electronics

MDPI

Article

Graph Convolution Network over Dependency Structure Improve Knowledge Base Question Answering

Chenggong Zhang [1,2,*], Daren Zha [2], Lei Wang [2], Nan Mu [2], Chengwei Yang [3], Bin Wang [4] and Fuyong Xu [4,*]

[1] Institute of School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100043, China
[2] Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100864, China; zhadaren@iie.ac.cn (D.Z.); wanglei@iie.ac.cn (L.W.); munan@iie.ac.cn (N.M.)
[3] School of Management Science and Engineering, Shandong University of Finance and Economics, Jinan 250014, China; yangchengwei2006@163.com
[4] School of Information Science and Engineering, Shandong Normal University, Jinan 250358, China; wang_bean_068@163.com
* Correspondence: zcg870108@163.com (C.Z.); fyxu0908@outlook.com (F.X.)

Abstract: Knowledge base question answering (KBQA) can be divided into two types according to the type of complexity: questions with constraints and questions with multiple hops of relationships. Previous work on knowledge base question answering have mostly focused on entities and relations. In a multihop question, it is insufficient to focus solely on topic entities and their relations since the relation between words also contains some important information. In addition, because the question contains constraints or multiple relationships, the information is difficult to capture, or the constraints are missed. In this paper, we applied a dependency structure to questions that capture relation information (e.g., constraint) between the words in question through a graph convolution network. The captured relation information is integrated into the question for re-encoding, and the information is used to generate and rank query graphs. Compared with existing sequence models and query graph generation models, our approach achieves a 0.8–3% improvement on two benchmark datasets.

Keywords: dependency structure; graph convolution network; question answering

Citation: Zhang, C.; Zha, D.; Wang, L.; Mu, N.; Yang, C.; Wang, B.; Xu, F. Graph Convolution Network over Dependency Structure Improve Knowledge Base Question Answering. *Electronics* **2023**, *12*, 2675. https://doi.org/10.3390/electronics12122675

Academic Editor: Ping-Feng Pai

Received: 20 April 2023
Revised: 1 June 2023
Accepted: 2 June 2023
Published: 14 June 2023

1. Introduction

The rapid development of information technology has created a need to accurately extract information from large-scale data, making question answering (QA) systems an important area of research. In the 1960s, QA systems primarily relied on expert systems, which involved numerous rules or templates. As technology advanced, QA systems shifted towards information-retrieval-based approaches. Retrieval-based QA systems rely on keyword matching and information extraction to analyze surface-level meaning and to extract answers from relevant documents. However, these systems can only provide answers to predefined questions.

To overcome this limitation, large-scale commercial engines have been developed. Community-based QA systems, which are built upon keyword matching retrieval, utilize historical questions from users and recommend answers to new questions. In recent years, the growth of the World Wide Web has led to the accumulation of vast amounts of high-quality data. This has paved the way for the emergence of extensive knowledge bases (KBs) that contain structured data. Natural language questions can be mapped to structured queries on these knowledge bases. KBQA (knowledge base question answering) aims to correctly understand the semantics of user questions and to use fact retrieval, matching, and reasoning techniques within the knowledge base to find answers.

In summary, as information technology continues to advance, QA systems have evolved from rule-based expert systems to retrieval-based approaches. With the availability of large-scale knowledge bases, KBQA systems have emerged to effectively understand

user questions and to provide accurate answers through fact retrieval and reasoning within the knowledge base. The main process of KBQA is shown in Figure 1.

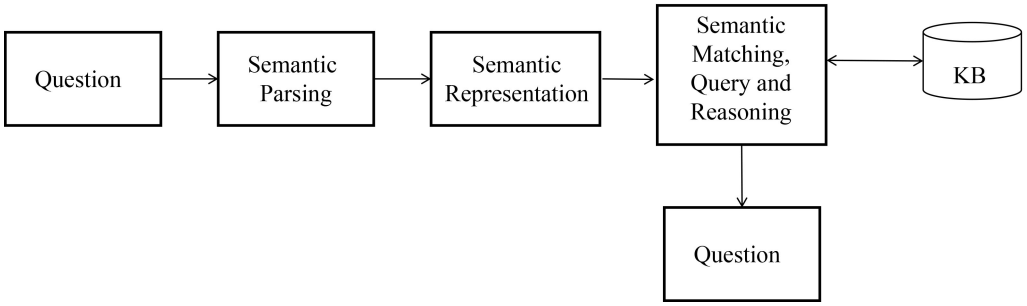

Figure 1. Main process of KBQA.

