by a Max-pooling layer. The details of the convolution networks setting are shown in Fig. 3. To achieve better convolution consequent by obtaining different convolution fields, two different sizes of (1×3) and (1×4) convolution kernels are set up for different convolutional networks. We set the kernels of the first convolution layer is (1×1).

The convolution module is executed in parallel by multiple convolution networks with multiple convolution layers. To ensure that the scaling of the input variables is constant in each layer, and avoid their explosion or diffusion in the last layer, the method proposed by He *et al.* is used to initialize the weights [27]. The distribution is as follows:

$$W \sim N[0, \sqrt{\frac{2}{\hat{n}_i}}], \quad (8)$$

$$\hat{n}_i{=}h_i \cdot w_i \cdot d_i, \quad (9)$$

where h_i and w_i denote the height and width of the convolution kernel in each convolutional layer, respectively, and d_i denotes the number of convolution kernels. To increase the robustness of the training process of the model and ensure that the gradient does not explode or disappear during the training process, SELU [28] is used as the activation function, which is described by Eq. (10):

$$\text{selu}(x) = \lambda \left\{ \begin{array}{ll} x, & \text{if } x > 0 \\ \alpha(e^x - \alpha), & \text{if } x \leq 0, \end{array} \right. \quad (10)$$

where x is the input of the activation function, and the super parameters α and λ are fixed values.

Cross Full-Connected Module: To allow CNN to better extract the local information, cross forward propagation is adopted between different convolution networks in the full-connected layer to enhance the high dimensional output. In detail, the output of CNN-2 with convolution kernel (1×4) is superimposed on the output of CNN-1 with convolution kernel (1×3) in the full-connected layer. Meanwhile, the output of CNN-4 with convolution kernel (1×3) is superimposed on the output of CNN-3 with convolution kernel (1×4) in the full-connected layer. Referring to Cross Full Connection Module in Fig. 4, the detailed operations are as follows:

