Graph-Based Tri-Attention Network for Answer Ranking in CQA

Wei Zhang[1], **Zeyuan Chen**[1], **Chao Dong**[1], **Wen Wang**[1], **Hongyuan Zha**[2], **Jianyong Wang**[3]
[1]School of Computer Science and Technology, East China Normal University
[2]School of Data Science, Shenzhen Research Institute of Big Data, The Chinese University of Hong Kong, Shenzhen
[3]Department of Computer Science and Technology, Tsinghua University
zhangwei.thu2011@gmail.com, chenzyfm@outlook.com, {51174500084, 51164500120}@stu.ecnu.edu.cn

Abstract

In community-based question answering (CQA) platforms, automatic answer ranking for a given question is critical for finding potentially popular answers in early times. The mainstream approaches learn to generate answer ranking scores based on the matching degree between question and answer representations as well as the influence of respondents. However, they encounter two main limitations: (1) Correlations between answers in the same question are often overlooked. (2) Question and respondent representations are built independently of specific answers before affecting answer representations. To address the limitations, we devise a novel graph-based tri-attention network, namely GTAN, which has two innovations. First, GTAN proposes to construct a graph for each question and learn answer correlations from each graph through graph neural networks (GNNs). Second, based on the representations learned from GNNs, an alternating tri-attention method is developed to alternatively build target-aware respondent representations, answer-specific question representations, and context-aware answer representations by attention computation. GTAN finally integrates the above representations to generate answer ranking scores. Experiments on three real-world CQA datasets demonstrate GTAN significantly outperforms state-of-the-art answer ranking methods, validating the rationality of the network architecture.

Introduction

Community-based question answering (CQA) is a pivotal online service gathering the wisdom of the crowd. It enables users to ask and answer questions, and draws much research interest (Ji et al. 2012). Popular CQAs, including StackOverflow, Quora, and Zhihu, have been the crucial entries for modern knowledge sharing and retrieval due to the accumulated millions of questions and answers. Consequently, it is of great importance for CQA platforms to support high-quality answer selection for ordinary users. Although the votes (e.g., thumbs-up) received by each answer can be the standards to prioritize high-quality answers, their accumulation inevitably needs notable time costs (see Figure 2). As such, automatic answer ranking is indispensable for quickly finding potentially popular answers with high quality.

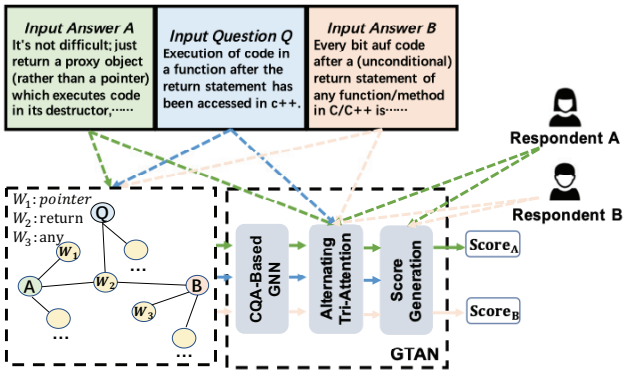


Figure 1: An illustrative example of automatic answer ranking in CQA for one question and its answers.

To this end, some conventional approaches (Shah and Pomerantz 2010; Hu et al. 2013; Tymoshenko and Moschitti 2015; Omari et al. 2016) conduct tedious feature engineering to find effective features, whereas they are labor-intensive and have limited generalization ability. Thus recent efforts (Shen et al. 2015; Qiu and Huang 2015; Fang et al. 2016; Zhang et al. 2017; Zhao et al. 2017; Hu et al. 2018; Lyu et al. 2019) are devoted to deep representation learning approaches for measuring the matching degree between target questions and answers. In particular, a few recent studies (Zhao et al. 2017; Hu et al. 2018; Lyu et al. 2019; Xie et al. 2020) additionally build respondent representations based on their IDs or historical answers, hoping to reveal user expertise on answer quality. It is worth noting that the respondent role embodies the unique characteristics of CQA, as compared to general question answering.

However, the above approaches are subject to two inherent limitations. Firstly, they learn each answer representation independently, yet the correlations between answers belonging to the same question are overlooked, which provide some clues in characterizing answers (see Figure 3 where top-ranked answers associated with the same question have larger similarities). Secondly, each question and respondent are assumed to only have one context-free representation for affecting answer representations, which does not conform to real situations. On the one hand, different parts of a question might be addressed by its different answers. As shown in

Figure 1, answer A gives more information w.r.t. the word "execution" in the question, while answer B concentrates more on the word "statement". Thus it is more promising to build answer-specific question representations to further promote answer representations. On the other hand, a respondent is usually with mixed professional interests, but only a part of them is associated with a given question-answer pair. As such, it might be better to build target-aware representation for a respondent.

To address the above two limitations, we propose the novel GTAN model (short for Graph-based Tri-Attention Network) to take questions, answers, and respondents as input, and output answer ranking scores. As shown in Figure 1, we first construct a graph for the question and its answers, wherein word nodes act as bridges to connect answers. Consequently, GTAN customizes a CQA-based graph neural network (GNN) to encode answer correlations into their representations. Based on the graph-based representations obtained by GNN, an alternating tri-attention method is developed to first build target-aware respondent representations based on QA-guided attention gating and answer-specific question representations through answer-guided question attention. Then the two types of representations in turn affect answers to form context-aware answer representations by question and respondent-guided answer attention. As such, representations of questions, respondents, and answers are alternatively updated. GTAN ultimately integrates the above-obtained representations to compute answer scores. In summary, we make the following contributions:

(1) We highlight the two limitations of existing answer ranking methods by showing the necessities of encoding answer correlations into answer representations and learning target-aware respondent representations and answer-specific question representations.

(2) We propose GTAN that contains two innovations: a customized GNN for encoding answer correlations and an alternating tri-attention mechanism to learn question, respondent, and answer representations. To our knowledge, this is the first study to build heterogeneous graphs and applying GNN for answer ranking in CQA.

