Dynamic Graph Reasoning for Conversational Open-Domain Question Answering

YONGQI LI and WENJIE LI, The Hong Kong Polytechnic University
LIQIANG NIE, Shandong University

In recent years, conversational agents have provided a natural and convenient access to useful information in people's daily life, along with a broad and new research topic, conversational question answering (QA). On the shoulders of conversational QA, we study the conversational open-domain QA problem, where users' information needs are presented in a conversation and exact answers are required to extract from the Web. Despite its significance and value, building an effective conversational open-domain QA system is non-trivial due to the following challenges: (1) precisely understand conversational questions based on the conversation context; (2) extract exact answers by capturing the answer dependency and transition flow in a conversation; and (3) deeply integrate question understanding and answer extraction. To address the aforementioned issues, we propose an end-to-end Dynamic Graph Reasoning approach to Conversational open-domain QA (DGRCoQA for short). DGRCoQA comprises three components, i.e., a dynamic question interpreter (DQI), a graph reasoning enhanced retriever (GRR), and a typical Reader, where the first one is developed to understand and formulate conversational questions while the other two are responsible to extract an exact answer from the Web. In particular, DQI understands conversational questions by utilizing the QA context, sourcing from predicted answers returned by the Reader, to dynamically attend to the most relevant information in the conversation context. Afterwards, GRR attempts to capture the answer flow and select the most possible passage that contains the answer by reasoning answer paths over a dynamically constructed *context graph*. Finally, the Reader, a reading comprehension model, predicts a text span from the selected passage as the answer. DGRCoQA demonstrates its strength in the extensive experiments conducted on a benchmark dataset. It significantly outperforms the existing methods and achieves the state-of-the-art performance.

CCS Concepts: • **Information systems** → **Question answering;**

Additional Key Words and Phrases: Conversational question answering, open-domain question answering

ACM Reference format:
Yongqi Li, Wenjie Li, and Liqiang Nie. 2022. Dynamic Graph Reasoning for Conversational Open-Domain Question Answering. *ACM Trans. Inf. Syst.* 40, 4, Article 82 (January 2022), 24 pages.
https://doi.org/10.1145/3498557

The work described in this paper was supported by Research Grants Council of Hong Kong (PolyU/5210919, PolyU/15207821), National Natural Science Foundation of China (62076212) and PolyU internal grants (ZVQ0).
Authors' addresses: Y. Li and W. Li, The Hong Kong Polytechnic University, Hong Kong, China; emails: liyongqi0@gmail.com, cswjli@comp.polyu.edu.hk; L. Nie, Shandong University, Qingdao, 250100 China; email: nieliqiang@gmail.com.

1046-8188/2022/01-ART82 $15.00
https://doi.org/10.1145/3498557

1 INTRODUCTION

In recent years, the rise of machine learning techniques has accelerated the development of conversational agents, such as Alexa,[1] Siri,[2] and Xiaodu.[3] These conversational agents provide a natural and convenient way for people to chit-chat, complete well-specified tasks, and seek information in their daily life. People often prefer to ask conversational questions when they have complex information needs or are of interest to certain broad topics. Although current personal assistant systems are capable of completing tasks and even conducting small talk, they cannot handle information-seeking conversations with complicated information needs that require multiple turns of interaction [33]. It is therefore essential to endow conversational agents with the capability of answering conversational questions, which introduces a broad and new research area, namely conversational **question answering** (**QA**).

The previous conversational QA methods [9, 34] often assume that a single gold passage that contains the answer is given. This simplification is not realistic and neglects the fundamental role of retrieval. In this article, we consider the problem of conversational QA in an open-domain setting, namely conversational open-domain QA, which aims to satisfy the users' complex information needs with the exact answers extracted from the Web in a multi-turn conversation. On the one hand, users' information needs are presented in a series of questions within a conversation, along with the problems of co-reference and ellipsis, and so on. On the other hand, it requires to extract text spans from the Web as exact answers to answer users' questions. Conversational open-domain QA is a more practical but rarely studied task. It is worth mentioning that Qu et al. [32] have claimed this task as open-retrieval conversational QA. However, they simply append history questions into the current question and directly employ a traditional open-domain QA pipeline. The intrinsic characteristics of conversational open-domain QA are not fully explored.

Despite its significance and value, building an effective conversational open-domain QA system is non-trivial due to the following challenges.

(1) Precisely understand conversational questions based on the conversation context. Unlike self-contained questions in the single-turn QA, there are some conversational dependencies between the current question and the conversation context, including both previous questions and previous answers in a conversation. Each question must be interpreted according to the conversation context. For example, as shown in Figure 1, the second question "Who influenced her musical style?" cannot be understood without knowing what "her" refers to, which can be resolved using the conversation context. However, not all the conversation context is equally important in understanding the current turn question, which is usually relevant to only a portion of the conversation context. It is indispensable to identify the relevant context and filter out the irrelevant context.

(2) Extract exact answers by capturing the dependency and the flow among answers. It is hard to extract a text span from millions of lengthy passages in the Web as an exact answer to the current turn question. Fortunately, since the questions in a conversation are coherent, the corresponding answers are usually related and organized in the Web passages that are logically connected. As shown in Figure 1, the passages containing the three answers in a conversation are connected via hyperlinks. Quantitatively, we have observed that in the QBLink dataset [17] about 60% answers can be found in the two-hop connected passages containing the history answers. Furthermore, as conversation goes, the corresponding answers usually hop via the hyperlinks and form an answer flow. The answer flow provides certain clues to indicate where the current turn answer may be located, but it is tough to capture the answer flow and incorporate it into the existing QA system.

[1] https://www.alexa.com/.
[2] https://www.apple.com/siri/.
[3] https://dueros.baidu.com/.

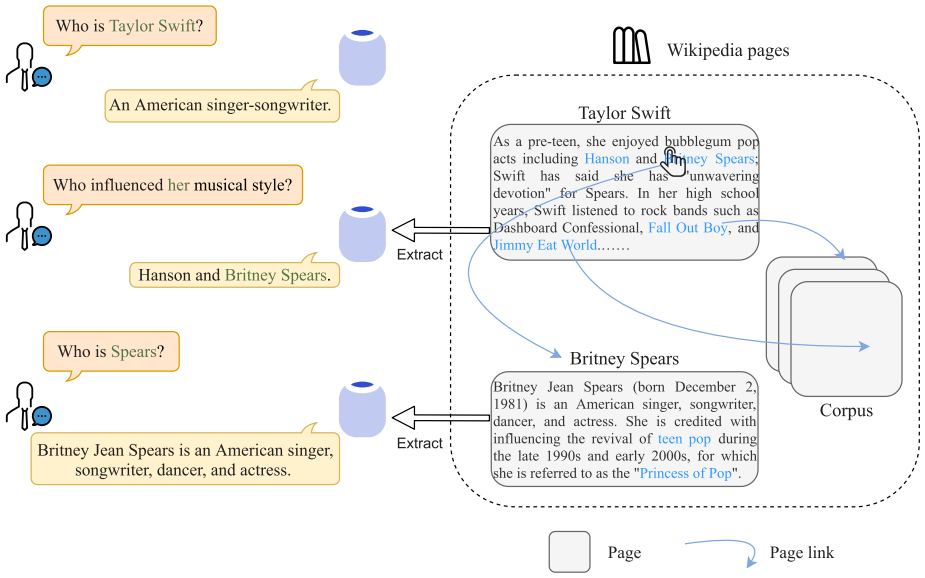


Fig. 1. Illustration of conversational open-domain QA. The user asks conversational questions in the conversation and the QA system extracts text spans from the Web corpus as exact answers to satisfy users' information needs. As conversation goes, the corresponding answers usually hop via the hyperlinks and form an answer flow, which provides certain clues to extract the answers.

(3) Deeply integrate question understanding and answer extraction. Certainly, if the question understanding component cannot understand the current question well, it is hard to accurately formulate users' information need and extract the correct answer. Reversely, if it is unable to extract the correct answer based on the retrieved passages, the question should be re-interpreted in order to provide more evidence for answer extraction. Therefore, how to design an effective framework, where these two components can work collaboratively and interact closely deserves careful consideration.

To address the aforementioned challenges, in this article, we present an end-to-end Dynamic Graph Reasoning method for Conversational open-domain QA, dubbed as DGRCoQA, as illustrated in Figure 2. DGRCoQA comprises a **dynamic question interpreter** (**DQI**), a **graph reasoning enhanced retriever** (**GRR**), and a Reader. DQI is developed to understand and formulate the conversational question, while GRR and the Reader are responsible to extract an exact answer to the question by retrieving the candidate answer passages from the Web and predicting a single specific text span as the answer, respectively. To be more specific, to better understand the conversational questions and explore more interactions among different components in the system, the DQI component is design to consider not only the *conversation context* and the current question but also the predicted candidate answers that are returned by the Reader, which can be deemed as the *QA context*. With the help of the QA context, DQI applies the attention mechanism to figure out more relevant information in the conversation context to refine the current question representation and then proceeds to another iteration of the answer extraction process. Since the candidate answers represent the feedback from the answer extraction and they are different in different iterations, DQI will dynamically attend to the different portions of the conversation context and formulate the dynamic representations that gradually get close to "optimal" for subsequent processing. Upon receiving the context-well-embedded question representation, GRR aims to retrieve from the Web the relevant answer passages from which the

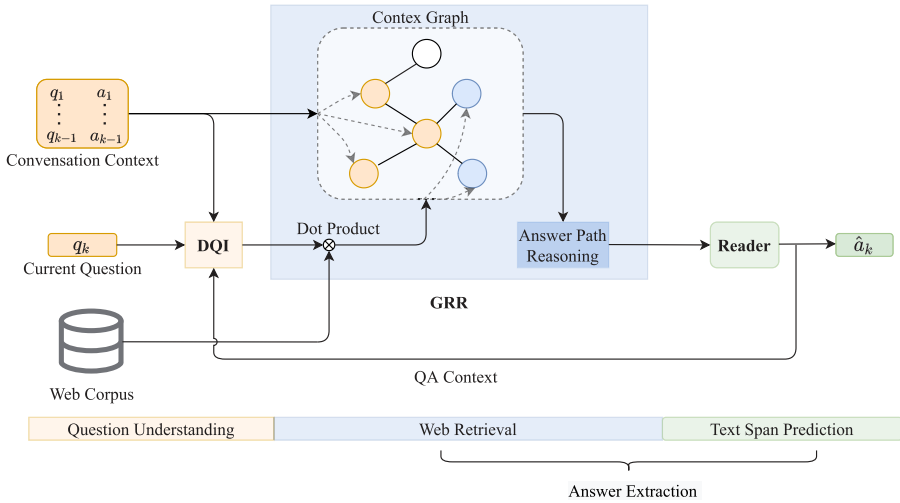


Fig. 2. Schematic illustration of our proposed DGRCoQA model, which comprises a DQI, a GRR, and a Reader. DQI is developed to understand and formulate the conversational question, while GRR and the Reader are responsible to extract an exact answer to the question by retrieving the candidate answer passages from the Web and predicting a single specific text span as the answer respectively.

answer can be located. GRR first constructs a context graph where the nodes are the passages containing history answers in the conversation context and the passages retrieved from the Web and the edges are the hyperlinks among them. It is noticed that the context graph is also dynamic because when the question representation formulated by DQI is dynamic, the passages retrieved are dynamic as well. To capture the answer flow, GRR generates a series of answer paths over the context graph, in which the last node in the path is a candidate passage and the other nodes are history answers. An answer path reasoning mechanism is introduced to rank these answer paths to reason about the most likely answer flow. In the light of this, passages are ranked based on the history answers and their paths over the context graph rather than being ranked separately. Finally, the best selected passages are delivered to the Reader, a MRC model, to predict a text span as the exact answer.