$$G_i = g_i + h_i, H_j = h_j + g_j \quad (11)$$

$$P_i = G_i + H_i, K_j = H_j + G_j \quad (12)$$

$$W_i = P_i + K_i, W_j = K_j + P_j. \quad (13)$$

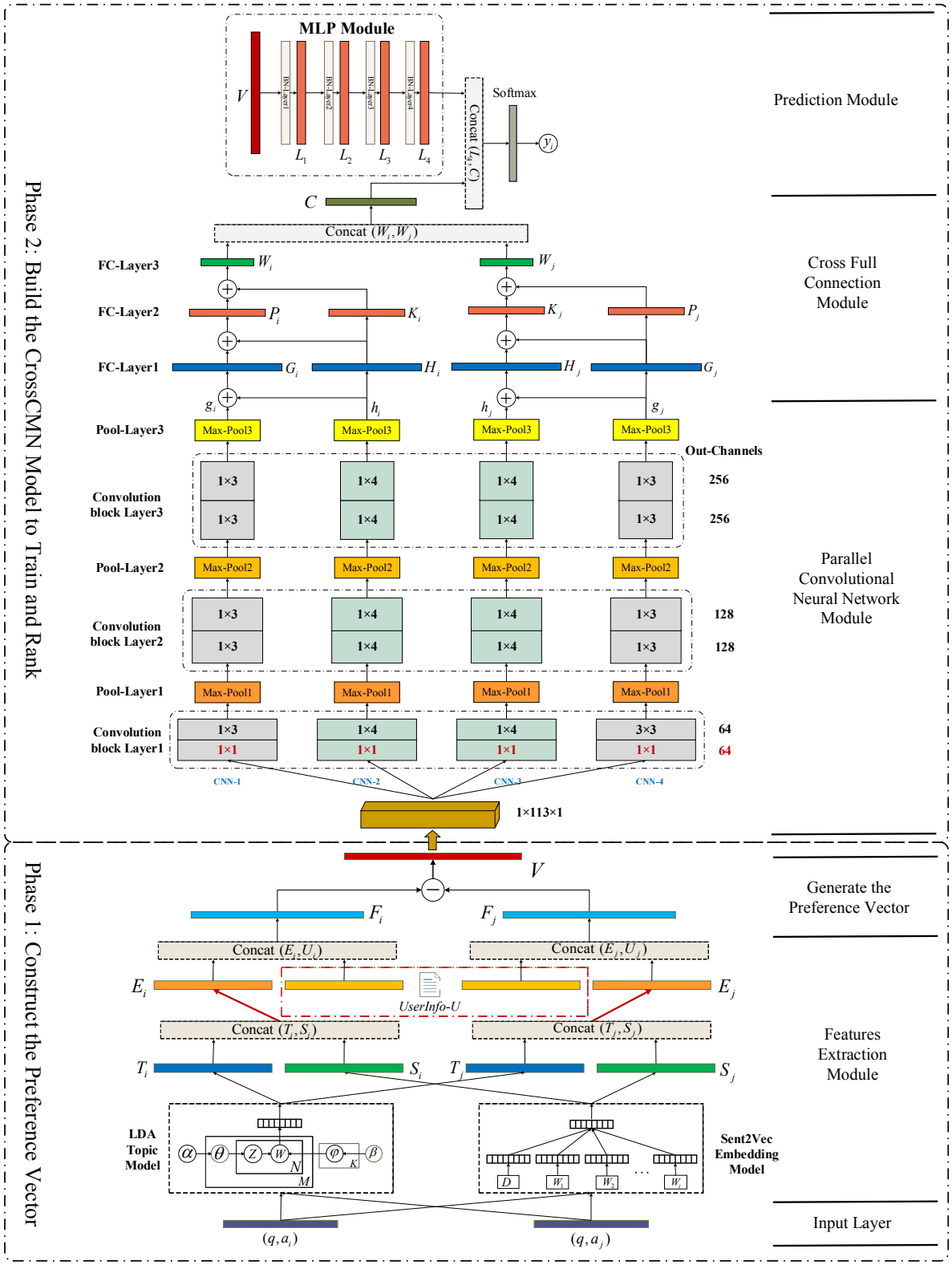


Fig. 3: The structure of PW-CrossCMN. Phase 1: Extract two main features by automated feature engineering and draw into UserInfo information to construct the preference vector according to the preference relationship. Phase 2: The characteristics of CNN and MLP are used to build the CrossCMN network to extract the local information and combination information of the training samples.

We concatenate W_i with W_j to generate C:

$$C{=}Concat(W_i, W_j). \tag{14}$$

In Fig. 3, the elements are as follows: $C \in \mathbb{R}$ is the final output of the cross full-connected module, g_i, h_i, h_j and g_j are the 1-D vectors that reshaped after the last Max-pooling layer, G_i, H_i, H_j and G_j are the outputs of FC-Layer1, P_i, K_i, K_j and P_j are the outputs of FC-Layer2, W_i and W_j are the outputs of FC-Layer3. In addition, to avoid over-fitting and improve the generalization ability, the dropout mechanism is introduced and the weights are regularized by $L2$.

Prediction Module: With its ability to extract more combinatorial features, the MLP Module is drawn into the Prediction Module. It composes with four layers and each neuron layer undergoes processing by batch normalization [29] before activation to ensure the same distribution of input data. The MLP output L_4 is concatenated with C to complete classification by softmax layer in the prediction module, as follows:

$$\begin{cases} y = -1, \text{if } soft\max(concat(L_4, C)) \leq 0.5 \\ y = +1, \text{if } soft\max(concat(L_4, C)) > 0.5. \end{cases} \tag{15}$$

This is a binary classification where $L_4 \in \mathbb{R}$ and y is -1 or $+1$. When $y = -1$, the score of the candidate answer is minus 1, indicating that the candidate answer has a weaker preference. When $y = +1$, the score of the candidate answer is added with 1, indicating that the candidate answer has a stronger preference. The process of preference prediction and ranking is carried out for all candidate answers of each question. Finally, the candidate answers are ranked according to their scores.

IV. EXPERIMENTS

In this paper, all the experiments are implemented on a 64-bit Ubuntu MATE 16.04 operating system. It has a server equipped with Intel (R) Xeon (R) CPU E5-2650 V4 @ 2.20 at 48 GHz, 128GB RAM, and two 12GB NVIDIA GeForce GTX TITAN XP graphics cards. Anacaonda3 and TensoFlow-1.8 with CUDA 9.0 are used to construct the development environments of the deep learning model.