(3) We demonstrate GTAN achieves the superior performance through extensive experiments on three real-world datasets and validate overcoming each limitation indeed promotes answer ranking performance.

Related Work

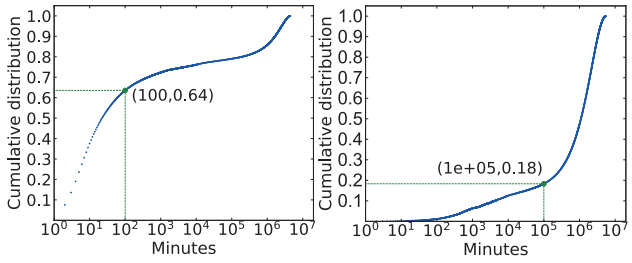
Existing approaches for answer ranking in CQA are mainly categorized into aspects: *feature-based* approaches and *representation learning-based* approaches. The first category heavily relies on manual-crafted features. An early study (Shah and Pomerantz 2010) proposes to use classification models (e.g., logistic regression) with the input of the features constructed from questions, answers, and respondents, such as the length of answer text and the number of questions answered by a respondent. More complex and advanced textual features, such as dependency-based structural features (Tymoshenko and Moschitti 2015) and novelty based features (Omari et al. 2016) have also been investigated to improve their performance. Besides, more than forty Boolean features are exploited to classify two answers into the same class or not in a separate stage (Joty et al. 2015). However, these feature-based approaches have low generalization ability due to the domain-specific features and are labor-intensive.

To alleviate the above issues, representation learning-based approaches have become the paradigm. Most of them regard the problem as question-answer text matching and learn low-dimensional feature representations for questions and answers. The work (Shen et al. 2015) calculates word-level cosine similarities based on word embeddings of questions and answers obtained by Skip-gram (Mikolov et al. 2013). Neural tensor network (Socher et al. 2013) is further utilized to model the matching degree with a non-linear tensor layer (Qiu and Huang 2015). To incorporate the role of respondents, a few recent studies learn their representations so as to better characterize the matching between questions and answers. The work (Hu et al. 2018) considers multi-modal content (e.g., text and image) and social relations between respondents to enrich the representations of their corresponding questions and answers. Both the studies (Zhao et al. 2017) and (Lyu et al. 2019) decompose the matching computation into two parts: question-answer matching (the same as previous studies), and question-respondent matching (newly added to incorporate the effect of respondents). The enhancement part of the study (Lyu et al. 2019) over the work (Zhao et al. 2017) is attributed to the answer representation learning with latent user expertise and hierarchical attention mechanisms, as compared to the previous work that models answer embeddings independent on their authors. Moreover, AUANN (Xie et al. 2020) employs generative adversarial networks (GANs) (Goodfellow et al. 2014) to help to learn from user past answers for acquiring relevant user representations, which are then directly fed into score computation.

As aforementioned, the above representation learning-based models have two limitations which motivate this study to pursue more effective representations. GNNs (Kipf and Welling 2017; Veličković et al. 2018), which have already been applied to general question answering (Cao, Aziz, and Titov 2019; Tu et al. 2019), and attention mechanism (Bahdanau, Cho, and Bengio 2015; Zhang et al. 2018) are the backbones to devise the CQA-based graph neural network for the first limitation and alternating tri-attention mechanism for the second limitation, respectively.

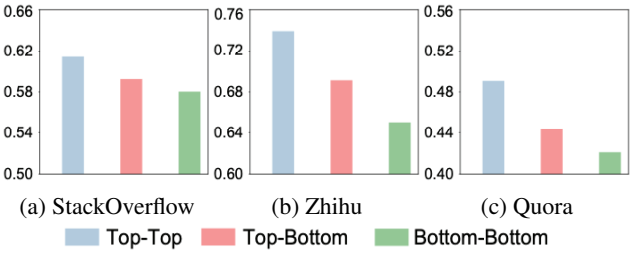
Preliminaries

Problem Formulation Assume we have a question set $\mathcal{Q}^{tr}$ for training. Taking question $q \in \mathcal{Q}^{tr}$ as an illustrative example throughout this paper, it consists of a textual description of length $l(q)$, denoted as $(w_1^q, w_2^q, ..., w_{l(q)}^q)$, and a set of answers, denoted as $\mathcal{A}_q = \{a_1^q, a_2^q, ..., a_{n(q)}^q\}$, where $n(q)$ is the number of answers. In reality, each answer is provided by a user (a.k.a. respondent). Hence we have a corresponding user set $\mathcal{U}_q = \{u_1^q, u_2^q, ..., u_{n(q)}^q\}$ for question q. Furthermore, we denote a_i^q as a composition of $l(a_i^q)$ words, i.e., $a_i^q = (w_{i,1}^q, w_{i,2}^q, ..., w_{i,l(a_i^q)}^q)$. All the words come from



(a) Answer Occurrence (b) Vote Occurrence

Figure 2: Distributions of the occurrence of answers and votes with respect to time interval in StackOverflow.



(a) StackOverflow (b) Zhihu (c) Quora

Figure 3: Textual similarities between answers with similar or different quality levels.

a pre-defined vocabulary set $\mathcal{W}$.

Given the above notations, the aim of automatic answer ranking in CQA is to learn a function: $f(q, a_i^q, u_i^q) \rightarrow s_i^q$ to generate score s_i^q of answer a_i^q, which is provided by user u_i^q for question q. Since the set of ground-truth vote counts, i.e., $\mathcal{V}_q = \{v_1^q, v_2^q, ..., v_{n(q)}^q\}(v_i^q \in \mathbb{Z}^*)$ is known in the training set, it is used to train the score function. To be specific, given two answers $a_i^q, a_j^q \in \mathcal{A}_q$ satisfying $v_i^q > v_j^q$, we aim to learn a score function $f(\cdot)$ to make the scores s_i^q and s_j^q satisfy the following inequality: $s_i^q > s_j^q$. The above procedure holds true for every question in $\mathcal{Q}^{tr}$. Through the learned score function, we can generate scores to automatically rank answers for a newly arrived question. Without causing ambiguity, we omit the superscript q in the remaining of the paper for simplicity.