A knowledge base (KB) stores many complex structured information sets commonly represented by triples (entity, entity, and the relations between them). The task of knowledge base question answering (KBQA) is to answer the users' natural language questions using a knowledge base. For example, as shown in Figure 2, the triple starring (Jackie Chan, New Fist of Fury), release date (New Fist of Fury, 8 July 1976), and directed by (New Fist of Fury, Lo wei) can be used to answer the question "Who was the director of Jackie Chan's first starring film?".

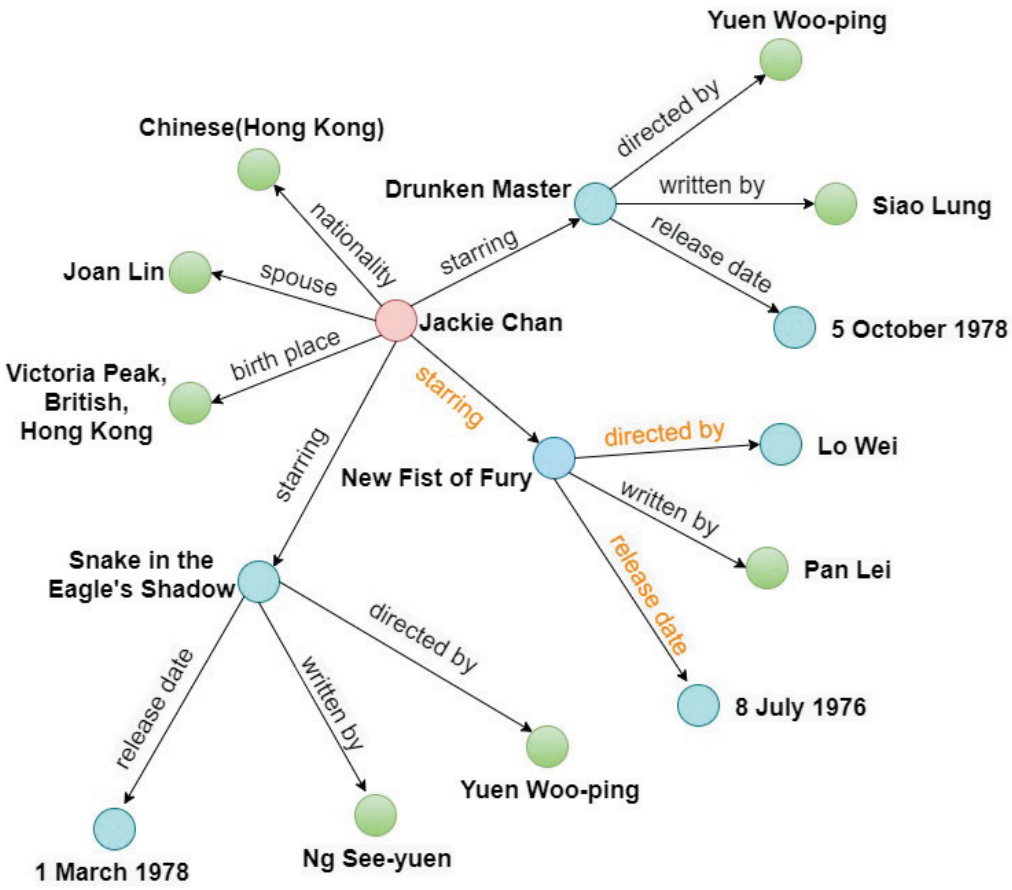

Figure 2. Regarding the triples involved in the question "Who was the director of Jackie Chan's first starring film?" in the knowledge graph; bold letters represent entities, pink circles represent topic entities, blue circles represent traversed entities, green circles represent irrelevant entities, and orange letters represent critical paths.