A. Dataset and Evaluating Criterion

Dataset: In our scheme, Zhihu and HealthTap datasets are used to construct training samples. In phase 1, approximately 3000 and 8000 questions and their associated answers are randomly selected from HealthTap and Zhihu, respectively. Each of these questions has at least three answers to ensure the validity of the training samples. We then construct the training set P with balanced positive and negative samples. The 10% samples of P are randomly selected as the validation dataset for each training epoch. The remaining questions and answers then serve as the testing set. To thoroughly verify the performance of our scheme, we randomly selected 1000 questions from the testing set and repeated the process 10 times. The construction process of the training set, validation set, and testing set are consistent for HealthTap and Zhihu. The details are summarized in Table I.

Evaluation Criterion: The performance of our scheme is measured by $P@K$, because precision is more important than recall in the answer selection [7]. The method is a widely accepted metric and could effectively evaluate the scheme performance. $P@K$ is represented in Eq. (16) as follows:

$$P@K = \frac{|C \cap T|}{|C|}, \tag{16}$$

where C is a set of the top K answers in the ranking list, and T is the set of the true ones in C. For a question, the real answers are decided by the "votes". $P@K$ stands for the proportion of true answers among those selected in the top K.

B. Performance Comparison with Baselines

To prove the validity of our scheme, PW-CrossCMN is compared with the following six state-of-the-art baselines.

LR: Logistic Regression (LR) is an effective and easy to understand classification algorithm that can be used to solve binary or multivariate classification problems. For an input sample x, its associated probability is output by LR to determine the category of x. In our experiments, LR is mainly used to solve binary classification problems.

XgbTree: eXtreme Gradient Boosting Tree (xgbTree) is an improved boosting algorithm based on the Gradient Boosting Decision Tree (GBDT). It uses regression tree as an internal decision tree. Calefato *et al.* evaluated 26 existing answer ranking models, and the experimental results show that xgbTree achieves the best performance in answer ranking [8].

RankSVM: Ranking SVM is a pair-wise learning method based on SVM to complete the ranking task. The basic idea is to transform the ranking problems into pair-wise classification problems and solve it by using the SVM classification model. RankSVM was used by Cao *et al.* to automatically optimize search engine retrieval quality through click-through rate data [14]. Hieber *et al.* used this method to improve the answer ranking performance of the social Q&A portal [15].

AdaRank: AdaRank is likewise an improved boosting algorithm proposed by Xu *et al.* to solve the problem of document retrieval in LTR [12]. Different from the pair-wise and point-wise methods, AdaRank is a list-wise approach that can directly optimize the whole ranking list through evaluation criteria.

LambdaMART: LambdaMART is a list-wise approach of LTR using the underlying training method MART (That can similarly be regarded as GBDT). Lambda is the gradient used in the process of solving MART. It represents the direction and intensity of the next ranking optimization of the document list. LambdaMART had won the championship in "Yahoo! Learning to Rank Challenge" [4].

PLANE: Nie *et al.* provided the PLANE model to optimize the answers ranking, and it has made significant progress [7]. Except for the best answer, the model considers all other candidate answers of the same question are almost on a par. That means the non-best answers of the same question have no preference. According to this point, three preferences are constructed: positive, negative and neutral. The PLANE

TABLE I: Statistics of Total Data, Training Set and Selected Testing Samples in Two Datasets

CQA System	Total Data		Training set				Selected Testing Samples
	Questions	Answers	Questions	Answers	Pos.Pre.Pairs	Neg.Pre.Pairs	Question (10 times)
Zhihu.com	71856	168782	8328	61101	123300	123300	1000
HealthTap	40000	58093	2938	11725	50580	50580	1000

'Pos.', 'Pref.' and 'Neg.' are short for 'positive', 'preference' and 'negative', respectively.

TABLE II: Performance Comparison with the State-of-the-Art Baselines over Two Datasets.