Data Analysis In this part, we conduct preliminary data analysis for the following two aspects: 1) the distribution of time intervals w.r.t. questions, answers, and votes; 2) the existence of answer correlations. We have three real-world CQA datasets (i.e., StackOverflow, Zhihu, and Quora) for analysis, the details of which will be illustrated in the experimental section.

Due to the lack of vote timestamps in Zhihu and Quora, we only take StackOverflow for illustrating the first aspect. We first depict the distributions of time intervals between the timestamps when a question is raised and when its answers occur. As Figure 2a shows, about 64% of the answers are posted within 100 minutes after their questions are raised, accounting for a large proportion of all the answers. By contrast, in Figure 2b, merely 18% of the votes are collected within a time interval of 10^5 minutes (over 2 months) after the corresponding answers are provided. The disproportionate distribution implies that far before we can observe enough votes for answer ranking, an automatic answer ranking mechanism is needed to return potentially popular answers, which lays the foundation of the problem.

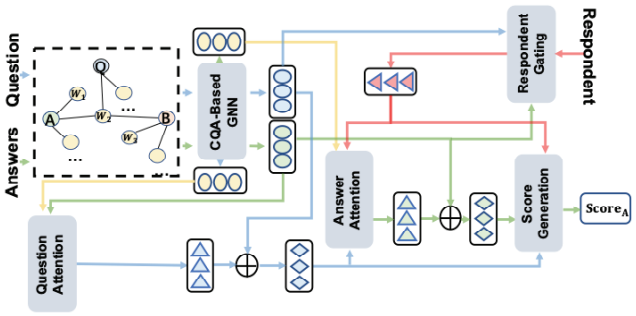


Figure 4: Architecture of GTAN (blue: questions, green: answers, red: respondents, yellow: words).

Furthermore, the existence of answer correlations is verified by comparing the textual similarities calculated on answers with different types of ranks. To accomplish this, we represent each answer through a pretrained Doc2Vec model (Le and Mikolov 2014) and use cosine similarity, which is insensitive to text length. Given answers' vote number, we rank them in descending order and thus high-quality answers are in top positions. We regard the answers in the range of the first 25-th percentile as "Top" ones and those after 75-th percentile as "Bottom" ones. For each question, we compute the average cosine similarities between "Top" and "Top" answers, "Top" and "'Bottom' answers, and "Bottom" and "Bottom" answers. Figure 3 reports that the similarities between "Top" and "Top" answers (i.e., "Top-Top") are consistently larger than the other two kinds of similarities in the three datasets. The law behind this phenomenon might be that high-quality answers belonging to the same question share some spirits, which is reflected by textual similarities. We also measure the cosine similarity between different types of answer corresponding to the figure. The similarity values w.r.t. "Top-Top" are greater than those w.r.t. "Bottom-Bottom" by 30.5%, 52.5%, and 20.9% on the StackOverflow, Zhihu, and Quora datsets, respectively. Therefore the intuition of good answers with larger similarities then inspires us to build graphs to learn answer correlations through representation propagation.

Proposed Methodology: GTAN

Model Overview: The architecture of GTAN is depicted in Figure 4. It briefly presents the information flow from input question, answer, and respondent, to answer score. Basically speaking, the CQA-based graph neural network is first executed for correlation modeling and getting the graph-based question, answer, and word representations. Consequently, respondent gating and question attention are conducted to gain target-aware respondent representations and answer-

specific question representations, which are in turn used for building context-aware answer representations through answer attention. Based on the above-obtained representations, answer score generation can be ultimately performed.

CQA-Based Graph Neural Network

As the graph example shown in Figure 4, we build a text graph for each given question, which has three types of nodes, i.e., question, answer, and word. It is intuitive that answer correlations are reflected by their shared words, and this is also true for the relevance between a question and its answers. With this intuition, we have answer-word edges and question-word edges. The weights are assigned by term frequency-inverse document frequency (TF-IDF) to capture the word importance for answers and questions. For ease of calculation, the adjacency matrix $\mathbf{A}$ is used to contain the TF-IDF weights of the graph, wherein $\mathbf{A}_{i,j} = 1$ if $i = j$.

Before introducing the CQA-based GNN, we first assume the input representations of question q, question words $w_j (j \in \{1, ..., l(q)\})$, answers $a_i (i \in \{1, ..., n(q)\})$, and answer words $w_{i,j} (j \in \{1, ..., l(a_i)\})$ for each a_i are represented as $\mathbf{e}$, $\mathbf{e}_{w_j}$, $\mathbf{e}_{a_i}$, and $\mathbf{e}_{w_{i,j}}$, respectively. They are fixed during the training stage, so as to remain consistent with testing on new questions and answers. For ease of illustration, we use an integrated embedding matrix $\mathbf{E}^0$ to include all of them, where $\mathbf{E}^0_i$ indicates the i-th node of the graph.

Firstly, taking node i as an example, CQA-based GNN aggregates the representations of its neighboring nodes as:

$$\mathbf{H}^0_i = \sum_{j \in N_r(i)} \mathbf{A}_{i,j} \mathbf{E}^0_j \,, \tag{1}$$

where $N_r(i)$ denotes the type-r neighboring set of node i.