To summarize, the key contributions of this work are three-fold:

— We propose the DQI to understand conversational questions with both the conversation context and the QA context. On the one hand, the QA context is utilized to boost the modeling of conversation context and allows for dynamic question understanding. On the other hand, more interactions among different components are explored.
— We discover the close relations among answers and the important role of the answer flow in conversational QA. We propose the GRR to capture the answer flow by reasoning answer paths over a context graph. As far as we know, this is the first work in conversational QA that attempts to model the answer relations explicitly on a graph.
— Extensive experiments on the public dataset show significant performance improvement over the state-of-the-art methods. We have released the codes to the public.[4]

The rest of this article is structured as follows. In Section 2, we briefly review the related literature. In Section 3, we explain our proposed method in detail. This is followed by experimental

[4]https://github.com/liyongqi67/DGRCoQA.

results and analyses in Section 4. We finally conclude the work and propose future directions in Section 5.

2 RELATED WORK

Our work is closely related to conversational QA, open-domain QA, and conversational search.

2.1 Conversational QA

Different from recommended systems [6, 7] that route items to users for alleviating information overload, QA systems alleviate the problem by providing users simple and accurate answers. A great many QA systems [23, 29, 30, 51] were developed for this purpose, by utilizing external sources to obtain the correct answer, including KBQA, document-based QA, and community-based QA. And conversational QA, which aims to satisfy users' information needs in multi-turn conversations, is an emerging topic in the QA community. Based on the released large-scale dataset [9, 34, 35], researchers developed a series of methods to handle this problem. The previous conversational QA methods can be categorised into conversational **knowledge-based QA (KBQA)** [10, 35, 36] and conversational **machine reading comprehension (MRC)** [9, 34] regarding to the knowledge source. Conversational KBQA aims to answer conversational questions with the relevant facts in a knowledge base, whereas conversational MRC answers a series of questions over a given passage in a dialog. Comparatively, conversational open-domain QA is more challenging in a sense that it not only retrieves the relevant passages from the Web but also extracts the text answer spans. It is worth mentioning that conversational open-domain QA is rarely studied [1, 32]. Qu et al. [32] claimed this task as open-retrieval conversational QA and simply append history questions into the current question with a traditional open-domain QA pipeline. However, the characters of conversational open-domain QA are not fully explored.

The common problem that all above-mentioned conversational QA tasks have to address is how to understand the conversational questions. For example, there are often questions with ellipsis and/or coreference resolution problems. The common solution in the existing methods is to utilize the conversation context to reformulate the current question or refine its representation. A simple strategy is to pre-append all the conversation context [17] or heuristically select some words or sentences from the context [9] to expand the current question. Elgohary et al. [16] constructed the CANARD dataset, which is able to transform a context-dependent question into a self-contained question. Based on this dataset, a series of question rewriting methods have been proposed. Specifically, some regard question rewriting as a sequence-to-sequence task to incorporate the context into a standalone question. For example, Vakulenko et al. [37] used both retrieval and extractive QA tasks to examine the effect of sequence-to-sequence question rewriting on the end-to-end conversational QA performance. Differently, the authors in [40] modeled query resolution as a binary term classification problem. For each term appearing in the previous turns of conversation, it decides whether or not to add this term to the question in the current turn. Given the gold passage in conversational MRC, the authors in [33] designed a history attention mechanism to implement "soft selection" from conversation histories. The current conversational QA methods have presented various ways to modeling the conversation context, and utilizing the conversation context to enhance the current question representation. However, they ignore relations among answers (or answer passages) across the conversation turns. Some efforts [8, 24] in conversational MRC captures the conversational flow in the given passage by incorporating intermediate representations generated during the process of answering previous questions. Differently, we attempted to model relations more directly and explicitly by exploring the context graph.

2.2 Open-domain QA

Open-domain, which usually involves arbitrary topics in the real world, is used to distinguish from the closed-domain [38] and cross domain [20, 22] problem. Recently, open-domain QA [4, 41, 42] refers to the QA system that aims to answer questions from the Web, which has attracted wide attention from the academic field. In 2017, Chen et al. [4] first introduced neural methods to open-domain QA using a textual source. They proposed DrQA, a pipeline model with a TF-IDF based retriever and a neural network based reader that was trained to find an answer span given a question and a retrieved paragraph. Later, Wang et al. [41] added a ranker between the retriever and reader to rank the retrieved passages more precisely in 2018. Our proposed method also follows the retriever, ranker, and reader pipeline. In addition to TF-IDF and BM25 based retrievers, dense based retrievers have also been well developed recently [3, 25, 27], where all passages are offline encoded in advance to allow efficient large-scale retrieval. In [25], the authors showed that the retriever can be practically implemented using dense representations in open-domain QA, where embeddings are learned from a small number of questions and passages.

Benefiting from the widely application of Graph Neural Networks on various tasks [21, 45, 50, 52], there are also researchers exploiting the graph-based retriever for open-domain QA. In [2], authors introduced a graph based recurrent retrieval approach that is learned to retrieve reasoning paths over the Wikipedia graph to answer multi-hop open-domain questions. Distinctly, we aimed to capture the conversation flow with the help of a conversation context graph. Recently, multi-round retrieval methods have been proposed [14, 18, 46]. The authors in [18] proposed the multi-hop retrieval method, which iteratively retrieved supporting paragraphs by forming a joint vector representation of both question and returned paragraph. Das et al. [14] proposed a gated recurrent unit to update the question at each round conditioned on the state of the reader, where the retriever and the reader iteratively interact with each other. Our proposed method also includes a similar mechanism. Beyond existing methods, our focus is to explore the QA context, particularly the returned answers, to screen out the most useful information in the conversation context to better formulate the current question.

2.3 Conversational Search

Conversational search is also related conversational open-domain QA, since they both need to retrieve passages in multiple turns. The concept of search as a conversation has been around since the 1980s [12]. Until recently, the idea did not attract a lot of attention due to limitations in data and computing resources at the time. With the rapid adoption of smartphones and speakers, handling user's search requests in the conversational format is a pressing need in commercial systems [11, 19]. Recently, the TREC **Conversational Assistant Track** (**CAsT**) initiative constructed an evaluation benchmark for conversational search [13] and attracted researchers' attention. Based on the TREC CAsT datset, a series of methods are developed [13, 26, 48, 49]. Similar to question rewriting in conversational QA, some researchers reformulated conversational queries into de-contextualized and fully-grown and ad hoc queries that include all necessary information to represent the user's information needs, and then perform ad hoc retrieval with reformulated queries [48]. These solutions aim to reformulate the conversational query to an ad hoc query in the sparse bag-of-words space and then leverage standard sparse retrieval pipelines such as BM25 and BERT ranker [13, 26]. Besides of the query reformulation, some researchers aims to boost the retrieval by introducing dense retriever to conversational search. For example, Yu et al. [49] proposed a conversational dense retriever method benefiting from a teacher-student framework, which learns contextualized embeddings for multi-turn conversational queries and retrieves documents solely using embedding dot products.

Both conversational open-domain QA and conversational search require to retrieve passages in multi-turn conversations. However, conversational open-domain QA differs from conversational search from the following aspects: (1) Like the relation between open-domain QA and search [5], conversational open-domain QA is more challenging compared with conversational search because it requires not only to retrieve the relevant passages but also to extract text spans as exact answers. (2) The questions in conversational open-domain QA usually are natural language sentences and factoid, while queries in conversational search are more diverse and may only include some phrases. And (3) in conversational open-domain QA, there is usually one passage containing evidence for the system to extract the answer, while multiple passages can be regarded as positive to a query in conversational search [13, 32].

3 OUR PROPOSED METHOD

In this section, we first define some notations and give a overview of our method. And then we detail our proposed DRGCoQA model.

3.1 Notation and Overview

For the ease of problem formulation, we first declare some notations. In particular, we use bold capital letters (e.g., $\mathbf{X}$) and bold lowercase letters (e.g., $\mathbf{x}$) to denote matrices and vectors, respectively. We employ non-bold letters (e.g., X) to represent scalars, and Greek letters (e.g., λ) as parameters. We also use mathcal letters to denote set (e.g., $\mathcal{G}$). If not clarified, all vectors are in the column form.

Assume that the current turn in a conversation is k and the current question is q_k. We are given the conversation context $\mathcal{H}_k = \{q_1, a_1, \ldots, q_{k-1}, a_{k-1}\}$,[5] and a large passage collection $C = \{p_1, p_2, \ldots, p_{N_C}\}$, where N_C is the number of passages in C. Our research objective is to train an effective conversational open-domain QA system, towards extracting the correct answer a_k from a relevant passage in C.

As illustrated in Figure 2, our proposed DRGCoQA comprises the following three components. DQI receives as input the current question q_k, the conversation context $\mathcal{H}_k$, and the context, i.e., predicated answer $\hat{a}_k$, returned from the reader if any. It generates a question vector representation via an attention mechanism. Notably, it dynamically attend relevant information of $\mathcal{H}_k$ based on different returned candidate answers. We repeat the above process for N_t rounds. And then GRR is followed to retrieve a list of passages $\mathcal{P}_r = \{p_1, p_2, \ldots., p_{N_r}\}$ that is determined relevant to the question representation vector from a large passage collection C, where N_r is the number of retrieved passages. It first constructs a context graph $\mathcal{G}$, where node are passages that contains the history answers in conversation context and passages retrieved via a dense retriever, edges represent hyperlinks among passages. Hereafter, GRR selects passages that may contain the current answer and by reasoning the most possible answer path in the context graph. Finally, the passages $\mathcal{P}_r$ selected by GRR are passed to the Reader to extract the text answer span $\hat{a}_k$. We elaborate each component in the following sections.