Datasets	Methods	Learning Approach	P@1	P@2	P@3	P@4	P@5
HealthTap Dataset	LR	Point-wise	16.28	32.11	48.57	63.42	79.62
	RankSVM	Pair-wise	22.80	41.33	52.98	70.63	82.14
	AdaRank	List-wise	24.67	42.84	53.32	72.29	83.30
	LambdaMART	List-wise	28.32	47.53	60.82	74.51	86.67
	xgbTree	List-wise	30.55	48.37	65.27	75.45	86.93
	PLANE	Pair-wise	33.81	53.28	67.88	78.68	88.67
	PW-CNN	Pair-wise	**40.05**	**59.99**	**73.4**	**84.06**	**92.23**
	PW-DNN	Pair-wise	**42.55**	**62.13**	**75.65**	**85.33**	**92.86**
	PW-CrossCMN	Pair-wise	**43.11**	**63.15**	**76.31**	**85.89**	**93.03**
Zhihu Dataset	LR	Point-wise	33.62	50.43	65.57	73.81	84.16
	RankSVM	Pair-wise	34.22	54.47	68.41	79.13	88.11
	AdaRank	List-wise	33.31	55.38	70.34	82.16	90.68
	LambdaMART	List-wise	34.82	56.43	71.47	79.36	88.65
	xgbTree	List-wise	43.72	60.84	72.11	81.93	90.69
	PLANE	Pair-wise	44.61	63.93	74.36	83.54	91.67
	PW-CNN	Pair-wise	**44.99**	**63.35**	**75.37**	**84.57**	**91.81**
	PW-DNN	Pair-wise	**44.76**	**63.17**	**75.77**	**85.19**	**92.78**
	PW-CrossCMN	Pair-wise	**49.24**	**68.78**	**80.9**	**88.76**	**94.12**

model optimizes the objective function of SVM and makes it concurrently smooth and differentiable.

Besides, two simplified versions of PW-CrossCMN scheme are likewise implemented, namely PW-CNN and PW-DNN. Both versions are likewise composed of two phases. The phase 1 of the two simplified schemes is the same as PW-CrossCMN to construct the preference vector. In phase 2 of the PW-CNN scheme, it builds a simple CNN model with two convolution layers, two Max-pooling layers, and two full-connected layers. The softmax layer is the last layer to implement classification. In phase 2 of the PW-DNN scheme, it builds a deep neural network with the second to seventh layers are hidden layers, and the softmax layer is also the last layer. It's worth noting that, as the training samples, the data format of the preference vectors need to be converted in PW-CNN.

In all experiments, our scheme is compared with the best performance of all the mentioned baselines as obtained with proper parameters. In our scheme, the preferences of training samples according to our preference relationship are positive and negative. Different with our scheme, the preferences in the PLANE model are positive, negative and neutral. In other baselines, the preferences are decided by the preference relationship: the best answer is preferable to the non-best answers for a question. The experimental settings conform to the original settings of all baselines. Our scheme achieves the best experimental performance with 113-dimensional features, including 50-dimensional LDA topic features, 50-dimensional Sent2vec features and 13-dimensional UserInfo features. The value of K in $P@K$ is set from 1 to 5 in the main experiment.

The final experimental results are shown in Table II, from which the following conclusions can be drawn:

1) Compared with other approaches of LTR, point-wise approaches do not consider the relative preference between two answers. Therefore, as a point-wise approach, LR only transforms the ranking problem into scoring a single answer and ignoring the relative order of answers. Thus, its performance is comparatively suboptimal. Pair-wise baselines, such as RankSVM, PLANE, and our three schemes outperform the point-wise approach. The reason is that such approaches consider the relative order and model the preference relationship between two QA pairs. Moreover, the ultimate intention of answers ranking is to put the best answers at the top, so the comparison between all candidate answers for a question is very important. List-wise approaches such as AdaRank, LambdaMART, and xgbTree have advantages compared with the point-wise LR and the traditional pair-wise RankSVM. The main reason is that list-wise approaches can optimize the answers ranking directly by minimizing a specific loss function. The PLANE performance is better than list-wise approaches by constructing the sample preferences with positive, negative and neutral for the first time, and it also optimizes the SVM objective function.

2) Clearly, the performance of our schemes exceeds those of all state-of-the-art baselines, and the PW-CrossCMN obtains the best experimental results. For all candidate answers of the same question, there are obvious preferences between them. Thus, we construct reliable preference vectors as the training samples based on the preference relationship. More-

TABLE III: Performance Comparison of the Four Schemes with the Number of Different Similar Questions.