Secondly, CQA-based GNN updates the target node representation by differentiating node types as follows:

$$\bar{\mathbf{E}}^1_i = \mathrm{ReLU}(\mathbf{W}^1_\tau [\mathbf{E}^0_i \oplus \mathbf{H}^0_i \oplus [\mathbf{E}^0_i \odot \mathbf{H}^0_i]]) \,, \tag{2}$$

where $[\ \oplus\ ]$ denotes a row-wise concatenation and $[\ \odot\ ]$ represents the element-wise product. $\mathbf{W}^1_\tau$ is a type-specific transformation matrix for either questions, answers, or words (i.e., $\tau \in \{1, 2, 3\}$), thus keeping aware of the heterogeneity of the graph. Noting that CQA-based GNN is simpler than complex heterogeneous graph neural networks (Zhang et al. 2019) and recent studies (Wu et al. 2019; He et al. 2020) demonstrate simplified GNNs could retain comparable performance as compared to complex GNNs. To relieve over-smoothing issue (Li et al. 2019), we adopt a gate to control information flows from previous layer.

$$\mathbf{E}^1_i = \alpha * \bar{\mathbf{E}}^1_i + (1 - \alpha) * \mathbf{E}^0_i \,, \tag{3}$$

$$\alpha = \sigma(\mathbf{W}^G [\bar{\mathbf{E}}^1_i \oplus \mathbf{E}^0_i] + \mathbf{b}^G) \,, \tag{4}$$

where $\mathbf{W}^G$ and $\mathbf{b}^G$ are trainable parameters, and σ stands for sigmoid function.

The above two procedures are repeated multiple times to realize high-order propagation. Supposing the number of propagation is set to T, GTAN ultimately returns the embedding matrix $\mathbf{E}^T$, from which we extract the **graph-based representations** $\widehat{\mathbf{e}}$, $\widehat{\mathbf{e}}_{a_i}$, $\widehat{\mathbf{e}}_{w_j}$, and $\widehat{\mathbf{e}}_{w_{i,j}}$ corresponding to the input representations. The trainable parameters in the module are summarized as $\Theta^G = \{\mathbf{W}^G, \mathbf{b}^G, \mathbf{W}^t_1, \mathbf{W}^t_2, \mathbf{W}^t_3\}|^T_{t=1}$.

Alternating Tri-Attention Mechanism

This mechanism contains QA-guided attention gating for respondent representations, answer-guided question attention for question representations, and question and respondent-guided answer attention for answer representations.

QA-guided Attention Gating To capture the expertise relevant to the target question and answer, QA-guided attention gating is proposed to filter irrelevant information in each dimension of the respondent representation. Specifically, given the ID of respondent u_i, a look-up table manner is conducted over respondent embedding matrix $\mathbf{E}^U$ to get original respondent representation $\mathbf{e}_{u_i}$. Note that we focus on respondents that already have answers in this paper and regard modeling new respondents as future work. Consequently, the attention gating calculates **target-aware respondent representation** $\bar{\mathbf{e}}_{u_i}$ based on graph-based question and answer representations, which is defined as follows:

$$\bar{\mathbf{e}}_{u_i} = \sigma(\mathbf{W}^U [\widehat{\mathbf{e}} \oplus \widehat{\mathbf{e}}_{a_i}] + \mathbf{b}^U) \odot \mathbf{e}_{u_i} \tag{5}$$

where $\Theta^U = \{\mathbf{W}^U, \mathbf{b}^U, \mathbf{E}^U\}$ are trainable parameters.

Answer-guided Question Attention Using a unique vector for each question limits its expressive ability when affecting the representations of its different answers. Thus we propose to compute **answer-specific question representation** $\widetilde{\mathbf{e}}_i$. It is realized by regarding graph-based answer representations as queries in the following attention computation.

$$\alpha_j = \mathrm{Softmax}(\omega^Q \tanh(\mathbf{W}^Q [\widehat{\mathbf{e}}_{a_i} \oplus \widehat{\mathbf{e}}_{w_j}] + \mathbf{b}^Q)), \tag{6}$$

$$\widetilde{\mathbf{e}}_i = \sum_{j=1}^{l(q)} \alpha_j \widehat{\mathbf{e}}_{w_j}, \tag{7}$$

where α_j is an attention weight. $\Theta^Q = \{\omega^Q, \mathbf{W}^Q, \mathbf{b}^Q\}$ are trainable parameters. Eq. 6 indicates that if a question word has a closer relationship with the answer, it will contribute more to the question representation. Though quite simple, answer-specific question representations are indeed beneficial (see Table 3). In local tests, we have tried to incorporate respondent representations into queries but gain no significant improvements. This is intuitive since the major role of respondents is the expertise mainly reflected in the answer part. At last, we integrate answer-specific question representation $\widetilde{\mathbf{e}}_i$ with graph-based question representation $\widehat{\mathbf{e}}$ for answer a_i, i.e., $\bar{\mathbf{e}}_i = [\widetilde{\mathbf{e}}_i \oplus \widehat{\mathbf{e}}]$.

Question and Respondent-Guided Answer Attention We provide answer attention here to grasp the impact of the respondent and question on the answer representation. It takes $\bar{\mathbf{e}}_i$ and $\bar{\mathbf{e}}_{u_i}$ together as the query. As a result, the computation of the attention weight β_j over the target answer word representation $\widehat{\mathbf{e}}_{w_{i,j}}$ is formulated as follows:

$$\beta_j = \mathrm{Softmax}(\omega^A \tanh(\mathbf{W}^A [\bar{\mathbf{e}}_i \oplus \bar{\mathbf{e}}_{u_i} \oplus \widehat{\mathbf{e}}_{w_{i,j}}] + \mathbf{b}^A)). \tag{8}$$

It reveals that an important answer word should have greater relevance to both the question and the respondent. Similarly, we have $\Theta^A = \{\omega^A, \mathbf{W}^A, \mathbf{b}^A\}$ as learnable parameters. The **context-aware answer representation** $\widetilde{\mathbf{e}}_{a_i}$ is gotten by:

$$\widetilde{\mathbf{e}}_{a_i} = \sum_{j=1}^{l(a_i)} \beta_j \widehat{\mathbf{e}}_{w_{i,j}}. \tag{9}$$

Although the methodological aspects in alternating tri-attention mechanism shares partially similar spirits with existing studies attention-based studies (Lu et al. 2016; Nam, Ha, and Kim 2017; Zhang et al. 2018), it is seamlessly integrated with the GNN part for utilizing the graph-based representations to alternately learn effective respondent, question, and answer representations, showing some novel insights. Finally, we obtain an integrated answer representation $\bar{\mathbf{e}}_{a_i} = [\widetilde{\mathbf{e}}_{a_i} \oplus \widehat{\mathbf{e}}_{a_i}]$.