3.2 Dynamic Question Interpreter

To better understand the current question q_k with the conversation context $\mathcal{H}_k$, we propose the DQI component. As shown in Figure 3, DQI utilizes the QA context $\hat{a}_k$ from the reader to attend relevant information in $\mathcal{H}_k$ rather than heuristically appending all of the $\mathcal{H}_k$ to q_k. In light of this, more interactions among different components are considered in the system. And thus DQI

[5]It is noticed that the history answers are also included in H_k if the true history answers are allowed to use.

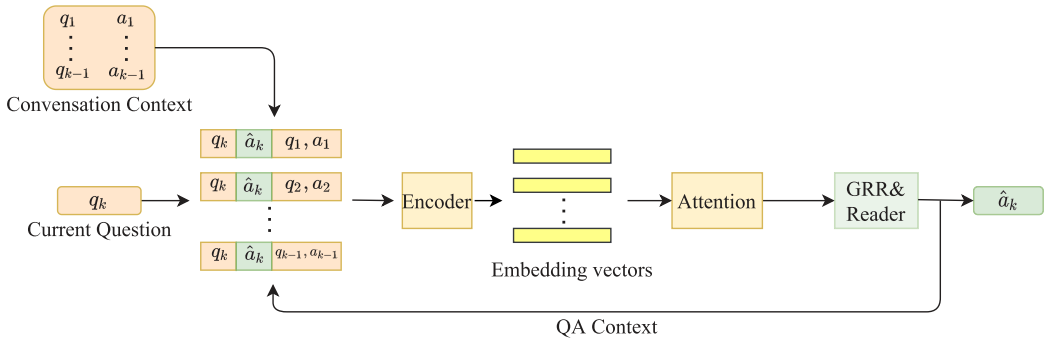


Fig. 3. Illustration of the DQI. DQI utilizes the QA context, i.e., predicted answers, to dynamically attend useful information of the conversation context. It is excepted to generate optimal question vectors for the subsequent components after receiving feedback from Reader.

generates the question embedding vector that is expected optimal for the following GRR and Reader components.

To obtain the QA context from the Reader, we first generate an initial question vector. We pre-append all history questions $\{q_1, q_2, \ldots, q_{k-1}\}$ to the current question q_k, denoted as q_k^*. To accommodate the BERT [15] based question encoder, we introduce two special tokens, [CLS] and [SEP]. q_k^* is represented as a text sequence "[CLS] q_1 [SEP], ..., q_{k-1} [SEP] q_k [SEP]". We obtain the question representation vector $\mathbf{v}_q$, which is formulated as,

$$\mathbf{v}_q = \mathbf{W}_q \mathbf{F}_q(q_k^*), \tag{1}$$

where $\mathbf{F}_q$ is the BERT based question encoder, $\mathbf{W}_q$ is the question projection matrix, and $\mathbf{v}_q \in \mathbb{R}^{d_q}$. Benefiting from the self-attention architecture in BERT, the correlation between the current question q_k and the items of conversation context is expected to learn. And then we deliver the current question vector $\mathbf{v}_q$ to the following GRR and Reader to obtain the predicated answer $\hat{a}_k$.

The predicted candidate answer $\hat{a}_k$ is then incorporated into the conversation context H_k and the current question q_k in the form of triplet $\{q_t^i\}_{i=1}^{k-1}$, where each triplet q_t^i contains the current question q_k, the candidate answer $\hat{a}_k$, and the ith history question q_i and answer a_i in H_k. It is formulated as "[CLS] q_k [SEP] $\hat{a}_k$ [SEP] q_i [SEP]a_i [SEP]". As such, interactions among the conversation text, the current question and the QA context are encouraged based on the self-attention mechanism in BERT. The representation of each triplet q_t^i, denoted as $\mathbf{v}_{q_t}^i$, is calculated as,

$$\mathbf{v}_{q_t}^i = \mathbf{W}_q \mathbf{F}_q(q_t^i). \tag{2}$$

A history attention network is then followed to aggregate the $k-1$ representation vectors into the refined question vector $\mathbf{v}_q$ with learned attention weights. Formally, the attention layer is defined as follows,

$$\begin{cases} \alpha_i = \dfrac{\exp(\mathbf{W}_a \mathbf{v}_{q_t}^i)}{\sum_{i=1}^{k-1} \exp(\mathbf{W}_a \mathbf{v}_{q_t}^i)}, \\ \mathbf{v}_q = \displaystyle\sum_{i=1}^{k-1} \alpha_i \mathbf{v}_{q_t}^i, \end{cases} \tag{3}$$

where $\mathbf{W}_a \in \mathbb{R}^{1 \times d_q}$, α_i denotes the attention weight of the ith triplet, and $\mathbf{v}_q \in \mathbb{R}^{d_q}$. The attention weights help to capture certain contexts that are more relevant to the current question. Basically, this is a turn-level attention mechanism and the more relevant items of conversation context

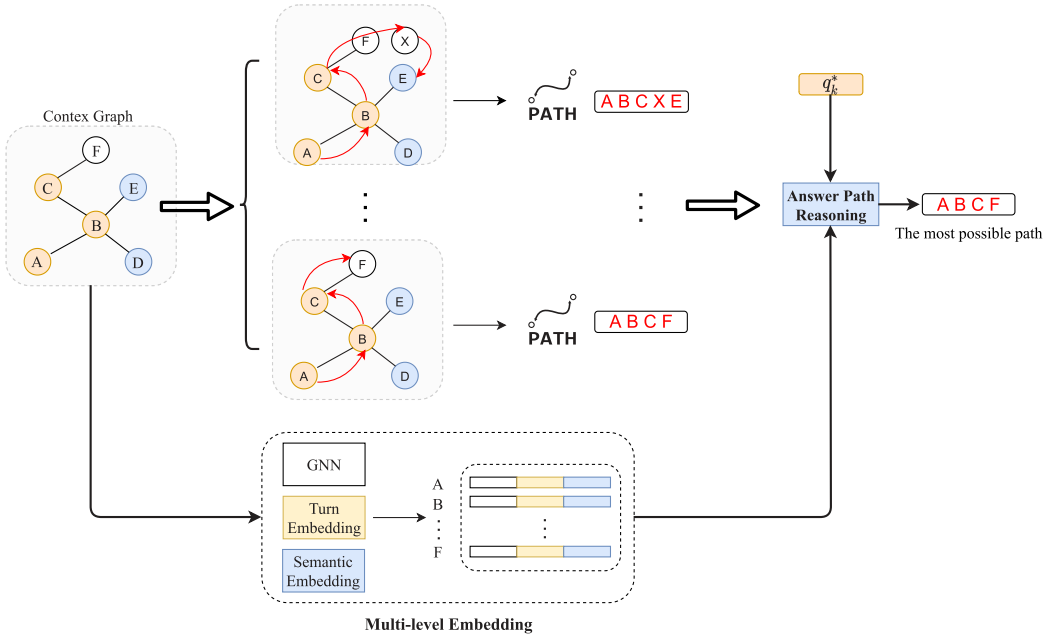


Fig. 4. Illustration of the GRR. GRR contains context graph construction, multi-level embedding layer, and answer path reasoning layer.

are expected to get more weights. We repeat the above process for N_t rounds, and obtain a number of predicated answer from the Reader. Based on different returned answers, DQI calculated different attention weights and thus dynamically attend relevant information in conversation context. Finally DQI generates the final question vector $\mathbf{v}_q$ in the final round.

3.3 Graph Reasoning Enhanced Retriever

Essentially, our proposed GRR aims to retrieves the relevant passages that contains the correct answer of the current question from a big collection C. Based on the observation that answers are highly related in conversational QA and logically distributed, we design GRR to model the answer flow by reasoning answer paths over a context graph. In general, the GRR goes through the following three steps: context graph construction, multi-level embedding, and answer path reasoning, as illustrated in Figure 4.

Context graph construction. We aims to construct a context graph $\mathcal{G} =< \mathcal{P}_\mathcal{G}, \mathcal{E} >$, where $\mathcal{P}_\mathcal{G}$ are some passages in C and $\mathcal{E}$ represent hyperlinks among them. To conduct a comprehensive context graph that contains enough information for our model to reason answer paths and retrieve the correct passages, we set $\mathcal{P}_\mathcal{G}$ from multi sources, formulated as

$$\mathcal{P}_\mathcal{G} = \{\mathcal{P}_{ha} \cup \mathcal{P}_{dr} \cup \mathcal{P}_{ex}\}, \tag{4}$$

where $\mathcal{P}_{ha}$, $\mathcal{P}_{dr}$, and $\mathcal{P}_{ex}$ denote the set of passages that contain previous gold/predicted answers, the passages returned by a dense retriever,[6] and the passages expanded via hyperlinks, respectively.

[6]We also include passages from a TF-IDF based retriever as supplementary. We pre-append the history questions to the current question and input the reformulated question to the TF-IDF retriever to retrieve the relevant passages.

Specifically, $\mathcal{P}_{ha}$ is obtained as follows,

$$\mathcal{P}_{ha} = \{p_i, p_i \in C \wedge p_i * \mathcal{H}_k\}, \tag{5}$$

where $p_i * \mathcal{H}_k$ denotes that one of the history answers in $\mathcal{H}_k$ can be extracted from the passage p_i.

$\mathcal{P}_{dr}$ is obtained from a dense retriever. In particular, for each passage p_i in C we obtain its representation $\mathbf{v}_p^i$ as follows,

$$\mathbf{v}_p^i = \mathbf{W}_p \mathbf{F}_p(p_i), \tag{6}$$

where $\mathbf{F}_p$ is the BERT based passage encoder, $\mathbf{W}_p$ is the passage projection matrix, $\mathbf{v}_p^i \in \mathbb{R}^{d_p}$, and d_p is equal to d_q. And we accumulate all the embedding vectors of passages in C as an embedding matrix $\mathbf{E}_p$, where $\mathbf{E}_p \in \mathbb{R}^{N_C \times d_p}$. It is critical that passage encodings are independent of questions in order to enable storing precomputed passage encodings and executing the efficient **maximum inner product search** (**MIPS**) algorithm [25, 27]. Otherwise, any new question would require re-processing the entire passage collection (or at least a significant part of it). Benefiting from this, we can calculate the similarity scores via the inner product of a given question embedding $\mathbf{v}_q$ and all of the passage efficiently, and select the top-N_{dr} passages $\mathcal{P}_{dr} = \{p_1, p_2, \ldots, p_{N_{dr}}\}$, where N_{dr} is the number of the passages in $\mathcal{P}_{dr}$.