Model	$P@K$	HealthTap					Zhihu				
		k=6	k=7	k=8	k=9	k=10	k=6	k=7	k=8	k=9	k=10
PLANE	P@1	30.91	27.21	26.53	24.29	21.17	42.86	41.10	40.82	39.21	37.72
	P@2	46.62	41.11	38.15	35.10	33.62	60.93	57.93	56.67	54.15	52.13
	P@3	62.03	54.82	49.96	45.51	42.73	71.21	67.62	65.13	63.49	61.48
	P@4	72.15	65.18	59.02	56.82	52.16	78.77	74.61	71.69	69.01	66.67
	P@5	81.60	74.74	67.36	63.37	59.74	84.65	79.53	76.61	74.08	71.38
PW-DNN	P@1	36.83	35.58	33.62	32.11	28.07	42.93	40.97	40.52	39.42	37.02
	P@2	56.63	51.96	47.38	45.82	42.33	60.75	58.48	56.85	54.83	52.16
	P@3	69.75	65.07	59.45	56.23	51.62	70.65	67.93	66.63	64.61	59.63
	P@4	79.12	73.41	67.13	63.57	60.63	79.12	75.34	72.01	69.82	65.17
	P@5	86.23	80.62	75.79	72.16	66.92	85.03	80.62	77.64	75.37	71.59
PW-CNN	P@1	35.05	33.17	30.29	28.86	26.97	43.75	43.12	42.51	40.94	38.55
	P@2	52.96	49.39	44.56	42.25	42.38	61.26	60.31	58.00	56.11	53.28
	P@3	66.75	61.36	55.62	53.69	51.24	72.23	71.03	67.89	66.82	63.13
	P@4	76.48	70.71	65.37	62.24	61.26	82.61	81.17	76.93	74.73	67.26
	P@5	85.51	78.92	72.18	70.64	67.39	86.98	85.24	81.15	77.06	72.71
PW-CrossCMN	P@1	38.92	35.76	33.66	33.01	30.92	43.82	43.53	42.62	41.05	38.82
	P@2	57.79	53.41	49.28	47.52	45.03	61.73	60.38	58.39	56.69	54.73
	P@3	70.46	65.54	59.95	58.24	55.74	73.51	71.16	68.76	67.28	63.84
	P@4	79.82	74.62	68.93	67.33	64.18	83.06	81.64	77.45	75.02	68.24
	P@5	86.97	82.15	77.42	74.28	71.25	87.23	85.42	82.03	78.36	73.69

k is the number of similar questions.

over, the network models we built can extract more effective information from the preference vectors. In addition, the PW-CrossCMN scheme achieves the best performance compared with PW-CNN and PW-DNN. It is because PW-CrossCMN implements the multi-network parallel convolution and the cross forward propagate of full-connected layers. Moreover, the scheme draws into the MLP module to extract the combination features in the prediction module.

3) The experimental results vary with different datasets. Those for on the Zhihu dataset for each method are always better than those for the HealthTap in Table II. One of the reasons is that the Zhihu datasets have twice more samples than HealthTap. The other reason is that for the Chinese dataset Zhihu, more refined word segmentation and delete stop-word processing is performed.

C. Robustness Evaluation

Our scheme returns the reliable answers for the user by searching similar questions in historical documents. In the real word, CQA systems can not accurately return similar questions for users' questions, especially the relevant questions are scarce. We thus have to enlarge k to introduce more similar questions [7], but it may introduce more noise in the pool of candidate answers and makes our scheme more difficult to complete the answer ranking task.

To validate the robustness of our schemes, we perform a plenty of experiments. The results are provided in Table III. We choose PLANE which is the best baseline to compare with our schemes. We then can get the conclusions as follows:

1) As the number of similar questions increases, the performance of each model decreases. This phenomenon is homologous on two different datasets. The noise of candidate answers in experimental data increases when k increases gradually. With the increase of similar questions, the number of irrelevant answers in candidate answers pool is also increasing. Therefore, the ranking performance of each model is declined.

2) The experimental results of our three models on two datasets are always better than those of the PLANE model, regardless of the value of k. Clearly, the performance attenuation in PW-CrossCMN scheme is less than that of the PLANE with the increase of k, which indicates the greater robustness of our scheme.

D. Evaluation of Feature Selection

The PW-CrossCMN model selects three feature types to construct the preference vector. Here we verify the validity of these features through relevant experiments. In Fig. 4, T is the Topic feature, S is the Sent2Vec feature, U is the UserInfo feature, and "$+$" represents a concatenation of the two types of feature.