Score Generation and Training

Answer Score Generation To generate the ranking score of answer a_i, we first concatenate the three representations $[\bar{\mathbf{e}}_i \oplus \bar{\mathbf{e}}_{a_i} \oplus \bar{\mathbf{e}}_{u_i}]$ to have $\mathbf{z}_i$. Afterwards, we feed $\mathbf{z}_i$ into a feed-forward neural network with K fully-connected (FC) layers. Hence the corresponding score is calculated as:

$$s_i = \mathrm{FC}_K\Big(\big(\cdots \mathrm{FC}_1(\mathbf{z}_i;\theta_1);\cdots\big);\theta_K\Big), \tag{10}$$

where FC_k denotes the k-th fully-connected layer. As usual, we summarize the trainable parameters as $\Theta^F = \{\theta_1, ..., \theta_K\}$. In summary, we have the score function $f(q, a_i, u_i; \Theta)$ to generate answer ranking score s_i, where $\Theta = \{\Theta^G, \Theta^U, \Theta^Q, \Theta^A, \Theta^F\}$ cover all trainable model parameters.

Model Training The aim of model training is to learn optimal parameters Θ by referring to the ground-truth vote counts of answers. By convention (Zhao et al. 2017; Lyu et al. 2019), we adopt pairwise learning to rank and define the loss function for question q ($q \in \mathcal{Q}^{tr}$) as follows:

$$\mathcal{L} = \sum_{i,j\ (v_i > v_j)} \max\Big(0, c + f(q, a_j, u_j; \Theta) - f(q, a_i, u_i; \Theta)\Big), \tag{11}$$

where c ($c = 1$ for experiments) is the specified margin for pairwise ranking. The initialization of the input representations (e.g., word embeddings) and the used optimization algorithm, as well as some hyper-parameter settings are clarified in the experimental part.

Experiments

In this section, we elaborate the experimental setup and analyze the experimental results, aiming to answer:
RQ1: Can GTAN achieve better answer ranking performance than the state-of-the-art methods for answer ranking?
RQ2: How do the key model components and information types used in GTAN contribute to the overall performance?

Experimental Setup

Datasets To ensure the reliability of the results, we conduct experiments on three real-world datasets, i.e., StackOverflow[1], Zhihu (collected from its website), and Quora (Lyu et al. 2019). They correspond to three representative and complementary CQA platforms.

[1]https://archive.org/download/stackexchange/stackoverflow.com-Posts.7z

Table 1: Detailed statistics of the three datasets. Avg. Len. denotes the average length of answers.

Dataset	#Que.	#Ans.	#Resp.	Vocab.	Avg. Len.
StackOverflow	139128	884261	40213	52457	81.6
Zhihu	19357	473338	133203	65880	85.6
Quora	12664	66779	6061	44149	64.4

For all the Chinese text in Zhihu, we adopt Jieba[2] for word segmentation. To filter some noisy data, we adopt the following procedures: (1) We filter respondents with less than 5 answers. (2) We remove answers with less than 5 words and questions with less than 5 answers or too many answers (e.g., 1000). (3) We discard words that appear infrequently (e.g., less than 10 times). The above procedures could be repeated several times to have a stable data size. Finally, the basic statistics of the three datasets are summarized in Table 1. Particularly, the constructed graphs have on average over 200 nodes for the StackOverflow and Quora datasets and over 1000 nodes for the Zhihu dataset. We split the datasets into training sets, validation sets, and test sets, according to the ratios of about 8 to 1 to 1.

Evaluation Metrics To keep consistent with previous studies (Lyu et al. 2019), we adopt the following ranking metrics: (1) Mean Reciprocal Rank (MRR), which measures how the best answer is ranked by different approaches; (2) Normalized Discounted Cumulative Gain (NDCG), which provides a position-aware performance; (3) P@1 (short for Precision@1), which calculates the ratio that the best answers are ranked at the first position. For each question, the best answer used to calculate MRR and P@1 is the one with the largest vote count within that question. Similarly, the ground-truth positions of answers are determined by sorting their vote counts in descending order.

Baselines The representative baselines are as follows:
- **Doc2Vec** (Le and Mikolov 2014) naturally extends Word2Vec (Mikolov et al. 2013) by associating sentence-level representations accompanied by word embeddings. We utilize Doc2Vec to gain answer representations and add FC layers on top of them to generate answer scores.
- **S-matrix** (Shen et al. 2015) gains a semantic similarity matrix between a question and an answer in the word level. It is followed by convolutional layers to model the matrix.
- **CNTN** (Qiu and Huang 2015) characterizes the complex interactions between questions and answers with the combination of convolutional networks and neural tensor networks. Dynamic k-max pooling (Kalchbrenner, Grefenstette, and Blunsom 2014) is leveraged within it.
- **AMRNL** (Zhao et al. 2017) decomposes the similarity computation into the question-answer part and question-respondent part. Thus the role of respondents is first considered for the studied problem.
- **UEAN** (Lyu et al. 2019) performs both word-level and sentence-level attention computations, which enable question topic and user expertise to affect answer representa-

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

Table 2: Performance comparison of all adopted approaches on the three datasets.