So far, we have obtained $\mathcal{P}_{ha}$ and $\mathcal{P}_{dr}$. We then obtain $\mathcal{P}_{ex}$ by collecting passages that connected passages in $\mathcal{P}_{ha}$ or $\mathcal{P}_{dr}$ via hyperlinks. In particular, both the passages that point to passages in $\mathcal{P}_{ha} \cup \mathcal{P}_{dr}$ and the passages that jump from $\mathcal{P}_{ha} \cup \mathcal{P}_{dr}$ via hyperlinks are included into $\mathcal{P}_{ex}$. We will detail how these hyperlinks are obtained in Section 4.1. Via Equation (5), we obtain a comprehensive passage set $\mathcal{P}_{\mathcal{G}}$. We regard the passages in $\mathcal{P}_{\mathcal{G}}$ as nodes and hyperlinks among them as edges to construct a context graph $\mathcal{G}$, where $N_{\mathcal{G}}$ denotes the number of passages in $\mathcal{G}$. Notably, we regard the hyperlinks as undirected and thus the context graph is an undirected graph. It is noticed that since the question vector $\mathbf{v}_q$ formulated by DQI is dynamic in each round, different passages are retrieved and thus the context graph is dynamic.

Multi-level embedding. For passages in the context graph $\mathcal{G}$, we aims to represent them from the following three levels: graph structure level, conversation turn level, and semantic level. As claimed in the Introduction, the gold passages that contain correct answers are usually logically distributed in the corpus and connected via hyperlinks. Therefore, we aim to represent the passages from the graph structure level to address the relations among passages. Besides, the conversation turn level is used to indicate the passages' positions in a conversation, which is important for the conversation QA system [33, 34]. And the semantic embeddings are also included to represent the context of passages. These multi views of representations have been verified in various tasks [43, 44], which makes our DGRCoQA model can learn the passage embeddings from diverse views.

To take full advantage of the graph structure information, we apply a Graph Attention Network [39] to update the embeddings of the passages in $\mathcal{G}$. For each passage p_i in $\mathcal{G}$, we input it to a BERT encoder to obtain its embedding $\mathbf{v}_p^i$ by Equation (1). For each node i in $\mathcal{G}$, we can update its embedding via a GAT layer as,

$$\begin{cases} \alpha_{ij} = \dfrac{\exp(\sigma(\mathbf{W}_G^1([\mathbf{W}_G^2 \mathbf{v}_p^i, \mathbf{W}_G^2]\mathbf{v}_p^j)))}{\sum_{l \in \mathcal{N}_i} \exp(\sigma(\mathbf{W}_G^1([\mathbf{W}_G^2 \mathbf{v}_p^i, \mathbf{W}_G^2]\mathbf{v}_p^l)))}, \\ \mathbf{v}_{pg}^i = \sum_{j \in \mathcal{N}_i} \alpha_{ij} \mathbf{W}_G^3 \mathbf{v}_p^j, \end{cases} \tag{7}$$

where $\mathbf{v}_{pg}^i$ represents the graph structure aware embedding for each passage p_i, $\mathbf{v}_{pg}^i \in \mathbb{R}^{d_{pg}}$, and $\mathcal{N}_i$ is the neighbor nodes of node i in graph $\mathcal{G}$. [,] represents the concatenate operation and σ denotes the activate operation. In practice, we use multi-head attention mechanism and multi GAT layers. To indicate a passage in $\mathcal{G}$ contains the history answers or not, and further to indicate

its conversation turn if it contains a history answer, we initialize a turn embedding matrix $\mathbf{E}_t$. $\mathbf{E}_t$ is an embedding matrix with the size $(N_t + 1) \times d_{pt}$, where N_t is the max turn in the dataset and d_{pt} is the dimension of each vector in $\mathbf{M}_t$. For each passage p_i in $\mathcal{G}$, we encode its conversation turn into a one-hot vector, denotes as $\mathbf{v}_t^i$. It is worth mentioning that we set the conversation turn as 0 for the passages do not contain history answers. Thereafter, we obtain the conversation turn level representation for each passage p_i in $\mathcal{G}$ as follows,

$$\mathbf{v}_{pt}^i = \mathbf{E}_t \mathbf{v}_t^i. \tag{8}$$

Notably, turn embedding matrix $\mathbf{E}_t$ is updated in the training process and expected to learn the turn position information. We also reserve passages' embeddings obtained via Equation (6) as semantic embeddings, denoted as $\mathbf{v}_{ps}$. A forward neural network is followed to fuse the three level embeddings, formulated as,

$$\mathbf{v}_m^i = \mathbf{W}_m \left[\mathbf{v}_{pg}^i, \mathbf{v}_{pt}^i, \mathbf{v}_{ps}^i\right], \tag{9}$$

where $[,,]$ denotes the concatenate operation, and $\mathbf{W}_m \in \mathbb{R}^{d_p \times (d_{pg}+d_{pt}+d_p)}$.

Answer path reasoning. Although a GAT is utilized to model the graph structure and a turn embedding matrix is introduced to indicate the history answers, the relations between the current answer and history answers are not explicitly explored. As claimed before, as conversation goes, the answers hop via hyperlinks and form an answer flow path. To catch the in-depth answer relations, we further propose the answer path reasoning mechanism to capture the answer flow.

We first generate a series of candidate answer paths based on the context graph $\mathcal{G}$. An answer path is denoted as a sequence of passages. Specifically, for each passage p_i in $\mathcal{G}$, we generate an answer path $\mathcal{AP}_i$, as follows,

$$\mathcal{AP}_i = \{p_{a_1}, f(p_{a_1}, p_{a_2}, \mathcal{G}), p_{a_2}, \ldots, p_{a_{k-1}}, f(p_{a_{k-1}}, p_i, \mathcal{G}), p_i\}, \tag{10}$$

where $p_{a_{k-1}}$ denotes the passage contain the history answer a_{k-1} and $f()$ denotes a path search algorithms. The path search algorithm $f()$ returns a shortest path between two nodes over a graph. For example, $f(p_{a_1}, p_{a_2}, \mathcal{G})$ returns a sequence of the internal nodes between node p_{a_1} and node p_{a_2} over the graph $\mathcal{G}$. If there is no path between the two nodes, we set $f()$ to return a virtual node to indicate there is no path. For example, as shown in Figure 4, the answer path $\{A, B, C, X, E\}$ is generated via Equation (10) for node E, and X is a virtual node. Notably, the randomly initialized embedding of the virtual node is updated in the training process and expected to learn to indicate that there is no path.

Since the answer paths for each passage in $\mathcal{G}$ are obtained, we then reason the most possible answer flow path. We input each answer path $\mathcal{AP}_i$ with their corresponding embeddings obtained from multi-level embedding layer to a **Recurrent Neural Network** (**RNN**), formulated as,

$$\mathbf{v}_{\mathcal{AP}}^i = \mathbf{RNN}(\mathcal{AP}_i), \tag{11}$$

where $\mathbf{v}_{\mathcal{AP}}^i$ denotes the answer path aware embedding for each passage, and **RNN** denotes the sequential modeling network. Specifically, we input the sequential passages in the answer path associated with their embeddings $\mathbf{v}_m^i$ into the RNN. We obtain the embedding vector from the last hidden state as the final sequential embedding of an answer path. It is expected that the in-depth relations among answers can be captured via the sequential modeling network. Based on the answer path aware embeddings $\mathbf{v}_{\mathcal{AP}}^i$, we calculate the similarity scores between the question and the passages in $\mathcal{G}$. Specifically, we obtain the question embedding $\mathbf{v}_q$ by Equation (1) and calculate the similarity score S_a^i of passage p_i to $\mathbf{v}_q$ as follows,

$$S_a^i = \frac{\exp\left(\mathbf{v}_{\mathcal{AP}}^i \mathbf{v}_q^T\right)}{\sum_{i=1}^{N_{\mathcal{G}}} \exp\left(\mathbf{v}_{\mathcal{AP}}^i \mathbf{v}_q^T\right)}. \tag{12}$$

Finally, the top-N_r passages $\mathcal{P}_r$ are selected according to their ranks based on similarity calculation, where N_r denotes the number of passages in $\mathcal{P}_r$. $\mathcal{P}_r$ is then passed to the Reader for further processing.

3.4 Reader

Given the list of passages $\mathcal{P}_r$, the Reader aims to extract a text span as an exact answer for the question. Following the conventional methods, we also incorporate a Ranker component to re-ranks the passages more precisely. We first construct the reformatted text sequence $[q, p_i]$ by concatenating the question text and the passage p_i text. $[q, p_i]$ is "[CLS] q_k^* [SEP] p_i [SEP]". For each token $token_j$ in $[q, p_i]$, a BERT encoder generates its representation $\mathbf{v}_{to}^j$, and for the whole text sequence, BERT encoder generates its representation $\mathbf{v}_{q,p}^i$. Following the previous work [32], we set the start and end tokens to [CLS] for unanswerable questions and expect the system can learn to discriminate these unanswerable questions.

Ranker conducts a listwise re-ranking of the top-N_r passages received from GRR and calculates the re-ranking score for each passage p_i in $\mathcal{P}_r$ as,

$$S_b^i = \frac{\exp\left(\mathbf{v}_{q,p}^i \mathbf{W}_{ra}\right)}{\sum_{i=1}^{N_r} \exp\left(\mathbf{v}_{q,p}^i \mathbf{W}_{ra}\right)}, \tag{13}$$

where $\mathbf{W}_{ra} \in \mathbb{R}^{d_q \times 1}$. Reader then predicts an answer span by computing two scores for each token j in passages in $\mathcal{P}_r$ as the start token and the end token, respectively, formulated as,

$$\begin{cases} S_s^j = \dfrac{\exp\left(\mathbf{v}_{to}^j \mathbf{W}_s\right)}{\sum_{i=1}^{N_{to}} \exp\left(\mathbf{v}_{to}^j \mathbf{W}_s\right)}, \\ S_e^j = \dfrac{\exp\left(\mathbf{v}_{to}^j \mathbf{W}_e\right)}{\sum_{i=1}^{N_{to}} \exp\left(\mathbf{v}_{to}^j \mathbf{W}_e\right)}, \end{cases} \tag{14}$$

where N_{to} is the number of tokens, and both $\mathbf{W}_s$ and $\mathbf{W}_e \in \mathbb{R}^{d_q \times 1}$. Finally, the text span with the highest score is extracted as the answer. We will detail the inference process in the following section.

3.5 Training and Inference

Training. Recall that we encode the large collection of passages C offline for efficient retrieval. Specifically, we follow the previous work [49] to pretrain a passage encoder so that it can provide reasonably good retrieval results to the subsequent components for further processing. After offline encoding, a set of passage vectors are obtained. Note that while the parameters of the passage encoder $\mathbf{F}_p$ are fixed after pretraining.