Figs. 4(a) and 4(b) illustrate the experimental results of the PW-CrossCMN model on Zhihu and HealthTap datasets, respectively. The experimental results clearly vary with different K in $P@K$. The feature combination of T, S, and U achieves the best performance compared with the other feature combinations.

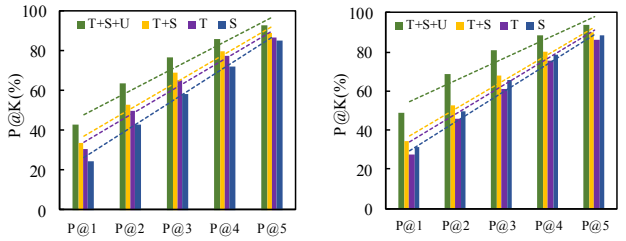
With the increase of feature dimensions and types, the PW-CrossCMN scheme can extract more useful feature information and achieve better experimental results. Thus, in our scheme, we select the combination of T, S, and U as the final feature vector.

With the combinations of the different features, the performances of PLANE, PW-DNN, PW-CNN, and PW-CrossCMN are shown in Fig. 5. The experimental results on the Zhihu dataset have the same conditions as in and Fig. 6. As seen in the figure, our schemes are still more effective than

TABLE IV: The Effect of Word Segmentation of Chinese Text on the Four Model Performances

Model	Without Word Segmentation					Word Segmentation				
	P@1	P@2	P@3	P@4	P@5	P@1	P@2	P@3	P@4	P@5
PLANE	17.61	33.68	47.64	63.16	77.03	44.61	63.93	74.36	83.54	91.67
PW-CNN	24.98	44.63	59.44	72.57	83.55	44.99	63.35	75.37	84.57	91.81
PW-DNN	23.21	41.19	55.82	69.59	81.34	44.76	63.17	75.77	85.19	92.78
PW-CrossCMN	23.74	42.88	56.98	71.44	83.91	49.24	68.78	80.90	88.76	94.12

The language of the Zhihu dataset is Chinese.



(a) PW-CrossCMN at HT (b) PW-CrossCMN at ZH

Fig. 4: (a) and (b) represent the performance of PW-CrossCMN scheme with different features on HT and ZH datasets, respectively. HT is the abbreviation of HealthTap and ZH is the abbreviation of Zhihu.

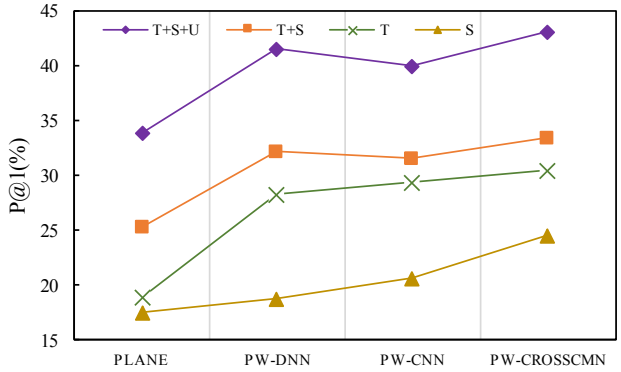


Fig. 5: These lines represent the performance of four models at P@1 with different features on HealthTap

the PLANE model. In all the methods, the PW-CrossCMN achieved the best performance.

By comparing Figs. 5 and 6, the experimental results vary with different models, features, and datasets. The Zhihu dataset is a general Chinese CQA system and the HealthTap dataset is a vertical English CQA system. This discrepancy between Zhihu and HealthTap causes different features. In addition, UserInfo features can improve model performance but do not contain any direct Q&A information. Rather, UserInfo mainly consists of personal information with a small amount of correlation with the Q&A text. Therefore, UserInfo features are not feasible to separately stablish the model.

E. The Effect of Word Segmentation in Chinese

Vocabulary is the smallest language unit that can independently express meaning in Chinese text, which is equivalent

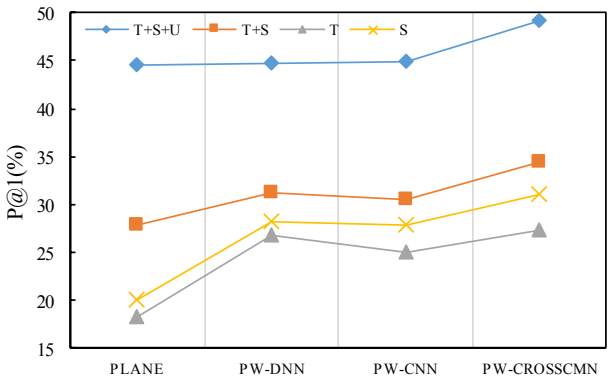


Fig. 6: These lines represent the performance of four models at P@1 with different features on Zhihu

to English words in a sentence. Word segmentation is highly important in extracting useful information from the Chinese text.