Method	StackOverflow			Zhihu			Quora		
	P@1	MRR	NDCG@3	P@1	MRR	NDCG@3	P@1	MRR	NDCG@3
Doc2vec (Le and Mikolov 2014)	0.2985	0.5136	0.6664	0.1658	0.3555	0.3987	0.4387	0.6336	0.7179
S-matrix (Shen et al. 2015)	0.3123	0.5353	0.6751	0.1817	0.3653	0.4022	0.4393	0.6383	0.7234
CNTN (Qiu and Huang 2015)	0.3142	0.5400	0.6823	0.2013	0.3849	0.4275	0.4431	0.6414	0.7278
TextGCN (Yao, Mao, and Luo 2019)	0.3248	0.5475	0.6867	0.2011	0.3739	0.4192	0.4128	0.6174	0.7032
AMRNL (Zhao et al. 2017)	0.3884	0.5971	0.7327	0.3134	0.4937	0.5657	0.6856	0.8110	0.8811
UEAN (Lyu et al. 2019)	0.3967	0.6068	0.7439	0.3354	0.5130	0.5887	0.6877	0.8123	0.8828
AUANN (Xie et al. 2020)	0.4066	0.6132	0.7473	0.3418	0.5188	0.5958	0.6933	0.8152	0.8889
Ours (GTAN)	**0.4230**	**0.6265**	**0.7597**	**0.3556**	**0.5313**	**0.6134**	**0.7235**	**0.8368**	**0.9003**

tions. The decomposition of similarity computation is inherited from AMRNL for score generation.

- **TextGCN** slightly modifies original TextGCN (Yao, Mao, and Luo 2019), a heterogeneous graph convolutional network for document classification, to make it applicable to the problem. Specifically, it replaces classification loss with the pairwise ranking loss shown in Eq. 11.

- **AUANN** (Xie et al. 2020) utilizes GANs to help learning representations of respondents based on their historical answers. The relationships between questions and answers are considered as well.

All the above baselines are tuned on validation datasets to select the hyper-parameter configurations.

Implementation of GTAN The word representations for CQA-based GNN are initialized by Word2Vec. We have also tried BERT (Devlin et al. 2019) to provide pre-trained embeddings, but no significant gains are observed in the local tests. The initial question and answer representations are obtained by performing mean-pooling over their word embeddings, respectively. Based on the performance of GTAN on validation datasets, we set the number of propagation layers $T = 2$. The dimension of representations for words, questions, answers, and respondents are all set to 64. The number of FC layers K is set to 2 for a non-linear transformation. Adam (Kingma and Ba 2015) is adopted for model optimization, with the initial learning rate of 0.0005. We implement the models by Tensorflow and run the experiments on a GPU (Nvidia GeForce GTX 1080 Ti) with 11GB memory. All the results are averaged over three runs.

Experimental Results

Overall Comparison (`RQ1`) Table 2 shows the overall comparison of GTAN with different baselines. We observe Doc2Vec performs poorly on the three datasets. This conforms to expectation since it only considers the answer aspect by learning answer representations, while neither questions nor respondents are utilized. By further comparing Doc2Vec with S-matrix, we find the performance is improved to a certain degree. This demonstrates the quality of answer text largely determines the ranking positions and considering the relations between answers and questions could bring additional gains. Compared to S-matrix, CNTN and TextGCN achieve better performance in most cases. It makes sense because the two models have more advanced text relevance modeling techniques (i.e., CNTN with neural tensor networks and TextGCN with GCN) than S-matrix which is mainly based on word-level similarities.

For the models of AMRNL and UEAN, they calculate the relevance between questions and respondents, and achieve significantly better results than the above-mentioned methods which totally overlook the role of respondents. This can be attributed to the fact that the expertise of respondents heavily affects the quality of answers regarding specific topics, since respondents might be knowledgeable of different topics. Therefore, considering two different modalities of answers, i.e., answer text and respondent, is critical for better performance. Moreover, UEAN performs better than AMRNL since the former one could acquire more expressive answer representations. This is because UEAN incorporates respondent representations into the attention computation over answer words, but AMRNL does not directly correlate answer and respondent representations AUANN further improves the performance, thanks to the introduction of GANs to learn better respondent representations.

In the end, our model GTAN consistently yields the best answer ranking performance. To be specific, GTAN significantly improves the best baseline AUANN from 40.66% to 42.30% by P@1 on StackOverflow, from 59.58% to 61.34% by NDCG@3 on Zhihu, and from 81.52% to 83.68% by MRR on Quora, which is verified by a paired t-test. Compared to AUANN, GTAN is capable of explicitly encoding answer correlations into their representations by GNN modeling, and learning answer-specific question representations and target-aware respondent representations, which are promising as demonstrated.

Ablation Study (`RQ2`) To investigate the contributions of key components and information types adopted by GTAN, we make the following variants of GTAN: (1) **"w/o Graph"** denotes removing CQA-based GNN, equivalent to feeding input representations directly into alternating tri-attention computation. (2) **"w/o T-MAT"** means only using a shared matrix in Eq. 2, without differentiating node types. (3) **"w/o Que"** represents discarding all question information in GTAN. (4) **"w/o Res"** denotes discarding all respondent information. (5) **"w/o Tri-Att"** keeps the GNN part but removes tri-attention computation, where the output representations of GNN, along with respondent representations, directly form the integrated representations, as shown in Eq. 10. (6) **"w/o Que-Att"** only removes the question attention computation, denoting each question has only one

Table 3: Ablation study of GTAN.

Method	StackOverflow		Zhihu		Quora	
	P@1	MRR	P@1	MRR	P@1	MRR
w/o Graph	0.3828	0.5958	0.3302	0.4894	0.7037	0.8173
w/o T-MAT	0.4104	0.6156	0.3325	0.5109	0.7117	0.8275
w/o Que	0.4108	0.6160	0.3372	0.5154	0.7123	0.8289
w/o Res	0.3211	0.5448	0.1957	0.3784	0.4343	0.6342
w/o Tri-Att	0.4027	0.6117	0.3356	0.5029	0.7085	0.8259
w/o Que-Att	0.4127	0.6166	0.3432	0.5191	0.7162	0.8299
w/o Res-Att	0.4122	0.6171	0.3363	0.5135	0.7130	0.8290
w/o Res-Gate	0.4109	0.6160	0.3470	0.5202	0.7099	0.8273
Ours (GTAN)	**0.4230**	**0.6265**	**0.3556**	**0.5313**	**0.7235**	**0.8368**

unique representation for similarity calculation. (7) **"w/o Res-Att"** removes respondent representations from Eq. 8, meaning not considering the role of respondents in answer attention computation. (8) **"w/o Res-Gate"** removes QA-guided Attention gating, meaning directly using original respondent representations in GTAN.