We have DQI, GRR, Ranker, and Reader components in our system. We respectively define the GRR loss L_{GRR} and the ranker loss L_{ranker} as follows,

$$\begin{cases} L_{GRR} = -\sum_{i=1}^{N_r} \left(y \log\left(S_a^i\right) + (1-y) \log\left(1 - \left(S_a^i\right)\right)\right), \\ L_{Ranker} = -\sum_{i=1}^{N_r} \left(y \log\left(S_b^i\right) + (1-y) \log\left(1 - \left(S_b^i\right)\right)\right), \end{cases} \tag{15}$$

where S_a^i and S_b^i are the GRR score and ranker score of the passage p_i in $\mathcal{P}_r$, respectively. The reader loss L_{reader} is formulated as,

$$\begin{aligned} L_{Reader} = &- \sum_{j=1}^{N_{to}} \left(y_1 \log \left(S_s^j \right) + (1 - y_1) \log \left(1 - \left(S_s^j \right) \right) \right) \\ &- \sum_{j=1}^{N_{to}} \left(y_2 \log \left(S_e^j \right) + (1 - y_2) \log \left(1 - \left(S_e^j \right) \right) \right), \end{aligned} \tag{16}$$

where y_1 and y_2 indicate whether the token is the start token and the end token, respectively. Considering the limitation of GPU memory, we first train the DQI, the ranker, and the reader jointly via the sum of their losses, and then train DQI and GRR separately.

Inference. For each passage in $\mathcal{P}_r$, we obtain the GRR score S_a and the ranker score S_b by Equations (12) and (13), respectively. For each token, the reader then assigns it the probabilities of being the start token S_s and the end token S_e. Following the convention [15, 32], we consider the top 20 text spans only to ensure tractability. Invalid predictions, including the cases where the start token comes after the end token, or the predicted span overlaps with a question in the conversation context, are all discarded. The predicted score S of a potential answer is then calculated as,

$$S = S_a + S_b + S_s + S_e. \tag{17}$$

The model finally outputs the answer span with the maximum overall score to respond to the current question.

4 EXPERIMENTS

4.1 Dataset

To conduct experiments, we used the public available dataset OR-QuAC [32], which expands the QuAC dataset [9] to the open-domain QA setting. Each conversation in this dataset contains a series of questions and answers. For each question, there is a god passage that contains a answer text span. There are totally 5,644 conversations covering 40,527 questions. The passage collection is from the English Wikipedia dump file of 10/20/2019, and it contains about 11 million passages.

To construct the context graph as mentioned before, we collected hyperlinks in the Wikipedia. Specifically, in a Wikipedia page, there are some words are tagged with hyperlinks and can be clicked to turn to another Wikipedia page. For example, as shown in Figure 1, there is a sentence "As a pre-teen, she enjoyed bubblegum popacts including Hanson and Britney Spears" in the Wikipedia page[7] titled "Taylor Swift". The words "Hanson" and "Britney Spears" are linked to Wikipedia pages titled "Hanson (band)"[8] and "Britney Spears",[9] respectively. We crawled the hyperlinks in the Wikipedia via Wikiextractor.[10] The statistics of the above dataset are summarized in Table 1.

4.2 Experimental Settings

Evaluation Protocols. As claimed before, conversational open-domain QA requires to first retrieve the gold passage from the Web and then predict a text span as the answer. Therefore, we evaluate a conversation open-domain QA system from the aspects of Web retrieval and text span prediction. To evaluate the Web retrieval performance, we applied the Recall, **Mean Reciprocal**

[7] https://en.wikipedia.org/wiki/Taylor_Swift.
[8] https://en.wikipedia.org/wiki/Hanson_(band).
[9] https://en.wikipedia.org/wiki/Britney_Spears.
[10] https://github.com/attardi/wikiextractor.

Table 1. Statistics of the Dataset

	items	Train	Dev	Test
QA pairs	# Dialogs	4,383	490	771
	# Questions	31,526	3,430	5,571
	# Avg.Questions per Dialog	7.2	7.0	7.2
Collection	# passages	11 million		
Hyperlinks	# Hyperlinks	105 million		
	# Avg. hyperlinks per passage	17		

Rank (**MRR**), NDCG as well as MAP to evaluate the Retriever component of the baselines and our methods. Following the previous work [9, 32], we employed the word-level F1 and the **human equivalence score** (**HEQ**), which are two metrics provided by the QuAC challenge [9] to evaluate the text span prediction. HEQ computes the percentage of examples for which system F1 exceeds or matches human F1. It measures whether a system can give answers as good as an average human. This metric is computed on a question level (HEQ-Q) and a dialog level (HEQ-D).

Implementation Details. We utilized the pretrained passages embedding vectors released in [49] to make a fair comparison. The d_q, d_p are both set to 128, and N_r is set to 5, the same as [32]. Limited by our GPU memory, N_t is set to 2. There are two GAT layers, where the numbers of heads are 128 and 64, respectively. The passage embeddings are stored in a 2080Ti card and the model is in another 2080Ti card. We used the GRU with one layers and 128 hidden size for the RNN. To train our model, we set the batch size to 1 and use Adam optimizer with learning rate 0.0001.

4.3 Baselines

To justify the performance of CGRCoQA, we compared CGRCoQA with the existing conversational open-domain QA methods [31, 32]. We also compared our model with the methods from open-domain QA [4, 47], conversational question rewrite [48], and conversational search [28, 49], as follows,

- **DrQA [4]:** This model uses a TF-IDF retriever to retrieve relevant passages and a RNN based reader to extract answers. The original distantly supervised setting is not applied, since the full supervision is allowed in the dataset and adopted for all the methods.
- **BERTserini [47]:** It is a open-domain QA problen method. This model uses a BM25 retriever implemented in Anserini[11] and a BERT based reader. Their BERT reader is similar to ours.
- **ConvDR [49]:** It is a dense retriever method for conversational search, which learns contextualized embeddings for multi-turn conversational queries and retrieves documents solely using embedding dot products.
- **CQE and CQE hybrid [28]:** It integrates a conversational question rewrite module into conversational search pipeline and train the pipeline in an end-to end manner. This model yields competitive retrieval effectiveness against state-of-the-art multi-stage approaches. CQE-hybrid is the variant of CQE, which combines of BM25 and dense retrieval scores.
- **CQR(BM25) and CQR(ANCE) [48]:** CQR is a weakly supervised GPT-2 rewriter method for conversational search. It aims to generate self-contained questions given conversational questions. **CQR(BM25)** is with a BM25 retriever and a Bert based reader for fair comparison. **CQR(ANCE)** is a CQR followed a dense retriever and a Bert based reader.

[11] http://anserini.io/.

Table 2. Performance Comparison between our Proposed Model and State-of-the-art Baselines Over Develop and Test Set

Methods	Dev						Test					
	Retriever		Reader				Retriever		Reader			
	Recall	MRR	MRR	H-Q	H-D	F1	Recall	MRR	MRR	H-Q	H-D	F1
DrQA	0.200	0.115	N/A	0.0	0.0	4.5	0.225	0.157	N/A	0.1	0.0	6.3
BERTserini	0.266	0.177	N/A	14.1	0.2	19.3	0.251	0.178	N/A	20.4	0.1	26.0
CQR(BM25)	0.532	0.375	0.488	15.5	0.2	22.4	0.302	0.202	0.293	21.5	0.2	27.2
CQR(ANCE)	0.675	0.576	0.629	18.1	0.4	28.6	0.584	0.457	0.493	27.7	1.0	34.2
ConvDR	0.844	0.662	0.729	18.4	0.2	30.0	0.750	0.616	0.675	35.7	1.4	39.9
LWS	-	-	-	6.0	0.2	20.2	-	-	-	11.8	**1.9**	23.1
ORConvQA	0.571	0.429	0.521	17.5	0.2	26.9	0.314	0.225	0.313	24.1	0.6	29.4
CQE	0.606	0.456	0.551	-	-	27.4	0.392	0.290	0.397	-	-	30.6
CQE-hybrid	0.633	0.489	0.573	-	-	27.6	0.431	0.318	0.434	-	-	32.1
DGRCoQA†	**0.866**	**0.775**	**0.782**	**19.7**	**0.6**	**31.5**	**0.793**	**0.746**	**0.751**	**43.8**	1.6	**43.9**

We use Recall@5 and MRR@5 to evaluate the passage retrieval performance. H-Q and H-D refer to HEQ-Q and HEQ-D, respectively. Unavailable and inapplicable results are marked by "N/A" and "-". Best results in each group are marked Bold. † means statistically significant improvement over the strongest baseline with $p < 0.05$ in terms of F1.

- **LWS [31]:** LWS is a learned weak supervision approach proposed for open-retrieval conversational QA. It can identify a paraphrased span of the known answer in a retrieved passage as the weak answer, and thus is less demanding on the retriever.
- **ORConvQA [32]:** It is an open-retrieval conversational QA method. It uses a dense retriever, ranker, and reader pipeline. ORConvQA utilizes a sliding window to append previous questions.

Following the previous work [32], we evaluated all methods under the setting of no true history answers. Since the methods from conversational search and conversational rewrite only handle a part of problems of the conversational open-domain QA, we apply similar downstream components to these methods as ours for fair comparison.

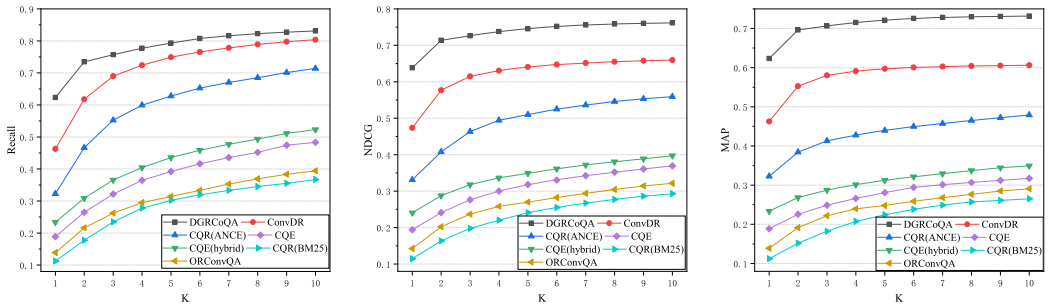
4.4 Overall Comparison

The results of all methods are summarized in Table 2, from which we have the following findings.

(1) As expected, traditional single-turn open-domain QA methods, i.e., DrQA and BERTserini, perform worse than the conversational QA, conversational question rewrite, and conversational search methods. Because they fail to solve the elliptical and coherence problems in conversations and thus cannot understand the conversational questions very well. It is noticed that LWS is a conversational open-domain QA method but worse than BERTserini. This is because LWS is a weak supervision approach, which suggests that span match weak supervision is sufficient to handle conversations with span answers [31].

(2) Regarding to the conversational question understanding, we find that ORConvQA, which heuristically appends the conversation context to the current question, performs worse than other conversational QA methods, including CQR, CQE, and ConvDR. Among them, CQE learns to incorporate the conversational question rewrite and retrieval components into an end-to-end framework. CQR and ConDR surpasses CQE, because they utilize the gold question as the supervised signal to embed a question vector and re-formulate a self-contained question text, respectively.