English words have natural spaces in between to separate them. It is easy to segment words by spaces when we preprocess English text. However, Chinese vocabularies have no separators and they are connected directly end-to-end in a sentence.

At present, the related technology of Chinese word segmentation has achieved considerable progress, and numerous excellent word segmentation methods have emerged. We adopt one of the widely used open source tools[7] to complete the word segmentation, and we remove the stop words in the Zhihu dataset.

The experimental results related to word segmentation in Zhihu are summarized in Table IV. Clearly, after word segmentation, the ranking performance of each model considerably improved at different depths of $P@K$. The performance of our models declined when words are not been segmented, but their answer ranking results remained remarkable than those of the PLANE model. This step likewise shows that our model has greater robustness than the baselines.

F. Network Optimization and Parameter Setting

The Tensorflow-1.8 framework is adopted to implement the network model of the PW-CrossCMN scheme. The learning rate directly controls the updating speed of the parameters, and therefore we employ the exponential decay method to

[7]https://github.com/fxsjy/jieba

realize the learning rate. The method mainly uses a large initial learning rate (over-sized initial learning rates lead to gradient disappearance) to rapidly obtain a better solution, and then gradually reduces the learning rate with iteration. This method not only accelerates the convergence speed but also increases the model stability in latter training periods.

We adopt the sliding average model method to increase the model robustness. This method implementation likewise requires the attenuation rate to control the model updating speed. We found that high attenuation rate leads to great model stability.

We fine-tune the parameters of the network with numerous experiments. The final values of parameters are as follows: the batch size is 256, the learning rate is 0.1 and the learning attenuation rate is 0.99. To alleviate over-fitting phenomenon and improve the generalization ability, we set the regularization coefficient at 0.0001 and the drop rate at 0.2. The average time spent on Zhihu and HealthTap datasets for training one epoch is 4.11s and 5.34s, respectively. This is an acceptable time for deep learning.

V. CONCLUSION

In this study, we propose a novel scheme for answer ranking in the CQA system. The scheme mainly consists of two phases. In Phase 1, the preference vector is constructed, and in phase 2, the CrossCMN model is built to train the preference vector. Numerous experiments are conducted on two typical datasets to demonstrate the performance of our model. The results show that our scheme achieves the best performance compared with those of all the state-of-the-art baselines. Moreover, our scheme is stable and robust even when the future dimension changes and noise increases. In addition, we analyze the impact of word segmentation in Chinese. Finally, the process of parameter adjustment is elaborated.

ACKNOWLEDGMENT

This work was supported in part by grants from the National Key Research and Development Program of China (No. 2016YFB1000903), and the National Natural Science Foundation of China (Grant No. 61627811 and 61401351), and the Key Project of Trico-Robot Plan of NSFC under Grant 91748208.

REFERENCES

[1] J. Yin, X. Jiang, Z. Lu, L. Shang, H. Li, and X. Li, "Neural generative question answering," *International Joint Conference on Artificial Intelligence*, pp. 2972–2978, 2016.

[2] C. Liu, S. He, K. Liu, and J. Zhao, "Curriculum learning for natural answer generation," *International Joint Conference on Artificial Intelligence*, pp. 4223–4229, 2018.

[3] Z. Ji and B. Wang, "Learning to rank for question routing in community question answering," *ACM International Conference on Conference on Information & Knowledge Management*, pp. 2363–2368, 2013.

[4] X. Cheng, S. Zhu, S. Su, and G. Chen, "A multi-objective optimization approach for question routing in community question answering services," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 9, pp. 1779–1792, 2017.

[5] S. Macavaney, A. Yates, A. Cohan, L. Soldaini, K. Hui, N. Goharian, and O. Frieder, "Characterizing question facets for complex answer retrieval," *International ACM Sigir Conference on Research and Development in Information Retrieval*, pp. 1205–1208, 2018.