Table 3 reports the results of the ablation study, from which we have the following key observations:

* "w/o Graph" suffers from noticeable performance degradation, showing the significant contribution of modeling answer correlations for ranking answers. Therefore we conclude that addressing the ***first limitation*** is indeed beneficial. Moreover, "w/o T-MAT" verifies that utilizing type-specific transformation matrices promotes the performance.
* Both "w/o Que" and "w/o Res" consistently underperform GTAN, conforming to the fact that in addition to answer text, considering question text and respondents is indispensable. One important observation is that "w/o Res" performs the worst among the variants, indicating that the expertise of respondents impacts a lot on answer popularity.
* The comparison between "w/o Que-Att" and GTAN validates the necessity of introducing answer-specific question representations. By further comparing "w/o Res-Gate" with GTAN, we can see obtaining target-aware respondent representations is effective as well. As such, we conclude that handling the ***second limitation*** is effective.
* In addition, the results of "w/o Tri-Att" show the advantages of the involved attention computations as a whole. And the comparison between "w/o Res-Att" and GTAN indicates the effectiveness of considering respondent effect on answer representations (e.g., topic authority).

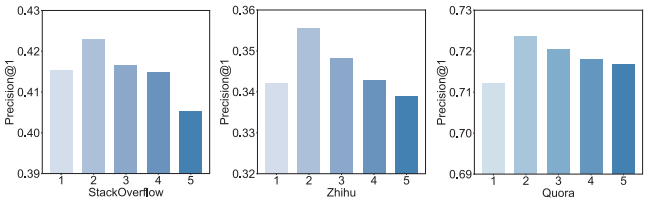
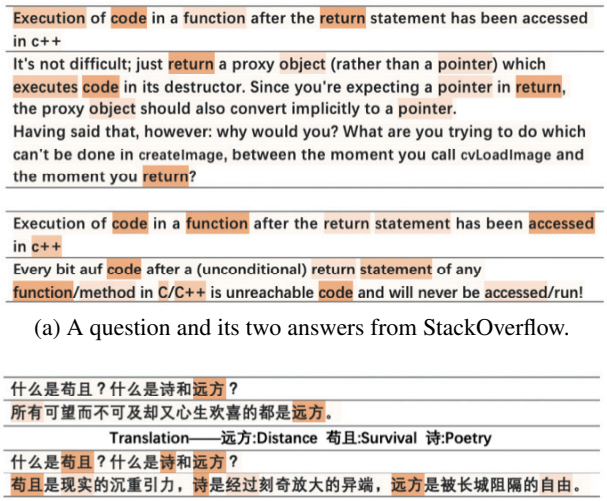


Figure 5: Result variation w.r.t. different layer number.

Impact of Propagation Layer Number We investigate whether it is beneficial for encoding high-order relations, not just the direct connections between answers and words.



(a) A question and its two answers from StackOverflow.

(b) A question and its two answers from Zhihu.

Figure 6: Attention visualization on questions and their answers. Darker colors mean larger attention weights.

We test the layer numbers in the range of $\{1, 2, 3, 4, 5\}$. Figure 5 depicts the variation trends of the results on the three datasets, from which we observe that:

* When increasing the layer number from 1 to 2, the improvements are noticeable. This makes sense because two-layered propagation could enable correlation modeling between different answers via answer-word edges, while one-layered propagation cannot achieve this. Thus it again verifies the importance of encoding answer correlations.
* When further adding propagation layers, the performance presents downward trends. The reason might be that adding more layers leads to over-fitting issue and makes answer representations not so distinguishable for judgment.

Training Efficiency We compare the training efficiency of GTAN with the best baseline model AUANN. Table 4 demonstrates that training GTAN gains much higher efficiency than training AUANN. This phenomenon is reasonable because AUANN demands recurrent neural networks for modeling word sequences, which incurs poor parallel computing in GPUs.

Table 4: Average training time cost per question (ms).

Method	StackOverflow	Zhihu	Quora
AUANN	63.33	128.56	41.22
Ours (GTAN)	**17.02**	**25.71**	**12.95**

Case Study To investigate how the tri-attention mechanism works on real cases, we present one question example and its two answers from StackOverflow and Zhihu, and visualize the attention weights in Figure 6. We observe: (1) Given a question, the mechanism enables its attention weights to be adaptive to different answers. As the second question example shows, its question is composed of two

parts. While the first answer only addresses the second part about what is "Distance", the second answer presents a more comprehensive view of "Survival" in the first part and "Poetry" and "Distance" in the second part. As a result, the question representations should be adaptable to their emphasis. (2) The attention mechanism could find some important words in answers. For the first example about coding, the two answers have opposite conclusions. The first answer gives more details about the solutions. And its contained technical terms "object" and "pointer" that do not appear in the question text could be addressed by our mechanism.

Conclusion

In this paper, we study automatic answer ranking in CQA. We address the two limitations overlooked by the conventional studies by devising a novel graph-based tri-attention network (GTAN). It has the innovations of combining CQA-based GNN and alternating tri-attention mechanism to learn answer correlations for the first limitation and answer-specific question representations and target-aware respondent representations for the second limitation. Moreover, GTAN effectively integrates question, answer, and respondent representations for answer score computation. We conduct extensive experiments on three real-world datasets collected from representative CQA platforms and the results demonstrate the superiority of the proposed model, validating the contributions of the key model components.

Acknowledgments

The corresponding author Wei Zhang would like to thank the anonymous reviewers for their valuable suggestions. This work was supported in part by National Natural Science Foundation of China under Grant No. 62072182, 61532010, 61521002, and U1609220, in part by the foundation of Key Laboratory of Artificial Intelligence, Ministry of Education, P.R. China, and in part by Beijing Academy of Artificial Intelligence (BAAI).