(3) The dense retriever based methods surpass the BM25 retriever based methods. For example, CQR(ANCE) and CQR(BM25) have the same conversational question understanding component,



(a) Recall@K on test set (b) NDCG@K on test set (c) MAP@K on test set

Fig. 5. Retrieval performance of baselines and our method versus the number of retrieved passages.

but CQR(ANCE) achieves better performance benefiting from the ANCE retriever. It is also verified that the dense retriever outperforms the heuristic ones in the single-turn QA [25, 27]. On the one hand, the semantic similarities can be learned and reflected via the distances in the embedding space. On the other hand, the dense retrievers allow joint learning of different components in the conversational open-domain QA pipeline.

(4) Our method achieves the best performance, substantially surpassing all baselines. In specific, our method has the DQI component, which not only identifies the relevant information in the conversation context but also bridges the question understanding and answer extraction. Besides, our GRR captures the answer flow and retrieves the relevant passages from the Web. It illustrates that the tight relations among answers in a conversation play a big part in the answer extraction.

To justify the performance of the retrieval passage list, we reported Recall@K, NDCG@K, and MAP@K by varying the number of retrieved passages in Figure 5. From Figure 5, we can see that when the number of returned passages increases, Recall, NDCG, and MAP rise. This is because more and more gold passages are retrieved with the increased number of returned passages. It is also noticed that our method significantly outperforms other methods in terms of the three metrics, which verifies the effectiveness of our retriever. Compared with that in terms of Recall, the performance gap between our method and baselines are larger in terms of NDCG and MAP. This is because our method rank the gold passages in the front positions than baselines. NDCG and MAP are sensitive to the rank position than Recall, therefore our DGRCoQA model surpasses the baselines more in terms of the NDCG and MAP.

4.5 Component-wise Evaluation

In this subsection, we conducted experiments to answer the following three questions: (1) Does the DQI add value to our model? (2) How much help does our model get from the GRR? And (3) Is it necessary to re-rank the passages in the Reader component.

In our DGRCoQA model, DQI incorporates an attention mechanism with the QA context to attend useful information in the conversation context. To answer the first question, we first eliminated the whole DQI component and only heuristically appended the history questions to the current question as replacement. And to verify the effectiveness of the QA context, we also only removed the returned QA context from Reader but kept the attention mechanism to see the performance. As to answer the second question, we eliminated the GRR component from our model and replaced it with a dense retriever in [49]. To answer the third question, we removed the Ranker component. These variants of our model to further verify the effectiveness of the components in our system are denoted as follows,

Table 3. Component-wise Validation of our Proposed Method by Disabling One Component Each Time

Methods	Retriever		Reader			
	Recall	MRR	MRR	H-Q	H-D	F1
w/o DQI	0.763	0.722	0.727	36.9	1.0	42.6
w/o QA context	0.777	0.725	0.731	37.4	1.2	43.1
w/o GRR	0.723	0.696	0.706	35.2	0.5	40.6
w/o Ranker	0.792	0.732	-	37.7	1.4	43.5
w/o Conversation context	0.334	0.166	0.333	25.5	0.5	31.0
DGRCoQA	**0.793**	**0.746**	**0.751**	**43.8**	**1.6**	**43.9**

— **w/o DQI:** DQI is excluded from the pipeline. That is, the question representation vector is generated via Equation (1) as described in the first round of retrieval. Since there is no dynamic question modeling, there is no multi-round retrieval process and no feedback from the Reader.
— **w/o QA context:** QA context is removed from the DQI component. Originally, the predicted candidate answer $\hat{a}_k$ is incorporated into the conversation context H_k and the current question q_k in the form of triplet, which is formulated as "[CLS] q_k [SEP] $\hat{a}_k$ [SEP] q_i [SEP]a_i [SEP]". To verify its effectiveness, the triplet is formulated as "[CLS] q_k [SEP] q_i [SEP]a_i [SEP]".
— **w/o GRR:** The GRR is excluded from the pipeline. The dense retriever mentioned before is used to replace our GRR. That is, the passage list $\mathcal{P}_{dr}$ rather than the list $\mathcal{P}_r$ is delivered to the Reader.
— **w/o Ranker:** The ranker is excluded from the pipeline. And the ranker's score S_b is removed from the final predicted score in Equation (17). **w/o Conversation context:** The conversation context $\mathcal{H}_k$ is removed from the DGRCoQA model and only the current question q_k is used.

The experimental results are summarized in Table 3. By jointly analyzing Table 3, we gained the following insights. (1) Removing the Ranker component degrades the QA performance. To be more specific, "w/o Ranker" drops by 0.4 in terms of F1. This statistic reveals the effectiveness of the ranker component, which ranks the passages with full interactions than the dot product operation. Although the ranker loss does not influence the retriever, the retriever performance also decreases. This is because that the joint training mechanism can improve all different components. (2) DGRCoQA shows the consistent improvements over "w/o DQI" and "w/o QA context". The improvements in terms of F1 are 1.3 and 0.8, respectively. Such phenomenon clearly reflects the great advantage of our dynamic question interpreter. And the gap between "w/o DQI" and "w/o QA context" further shows that QA context also plays an important role in the DQI component, because it let the question understanding can receive the feedback from the answer extraction and then re-understand the conversational questions. (3) DGRCoQA surpasses "w/o GRR". On the one hand, it indicates that our observation about the answer flow in conversational QA is helpful to extract exact answers; on the other hand, the performance gap verifies the effectiveness of the GRR component, which constructs a context graph and captures the answer flow on it. (4) By comparing the variants (except w/o conversational context, it is clear that when the GRR is excluded, the system performance drops the most. We believed that the improvement of our method mainly comes from the answer path reasoning in GRR. Meanwhile, when the Ranker is excluded, it only causes less impact on the overall performance. And (5) Removing conversation context from the

Table 4. Performance Comparison Among Different Methods for the Context Graph Construction

Methods	Retriever		Reader			
	Recall	MRR	MRR	H-Q	H-D	F1
w/o ha	0.734	0.708	0.714	36.2	1.0	41.8
w/o dr	0.276	0.268	0.281	24.2	0.5	28.7
w/o ex	0.760	0.723	0.724	37.8	1.4	43.6
-ga	**0.900**	**0.848**	**0.852**	**43.8**	**1.6**	**46.2**
DGRCoQA	0.793	0.746	0.751	40.3	**1.6**	43.9

model degrades the performance a lot, which is consistent with the experiment results in other conversational QA works [9, 32, 34]. This is because that some important entities or words are missed in the single question, and it is hard to precisely understand the user's information needs without the conversation context.

4.6 On the Graph Reasoning Enhanced Retriever

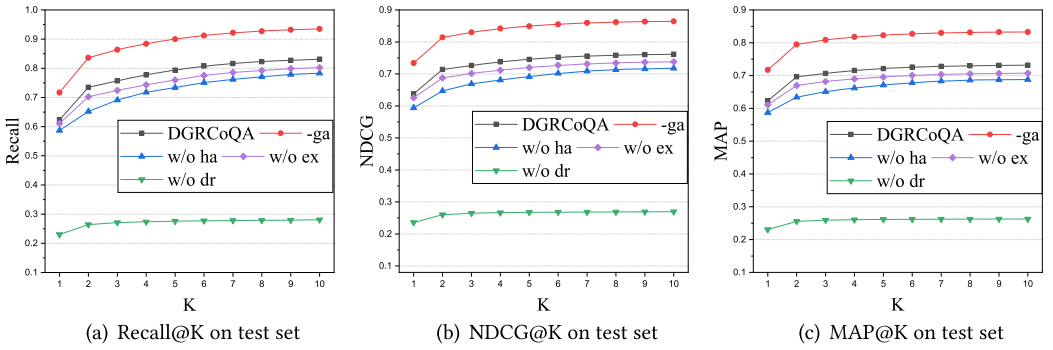
Recall that the GRR reasons the most possible answer path on a context graph to select the passage that contains the answer. GRR constructs a context graph based on the passages that contain history answers, the passages that initially dense retrieved via a dense retriever, and the passages expanded via hyperlinks. A natural question is the necessity of utilizing multi sources to construct the context graph. To answer this question, we designed some variants of our model as follows,

- **w/o ha:** The passages that contain history answers of the conversation context, denotes as $\mathcal{P}_{ha}$, are excluded from the context graph construction. We constructed the context graph based on the passages densely retrieved and expanded the set via hyperlinks.
- **w/o dr:** The passages that retrieved from the dense retriever, denotes as $\mathcal{P}_{dr}$, are removed when constructing the context graph.
- **w/o ex:** We did not expand passages via hyperlinks when constructing the context graph.
- **-ga:** We utilized the passages that contained previous gold answers rather than predicted answers to construct the context graph.

The comparison results among various context graph construction methods are displayed in Table 4. As the GRR takes responsibility for the Web retrieval, we also reported the retrieval performance versus the number of retrieved passages in Figure 6. From the results, we had the following observations:

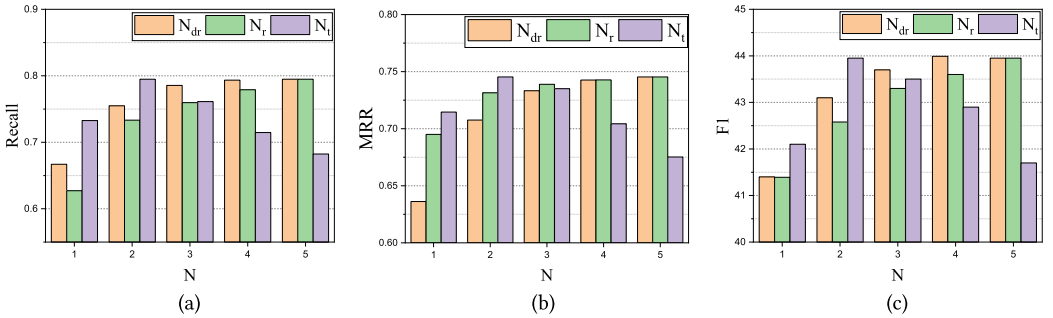
(1) It can be seen that "DGRCoQA w/o dr" is the worst variant in terms of all the used metrics. Quantitatively, retrieval performance drops from original 0.793 to 0.276 in terms of Recall and QA performance drops from 43.9 to 28.7 in terms of F1. Since the GRR cannot densely retrieve passages from the Web collection and only constructs the context graph based on the passages containing predicted history answers, the coverage of the context graph must be reduced greatly and hard to reason the gold passage. In a sense, we can regard our proposed GRR as a reasoning layer followed the previous retrieval methods and boost them by capturing the conversation flow. Its performance depends on the initial retriever but is beyond it.