[6] B. Kratzwald and S. Feuerriegel, "Adaptive document retrieval for deep question answering," *Empirical Methods in Natural Language Processing*, pp. 576–581, 2018.

[7] L. Nie, X. Wei, D. Zhang, X. Wang, Z. Gao, and Y. Yang, "Data-driven answer selection in community qa systems," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 6, pp. 1186–1198, 2017.

[8] F. Calefato, F. Lanubile, and N. Novielli, "An empirical assessment of best-answer prediction models in technical q&a sites," *Empirical Software Engineering*, pp. 1–48, 2019.

[9] D. H. Dalip, M. A. Gonc¸alves, M. Cristo, and P. Calado, "Exploiting user feedback to learn to rank answers in q&a forums: a case study with stack overflow," *in Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM*, 2013, pp. 543–552.

[10] C. Shah and J. Pomerantz, "Evaluating and predicting answer quality in community qa," *Proceeding of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 411–418, 2010.

[11] J. Bian, Y. Liu, E. Agichtein, and H. Zha, "Finding the right facts in the crowd: factoid question answering over social media," *The Web Conference*, pp. 467–476, 2008.

[12] J. Xu and H. Li, "Adarank: a boosting algorithm for information retrieval," *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 391–398, 2007.

[13] Q. V. Le and T. Mikolov, "Distributed representations of sentences and documents," *International Conference on Machine Learning*, pp. 1188–1196, 2014.

[14] Y. Cao, J. Xu, T. Liu, H. Li, Y. Huang, and H. Hon, "Adapting ranking svm to document retrieval," *International Acm Sigir Conference on Research & Development in Information Retrieval ACM*, pp. 186–193, 2006.

[15] F. Hieber and S. Riezler, "Improved answer ranking in social question-answering portals," *International Workshop on Search & Mining User-generated Contents ACM*, pp. 19–26, 2011.

[16] V. R. Carvalho, J. L. Elsas, W. W. Cohen, and J. G. Carbonell, "Suppressing outliers in pairwise preference ranking," *Proceeding of the 17th ACM Conference on Information and Knowledge Mining - CIKM*, pp. 1487–1488, 2008.

[17] T. M. Lai, T. H. Bui, and S. Li, "A review on deep learning techniques applied to answer selection," *COLING*, pp. 2132–2144, 2018.

[18] M. Tan, B. Xiang, and B. Zhou, "Lstm-based deep learning models for non-factoid answer selection," *arXiv: Computation and Language*, 2015.

[19] C. D. Santos, M. Tan, B. Xiang, and B. Zhou, "Attentive pooling networks," *arXiv preprint arXiv:1602.03609*, 2016

[20] W. Bian, S. Li, Z. Yang, G. Chen, and Z. Lin, "A compare-aggregate model with dynamic-clip attention for answer selection," *Acm on Conference on Information & Knowledge Management*, pp. 1987–1990, 2017.

[21] G. Shen, Y. Yang, and Z. Deng, "Inter-weighted alignment network for sentence pair modeling," *EMNLP*, pp. 1179–1189, 2017.

[22] Q. H. Tran, T. Lai, I. Zukerman, G. Haffari, T. H. Bui, and H. Bui, "The context-dependent additive recurrent neural net," *NAACL*, pp. 1274–1283, 2018 2018.

[23] Y. Tay, L. A. Tuan, and S. C. Hui, "Multi-cast attention networks for retrieval-based question answering and response prediction." *arXiv: Computation and Language*, 2018.

[24] T. Joachims, "Optimizing search engines using clickthrough data," *ACM Conference on Knowledge Discovery & Data Mining*, pp.133–142, 2002.

[25] H. Wu, W. Wu, M. Zhou, E. Chen, L. Duan, and H. Shum, "Improving search relevance for short queries in community question answering," *ACM*, pp. 43–52, 2014.

[26] T. Mikolov, K. Chen, G. S. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *International Conference on Learning Representations*, 2013.

[27] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," *International Conference on Computer Vision*, pp. 1026–1034, 2015.

[28] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, "Selfnormalizing neural networks," *Neural Information Processing Systems*, pp. 971–980, 2017.

[29] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *International Conference on Machine Learning*, pp. 448–456, 2015.