References

Bahdanau, D.; Cho, K.; and Bengio, Y. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *ICLR*.

Cao, N. D.; Aziz, W.; and Titov, I. 2019. Question Answering by Reasoning Across Documents with Graph Convolutional Networks. In *NAACL*, 2306–2317.

Devlin, J.; Chang, M.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Burstein, J.; Doran, C.; and Solorio, T., eds., *NAACL*, 4171–4186.

Fang, H.; Wu, F.; Zhao, Z.; Duan, X.; Zhuang, Y.; and Ester, M. 2016. Community-based question answering via heterogeneous social network learning. In *AAAI*, 122–128.

Goodfellow, I. J.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A. C.; and Bengio, Y. 2014. Generative Adversarial Nets. In *NIPS*, 2672–2680.

He, X.; Deng, K.; Wang, X.; Li, Y.; Zhang, Y.; and Wang, M. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In *SIGIR*.

Hu, H.; Liu, B.; Wang, B.; Liu, M.; and Wang, X. 2013. Multimodal DBN for Predicting High-Quality Answers in cQA portals. In *ACL*, 843–847.

Hu, J.; Qian, S.; Fang, Q.; and Xu, C. 2018. Attentive Interactive Convolutional Matching for Community Question Answering in Social Multimedia. In *MM*, 456–464.

Ji, Z.; Xu, F.; Wang, B.; and He, B. 2012. Question-answer topic model for question retrieval in community question answering. In *CIKM*, 2471–2474.

Joty, S. R.; Barrón-Cedeño, A.; Martino, G. D. S.; Filice, S.; Màrquez, L.; Moschitti, A.; and Nakov, P. 2015. Global Thread-level Inference for Comment Classification in Community Question Answering. In *EMNLP*, 573–578.

Kalchbrenner, N.; Grefenstette, E.; and Blunsom, P. 2014. A Convolutional Neural Network for Modelling Sentences. In *ACL*, 655–665.

Kingma, D. P.; and Ba, J. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Kipf, T. N.; and Welling, M. 2017. Semi-supervised classification with graph convolutional networks. *ICLR* .

Le, Q.; and Mikolov, T. 2014. Distributed representations of sentences and documents. In *ICML*, 1188–1196.

Li, G.; Müller, M.; Thabet, A. K.; and Ghanem, B. 2019. DeepGCNs: Can GCNs Go As Deep As CNNs? In *ICCV*, 9266–9275.

Lu, J.; Yang, J.; Batra, D.; and Parikh, D. 2016. Hierarchical Question-Image Co-Attention for Visual Question Answering. In *NIPS*, 289–297.

Lyu, S.; Ouyang, W.; Wang, Y.; Shen, H.; and Cheng, X. 2019. What We Vote for? Answer Selection from User Expertise View in Community Question Answering. In *WWW*, 1198–1209.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *NIPS*, 3111–3119.

Nam, H.; Ha, J.; and Kim, J. 2017. Dual Attention Networks for Multimodal Reasoning and Matching. In *CVPR*, 2156–2164.

Omari, A.; Carmel, D.; Rokhlenko, O.; and Szpektor, I. 2016. Novelty based Ranking of Human Answers for Community Questions. In *SIGIR*, 215–224.

Qiu, X.; and Huang, X. 2015. Convolutional neural tensor network architecture for community-based question answering. In *IJCAI*, 1305–1311.

Shah, C.; and Pomerantz, J. 2010. Evaluating and predicting answer quality in community QA. In *SIGIR*, 411–418.

Shen, Y.; Rong, W.; Sun, Z.; Ouyang, Y.; and Xiong, Z. 2015. Question/answer matching for CQA system via combining lexical and sequential information. In *AAAI*, 275–281.

Socher, R.; Chen, D.; Manning, C. D.; and Ng, A. Y. 2013. Reasoning With Neural Tensor Networks for Knowledge Base Completion. In *NIPS*, 926–934.

Tu, M.; Wang, G.; Huang, J.; Tang, Y.; He, X.; and Zhou, B. 2019. Multi-hop Reading Comprehension across Multiple Documents by Reasoning over Heterogeneous Graphs. In *ACL*, 2704–2713.

Tymoshenko, K.; and Moschitti, A. 2015. Assessing the impact of syntactic and semantic structures for answer passages reranking. In *CIKM*, 1451–1460.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; and Bengio, Y. 2018. Graph attention networks. *ICLR* .

Wu, F.; Jr., A. H. S.; Zhang, T.; Fifty, C.; Yu, T.; and Weinberger, K. Q. 2019. Simplifying Graph Convolutional Networks. In *ICML*, 6861–6871.

Xie, Y.; Shen, Y.; Li, Y.; Yang, M.; and Lei, K. 2020. Attentive User-Engaged Adversarial Neural Network for Community Question Answering. In *AAAI*, 9322–9329.

Yao, L.; Mao, C.; and Luo, Y. 2019. Graph convolutional networks for text classification. In *AAAI*, 7370–7377.

Zhang, C.; Song, D.; Huang, C.; Swami, A.; and Chawla, N. V. 2019. Heterogeneous Graph Neural Network. In *SIGKDD*, 793–803.

Zhang, W.; Wang, W.; Wang, J.; and Zha, H. 2018. User-guided Hierarchical Attention Network for Multi-modal Social Image Popularity Prediction. In *WWW*, 1277–1286.

Zhang, X.; Li, S.; Sha, L.; and Wang, H. 2017. Attentive interactive neural networks for answer selection in community question answering. In *AAAI*, 3525–3531.

Zhao, Z.; Lu, H.; Zheng, V. W.; Cai, D.; He, X.; and Zhuang, Y. 2017. Community-based question answering via asymmetric multi-faceted ranking network learning. In *AAAI*, 3532–3539.