(2) Both "DGRCoQA w/o ha" and "DGRCoQA w/o ex" perform worse than DGRCoQA. This phenomenon clearly illustrates that the passages containing previous predicted answers and expanded via hyperlinks are helpful to conduct a comprehensive context graph. On the one hand, the passages containing previous predicted answers are likely to be related to the current turn



(a) Recall@K on test set (b) NDCG@K on test set (c) MAP@K on test set

Fig. 6. Retrieval performance of different methods for the context graph construction versus the number of retrieved passages.



(a) (b) (c)

Fig. 7. QA performance on the test set versus the number of passages in $\mathcal{P}_{dr}$, the number of retrieved passages in $\mathcal{P}_r$, and the number of retrieval rounds, respectively.

answer and even have contained the current answer. On the other hand, although the gold passage is not covered in the initially retrieved passages, it may be included via the expansion.

And (3) it is worth mentioning that "DGRCoQA-ga" surpasses DGRCoQA. This phenomenon reflects that utilizing the passages containing previous gold answers is better than using predicted answers. It is because that the current question depends on not only the previous question but also previous answers, since user may be interested in the previous answer returned by the system and ask follow-up question about it. If the previous answers are predicted incorrectly, it has been tough to understand the questions depending on the previous answers. Therefore, utilizing gold answers can better understand questions and capture the answer flow. It is noticed that our DGRCoQA model learns to model the dependency between the current question and previous answers, which is overlooked by previous methods.

4.7 Additional Analyses

In this subsection, we quantitatively analyzed some key variables of our model, including the number of passages in $\mathcal{P}_{dr}$, the number of passages in $\mathcal{P}_r$, and the number of retrieval rounds. We reported the retrieval performance in terms of Recall@5 and MRR@5, as well as the QA performance in terms of F1, summarized in Figure 7. We then studied the impacts of the three hyper-parameters separately.

Impact of the number of passages in $\mathcal{P}_{dr}$. As mentioned in Section 3.3, the nodes of the context graph partially sources from a dense retriever. Therefore, the number of $\mathcal{P}_{dr}$, N_{dr}, is an import hyper-parameter that needs to be explored. We have conducted experiments to verify its

necessary in Section 4.6, but how the number of passages in $\mathcal{P}_{dr}$ impacts the performance is not experimented. To this end, we carried out experiments to verify the impact of the number of passages in $\mathcal{P}_{dr}$. The choices of N_{dr} are from 1 to 5, and the results in terms of Recall, MRR, and F1, are illustrated in Figures 7(a)–7(c). It is noticed if N_{dr} is set as 1, the performance greatly surpasses the performance when N_{dr} is set as 0. It demonstrates that if the passages from the dense retriever are not included to conduct the context graph, the performance drops a lot. This is because that these passages help to locate an appropriate area in the big passage collection, and thus it contributes to the retrieval of the gold passage. Besides, as the number of passages in $\mathcal{P}_{dr}$ increases, the F1 score first rises a lot and gradually increases slowly. This experimental result tells us that more irrelevant passages are also included as the N_{dr} increases.

Impact of the number of passages in $\mathcal{P}_r$. Ideally, the GRR is expected to retrieve one gold passage to the Reader to predict a text span. However, GRR is hard to rank the gold passage in the first position of the rank list. Therefore, a list of passages $\mathcal{P}_r$ rather a singe passage is delivered to Reader in practice. A natural question is that what is the optimal number of passages should be retrieved and how the number N_r influence the performance. Similarly, we conducted experiments to explore the number of passages in $\mathcal{P}_r$, and the number is set from 0 to 5. The results are summarized in Figure 7. We can observe that as the number of passages increases, the F1 score gradually rises because more and more relevant passages can be scanned by the Reader. Compared with the GRR, Reader can process the lengthy passages in a more fine-grained level and predict the text span. However, it is more GPU-memory consuming, thus we only set N_r as 5 following the previous methods [32].

Impact of the number of retrieval rounds. As claimed before, our proposed DQI repeats N_t rounds to dynamically attend the conversation context, thus it is worth to exploit the number of retrieval rounds. However, due to the limitation of the GPU memory, we only tuned this value when evaluating the system. The results are shown in Figure 7. It is observed that as the N_t increases, the performance rises and then gradually degrades. This reflects that increasing the retrieval rounds does not improve the performance continuously. On the one hand, we did not tune the number of retrieval rounds in the training process. On the other hand, multi-round retrieval may bring more noise and probably suffer from the error propagation problem.

Besides, we reported the training and test time for our model and other available models. In the training phase, our model took about 216 minutes every epoch. And the time for ORConvQA and ConvDR is 178 minutes and 144 minutes, respectively. In the test phase, it took about 0.26s for our model to answer one question, while the time for ORConvQA, CQR(BM25), and CQR(ANCE) is 0.21s, 0.13s, 0.22s, and 0.24s, respectively. The time consuming of our model is bigger than others, which is because that the graph reasoning based retriever and multi-round mechanism naturally take more time.

4.8 Case Study

To gain deeper insights into the performance of our proposed method in conversational open-domain QA, we listed some examples in Table 5. Each example includes the current question, history questions, and attention weights calculated in Equation (3). To illustrate our proposed DQI and GRR, we also listed the labels of the retrieved passages from the GRR of the first round and second round. From examples in Table 5, we had the following observations.

Current questions. (1) Analyzing the current questions, we found that the coreference problem is common. For example, the pronoun "he" in question "Did he play any live shows?" refers to the name "Hank Snow" in the first history question. (2) It is also observed that there may be two coreferences in one question. For example, for the question "Did he accept and appear on the show", "he" and "the show" refer to "Sakis Rouvas" and "the first tv appearance" in the history

Table 5. Case Study

Current question	History questions	Attention Weights	1-round Retrieval	2-round Retrieval
Did he play any live shows?	What led Hank Snow to Nashville?	0.4015	[1 0 0 0 0]	[1 0 0 0 0]
	Where did he get his start in Nasville?	0.1832		
	What was his most popular song?	0.2073		
	Did he win any awards	0.2080		
Why did he burn it?	Did Varg Vikernes commit any arson?	0.5733	[0 0 0 1 0]	[1 0 0 0 0]
	When was the first case?	0.4267		
Did he accept and appear on the show?	What was Sakis Rouvas' first tv appearance?	1	[0 0 0 0 0]	[1 0 0 0 0]

The test samples of our proposed method. Attention weights refer to the attention score computed in Equation (3). 1-round Retrieval, 2-round Retrieval refer to the labels of the top-5 retrieved passages from GRR in the first round of Retrieval and the second round of Retrieval, respectively.

question, respectively. (3) Moreover, the current question may depend on the history answers. For the second question "Why did he burn it?", we cannot clarify what the word "it" means only based on the history questions, and which turns out to be "the Fantoft Stave Church" in the previous answer "On 6 June 1992, the Fantoft Stave Church, dating from the 12th century and considered architecturally significant, was burned". This observation is consistent with our experiment results that DGRCoQA-ga largely surpass DGRCoQA without the previous true answers.

Attention weights. (1) By jointly analyzing the history questions and attention weights, we found that the attention weights approximately reflect the importance of history questions. For example, in the first and second example, the first history question obtains the largest attention score. This adheres to our intuition because the first history question usually mentions the topic of the conversation. (2) It surprises us that the second history question of the second example also achieves a comparative attention score. As claimed before, in the second example, the gold passage and answer of the second history question provide important information to help understand the current question. Therefore, a comparative attention score becomes reasonable because it makes it easier to retrieve the corresponding evidence to clarify the current question. (3) The attention weights are not significantly discriminative. For example, in the first example, the attention weight of the fourth history question is 0.2080 although it may only provide little of useful information. This is partly due to the fact the questions in a conversation are always related more or less, since they usually have a same or similar topic.

Retrieval result. Analyzing the results of 1-round of retrieval and 2-round of retrieval, we found that 2-round of retrieval can make a supplement for the 1-round of retrieval in some cases. For example, in the third example, 1-round of retrieval does not retrieve the gold passage but 2-round of retrieval does. This may illustrate our dynamically multi-round retrieval method works partly due to its ability of handling the complex questions. And this observation is consistent with some recent studies [14, 46] that focus on applying the multi-round retrieval method to answer the complex questions in open-domain QA.

5 CONCLUSION AND FUTURE WORK

In this work, we explore a rarely studied but practical task, conversational open-domain QA. Different from the previous conversational QA tasks, conversational open-domain QA requires to not only understand conversational questions but also extract an exact answer from the Web. To build a robust conversational open-domain QA system, we pay attention to the three issues: (1) precisely understand conversational questions with the conversation context; (2) extract exact answers by capturing the answer dependency and transition flow in a conversation; and (3) deeply

integrate question understanding and answer extraction. We propose the DGRCoQA model to handle the three issues, which includes DQI, GRR, and Reader. As far as we know, this is the first work in conversational QA that models the answer relations explicitly on a graph. To justify our method, we perform extensive experiments on the public dataset, and the experimental results demonstrate the effectiveness of our model.

In future, we plan to deepen and widen our work from the following aspects: (1) In this work, we utilized the hyperlinks in Wikipedia to construct the context graph. Due to the practical concern that once passages are not from Wikipedia, the hyperlinks among passages are hard to obtain. We will extend our method by introducing the **knowledge graphs** (**KGs**). On the one hand, the KGs help to conduct the graph structure; on the other hand, the edges usually contain extra information than links and can be utilized to boost the system. (2) The OR-QuAC dataset is from the conversational MRC area. There are still some differences between conversational open-domain QA and conversational MRC. Thus, it is necessary to conduct a dataset more suitable for the real conversational open-domain QA, where people ask questions for the purpose of their real information need. And (3), as shown in the Case Study, there are still complex questions in conversational QA. We will focus on these complex questions in our future work.

REFERENCES

[1] Raviteja Anantha, Svitlana Vakulenko, Zhucheng Tu, Shayne Longpre, Stephen Pulman, and Srinivas Chappidi. 2021. Open-domain question answering goes conversational via question rewriting. In *Proceedings of the International Conference of the North American Chapter of the Association for Computational Linguistics*. 520–534.

[2] Akari Asai, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. 2020. Learning to retrieve reasoning paths over wikipedia graph for question answering. In *Proceedings of the International Conference on Learning Representations*.

[3] Wei-Cheng Chang, X. Yu Felix, Yin-Wen Chang, Yiming Yang, and Sanjiv Kumar. 2019. Pre-training tasks for embedding-based large-scale retrieval. In *Proceedings of the International Conference on Learning Representations*.

[4] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*. ACL, 1870–1879.

[5] Danqi Chen and Wen-tau Yih. 2020. Open-domain question answering. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*. ACL, 34–37.

[6] Tong Chen, Hongzhi Yin, Guanhua Ye, Zi Huang, Yang Wang, and Meng Wang. 2020. Try this instead: Personalized and interpretable substitute recommendation. In *Proceedings of the International Conference on Research and Development in Information Retrieval*. ACM, 891–900.