Previous work [1–3] on KBQA mainly focused on external resources, pattern matching, or the construction of handcrafted features [4,5] to address simple questions. These methods need labeled logical supervision. However, these methods have difficulty dealing with complex questions containing constraints, e.g., "the first" in the question "Who is the first president of the United States".

To address constraints in natural language questions, staged query graph generation methods [6–8] have been proposed. These methods first identify the single-hop relation path and then add constraints to the relation path from a query graph. The reply is obtainable by executing the query graph in the knowledge base. However, in reality, there are questions of not only single relations but also multihop relations, such as "Who is the wife of the founder of Facebook?" There are two hops between the answer and "Facebook", namely, "wife" and "founder". To answer this type of question, the longer relation path has to be considered, which will increase the search space exponentially. The beam search method was introduced by References [9,10] to reduce the search space by considering the best matching relation to reduce the number of multihop relation paths. Lan et al. [11] proposed modifying the staged query graph generation method to deal with longer relation paths and large search spaces. However, allowing a longer relationship path causes constraints to be ignored or connected with the wrong entity, resulting in errors in the prediction of the intermediate relationships. If the prediction of the intermediate relationship is wrong, the subsequent prediction will also be wrong. In query graph generation's operation, it is therefore particularly significant to analyze the relation between words.

A dependency tree can help the model capture the long-distance relationship between words. Models that use the dependency parses [12,13] have been demonstrated to be very effective in relationship extraction, since they capture long-distance semantic relations. Multihop questions generally contain constraints and multiple relations. For example, for the query "What posts did John Adams hold before he was president?", the constraint is "before", and the answer is related to "John Adams" via two hops, namely, "president" and "job". To solve this situation, the relations between words need to be focused on to reach the correct answers. We use the dependency analysis of the input question to assist the model in selecting relations. An efficient graph convolution operation [14] was used to encode the input question's dependency structure to extract the entity-centered representation.

In this paper, to focus on the relationship between words and the constraints in a question due to a long relation path, we propose a dependency structure for a question based on a graph convolution network (GCN), which encodes the dependency formation above the input query with efficient graph convolution actions to improve the attention paid to the constraints in a question and then guides the actions of the query graph generation and final ranking. This study makes three research contributions:

- For underutilization of the relationships between words in the question, we propose a question answering method on a knowledge base by applying GCNs, which permits it to efficiently pool information above arbitrary dependency formations and to produce a more effective sequence vector representation.
- For the problem of an incorrect relation selection in the process of query graph generation, we analyze the dependency structure to establish the relation between words and use the structure to obtain a more effective representation to further affect the ranking and action selection of the query graph.
- On the WebQuestionsSP (WQSP) and ComplexQuestions (CQ) datasets, our method performs well, and it is more effective in ranking query graphs.

The remainder of this paper is organized as follows. Related work about KBQA is introduced in Section 2. Section 3 describes the proposed methods in this paper. Section 4 introduces the experiments and shows the results in this paper. Section 5 concludes this paper and provides suggestions about KBQA.

2. Related Work

The current approaches that are proposed to deal with the KBQA task can be approximately classified into two categories: semantic parsing (SP) and embedding-based approaches [15,16]. These systems [17,18] are effective and provide an in-depth explanation of the query, but they need reinforcement learning or expensive data annotations. However, most SP-based approaches rely on aspects or handcrafted rules that limit their scalability and transferability.

Recently, embedding-based methods [19,20] for KBQA have become increasingly popular. Unlike SP-based methods, embedding-based approaches first allocate competitors from the KG, depicting these competitors as distributed representations, and then choose and rank these representations. Some embedding-based models directly predict solutions [21,22], while others concentrate on separating relation trails and require further procedures to obtain an answer [7,23]. Our method follows the same procedure as embedding-based models and regards query graph generation as a multistep relation path extraction process. References [9,10,24] proposed considering better relations. Lan et al. (2020) [11] proposed modifying a query graph generation process from longer relations. However, the current method is defective in its action accuracy for query graph generation. Extending the relationship path and allowing for longer relationship paths means increasing intermediate relationships, and the information in the question may be omitted. Therefore, capturing the relationship between words is particularly important in the process of forming query graphs because it affects whether the information in the question is fully utilized.