[7] Yifan Chen, Yang Wang, Pengjie Ren, Meng Wang, and Maarten de Rijke. 2021. Bayesian feature interaction selection for factorization machines. *Artificial Intelligence* 302, 0004-3702 (2021), 103589.

[8] Yu Chen, Lingfei Wu, and Mohammed J. Zaki. 2020. GraphFlow: Exploiting conversation flow with graph neural networks for conversational machine comprehension. In *Proceedings of the International Joint Conference on Artificial Intelligence*. AAAI, 1230–1236.

[9] Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wen-tau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. QuAC: Question answering in context. In *Proceedings of the International Conference on Empirical Methods in Natural Language Processing*. ACL, 2174–2184.

[10] Philipp Christmann, Rishiraj Saha Roy, Abdalghani Abujabal, Jyotsna Singh, and Gerhard Weikum. 2019. Look before you hop: Conversational question answering over knowledge graphs using judicious context expansion. In *Proceedings of the International Conference on Information and Knowledge Management*. ACM, 729–738.

[11] W. Bruce Croft. 2019. The importance of interaction for information retrieval. In *Proceedings of the International Conference on Research and Development in Information Retrieval*. ACM, 1–2.

[12] W. Bruce Croft and Roger H. Thompson. 1987. I3R: A new approach to the design of document retrieval systems. *Journal of the American Society for Information Science* 38, 6 (1987), 389–404.

[13] Jeffrey Dalton, Chenyan Xiong, Vaibhav Kumar, and Jamie Callan. 2020. Cast-19: A dataset for conversational information seeking. In *Proceedings of the International Conference on Research and Development in Information Retrieval*. ACM, 1985–1988.

[14] Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, and Andrew McCallum. 2018. Multi-step retriever-reader interaction for scalable open-domain question answering. In *Proceedings of the International Conference on Learning Representations*.

[15] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the International Conference of the North American Chapter of the Association for Computational Linguistics*. ACL, 4171–4186.

[16] Ahmed Elgohary, Denis Peskov, and Jordan Boyd-Graber. 2019. Can you unpack that? Learning to rewrite questions-in-context. In *Proceedings of the International Conference on Empirical Methods in Natural Language Processing*. ACL, 5920–5926.

[17] Ahmed Elgohary, Chen Zhao, and Jordan Boyd-Graber. 2018. A dataset and baselines for sequential open-domain question answering. In *Proceedings of the International Conference on Empirical Methods in Natural Language Processing*. ACL, 1077–1083.

[18] Yair Feldman and Ran El-Yaniv. 2019. Multi-hop paragraph retrieval for open-domain question answering. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*. ACL, 2296–2309.

[19] Jianfeng Gao, Chenyan Xiong, and Paul Bennett. 2020. Recent advances in conversational information retrieval. In *Proceedings of the International Conference on Research and Development in Information Retrieval*. ACM, 2421–2424.

[20] Zan Gao, Leming Guo, Weili Guan, An-An Liu, Tongwei Ren, and Shengyong Chen. 2020. A pairwise attentive adversarial spatiotemporal network for cross-domain few-shot action recognition-R2. *IEEE Transactions on Image Processing* 30, 1 (2020), 767–782.

[21] Zan Gao, Yin-ming Li, Wei-li Guan, Wei-zhi Nie, Zhi-yong Cheng, and An-an Liu. 2020. Pairwise view weighted graph network for view-based 3d model retrieval. In *Proceedings of the International Conference on Research and Development in Information Retrieval*. ACM, 129–138.

[22] Zan Gao, Yanbo Liu, Guangpin Xu, and Xianbin Wen. 2021. Pairwise attention network for cross-domain image recognition. *Neurocomputing* 453, C (2021), 393–402.

[23] Heyan Huang, Xiaochi Wei, Liqiang Nie, Xianling Mao, and Xin-Shun Xu. 2018. From question to text: Question-oriented feature attention for answer selection. *ACM Transactions on Information Systems* 37, 1 (2018), 1–33.

[24] Hsin-Yuan Huang, Eunsol Choi, and Wen-tau Yih. 2018. FlowQA: Grasping flow in history for conversational machine comprehension. In *Proceedings of the International Conference on Learning Representations*.

[25] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the International Conference on Empirical Methods in Natural Language Processing*. ACL, 6769–6781.

[26] Vaibhav Kumar and Jamie Callan. 2020. Making information seeking easier: An improved pipeline for conversational search. In *Proceedings of the International Conference on Empirical Methods in Natural Language Processing: Findings*. ACL, 3971–3980.

[27] Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*. ACL, 6086–6096.

[28] Sheng-Chieh Lin, Jheng-Hong Yang, and Jimmy Lin. 2021. Contextualized query embeddings for conversational search. *CoRRabs/2104.08707* (2021).

[29] Piero Molino, Luca Maria Aiello, and Pasquale Lops. 2016. Social question answering: Textual, user, and network features for best answer prediction. *ACM Transactions on Information Systems* 35, 1 (2016), 1–40.

[30] Aditya Pal, F. Maxwell Harper, and Joseph A. Konstan. 2012. Exploring question selection bias to identify experts and potential experts in community question answering. *ACM Transactions on Information Systems* 30, 2 (2012), 1–28.

[31] Chen Qu, Liu Yang, Cen Chen, W. Bruce Croft, Kalpesh Krishna, and Mohit Iyyer. 2021. Weakly-supervised open-retrieval conversational question answering. In *Proceedings of the International Conference on European Conference on Information Retrieval*. Springer, 425–434.

[32] Chen Qu, Liu Yang, Cen Chen, Minghui Qiu, W. Bruce Croft, and Mohit Iyyer. 2020. Open-retrieval conversational question answering. In *Proceedings of the International Conference on Research and Development in Information Retrieval*. ACM, 539–548.

[33] Chen Qu, Liu Yang, Minghui Qiu, Yongfeng Zhang, Cen Chen, W. Bruce Croft, and Mohit Iyyer. 2019. Attentive history selection for conversational question answering. In *Proceedings of the International Conference on Information and Knowledge Management*. ACM, 1391–1400.

[34] Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics* 7, 0 (2019), 249–266.

[35] Amrita Saha, Vardaan Pahuja, Mitesh Khapra, Karthik Sankaranarayanan, and Sarath Chandar. 2018. Complex sequential question answering: Towards learning to converse over linked question answer pairs with a knowledge graph. In *Proceedings of the AAAI Conference on Artificial Intelligence*. AAAI, 705–713.

[36] Tao Shen, Xiubo Geng, QIN Tao, Daya Guo, Duyu Tang, Nan Duan, Guodong Long, and Daxin Jiang. 2019. Multi-task learning for conversational question answering over a large-scale knowledge base. In *Proceedings of the International Conference on Empirical Methods in Natural Language Processing*. ACL, 2442–2451.

[37] Svitlana Vakulenko, Shayne Longpre, Zhucheng Tu, and Raviteja Anantha. 2021. Question rewriting for conversational question answering. In *Proceedings of the International Conference on Web Search and Data Mining*. ACM, 355–363.
[38] Maria Vargas-Vera and Miltiadis D. Lytras. 2010. Aqua: A closed-domain question answering system. *Information Systems Management* 27, 3 (2010), 217–225.
[39] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention Networks. In *Proceedings of the International Conference on Learning Representations*.
[40] Nikos Voskarides, Dan Li, Pengjie Ren, Evangelos Kanoulas, and Maarten de Rijke. 2020. Query resolution for conversational search with limited supervision. In *Proceedings of the International Conference on Research and Development in Information Retrieval*. ACM, 921–930.
[41] Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerry Tesauro, Bowen Zhou, and Jing Jiang. 2018. R3: Reinforced ranker-reader for open-domain question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*. AAAI, 5981–5988.
[42] Shuohang Wang, Mo Yu, Jing Jiang, Wei Zhang, Xiaoxiao Guo, Shiyu Chang, Zhiguo Wang, Tim Klinger, Gerald Tesauro, and Murray Campbell. 2018. Evidence aggregation for answer re-ranking in open-domain question answering. In *Proceedings of the International Conference on Learning Representations*.
[43] Yang Wang. 2021. Survey on deep multi-modal data analytics: Collaboration, rivalry, and fusion. *ACM Transactions on Multimedia Computing, Communications, and Applications* 17, 1s (2021), 1–25.
[44] Yang Wang, Lin Wu, Xuemin Lin, and Junbin Gao. 2018. Multiview spectral clustering via structured low-rank matrix factorization. *IEEE Transactions on Neural Networks and Learning Systems* 29, 10 (2018), 4833–4843.
[45] Yinwei Wei, Xiang Wang, Liqiang Nie, Xiangnan He, Richang Hong, and Tat-Seng Chua. 2019. MMGCN: Multi-modal graph convolution network for personalized recommendation of micro-video. In *Proceedings of the International Conference on Multimedia*. ACM, 1437–1445.
[46] Wenhan Xiong, Xiang Lorraine Li, Srini Iyer, Jingfei Du, Patrick Lewis, William Yang Wang, Yashar Mehdad, Wentau Yih, Sebastian Riedel, Douwe Kiela, and Barlas Oguz. 2020. Answering complex open-domain questions with multi-hop dense retrieval. In *Proceedings of the 9th International Conference on Learning Representations* (2020).
[47] Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. 2019. End-to-end open-domain question answering with BERTserini. In *Proceedings of the International Conference of the North American Chapter of the Association for Computational Linguistics*. ACL, 72–77.
[48] Shi Yu, Jiahua Liu, Jingqin Yang, Chenyan Xiong, Paul Bennett, Jianfeng Gao, and Zhiyuan Liu. 2020. Few-shot generative conversational query rewriting. In *Proceedings of the International Conference on Research and Development in Information Retrieval*. ACM, 1933–1936.
[49] Shi Yu, Zhenghao Liu, Chenyan Xiong, Tao Feng, and Zhiyuan Liu. 2021. Few-shot conversational dense retrieval. In *Proceedings of the International Conference on Research and Development in Information Retrieval*. ACM, 829–838.
[50] Haoyu Zhang, Meng Liu, Zan Gao, Xiaoqiang Lei, Yinglong Wang, and Liqiang Nie. 2021. Multimodal dialog system: Relational graph-based context-aware question understanding. In *Proceedings of the International Conference on Multimedia*. ACM, 695–703.
[51] Richong Zhang, Yue Wang, Yongyi Mao, and Jinpeng Huai. 2019. Question answering in knowledge bases: A verification assisted model with iterative training. *ACM Transactions on Information Systems* 37, 4 (2019), 1–26.
[52] Xiao Zhang, Meng Liu, Jianhua Yin, Zhaochun Ren, and Liqiang Nie. 2021. Question tagging via graph-guided ranking. *ACM Transactions on Information Systems* 40, 1 (2021), 1–23.

Received July 2021; revised October 2021; accepted November 2021