Our work also uses a dependency structure to help model the captured relations between words. A dependency tree can help the relation extraction model capture the long-distance relations between words. One common approach [12,13] is exploiting structure features on parsed tree below the lowest common ancestor (LCA).

Our method is based on the existing query graph generation process method. We add a dependency structure to the query to obtain the relation between the words and to further improve the attention paid to the constraints in a question. Compared with previous methods, we introduce the dependency structure of the question and analyze it through a graph convolution network to focus more attention on the constraints. In summary, to obtain a more effective representation, a graph convolution network is used, which allows for efficiently pooling information from an arbitrary dependency structure to achieve an effective action and to increase the accuracy of the intermediary relation selection in the query graph generation process.

3. Method

3.1. Query Graph Generation

Formally, our method followed Lan et al. (2020) [11], which is an extension of the existing staged query graph generation method. We use beam search to iteratively generate candidate query graphs. The grounded entity represents the existing entity in the knowledge base. The existential variable and lambda variable are ungrounded entities, where the lambda variable represents the answer. Finally, the aggregate function is used to perform function operations on specific entities, which usually captures some numerical features.

We assume that a set of query graphs is generated after the $k-th$ iteration, denoted as G_k. At the $k+1$ iteration, we apply, extend, connect, and aggregate (the details are shown in Figure 3) actions to grow G_k by one more edge and nodes. The extended action is used to extend the core relation path by finding the relation. The action of the connection is to find other grounded entities in the question and to connect them to the existing nodes. We denote G'_{k+1} as the resulting query graph. After each iteration, a large number of query graphs with applied actions will be generated. We use graph convolutional networks (explained in Section 3.2) to select query graphs that use the correct action, which will affect their scores.

Then, we describe how the query graph is generated. At every iteration, the actions {*extend*, *connect*, *aggregate*} will apply to query graph candidates. As shown in Figure 3, we show how the three actions act on the query graph (in fact, there is no sequence for the three actions) for the question "Who was the director of Jackie Chan's first starring film?". First, in query graph (a), starting from a grounding entity "Jackie Chan", a core relation path is found to connect entities and answers. If there are no redundant constraint words and other relations, the answer is x. However, because the question contains other relations, query graph (b) applied an extended action to extend the core relation path. The query graph (c) applies a connection action to find other grounded entities in the question and connects them to the existing nodes. The query graph (d) applies an aggregate action to add constraint nodes to the grounded entity or existential variable.

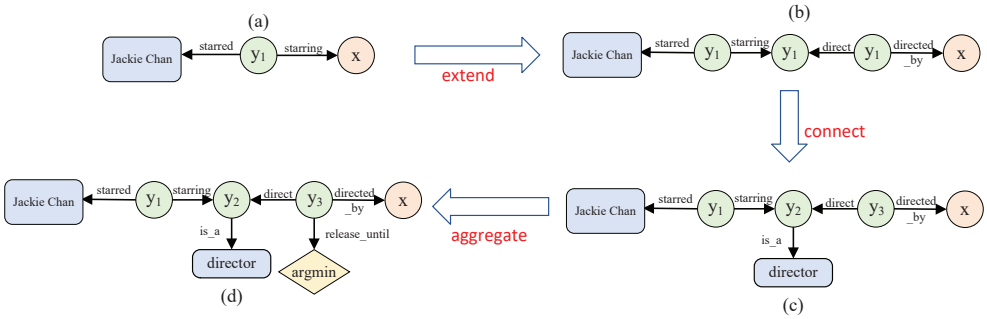

Figure 3. A possible sequence of the graph generation for "Who was director of Jackie Chan's first starring film?" Note that (**b**–**d**) are the results of the extend, connect and aggregate actions, respectively.

In practice, the order of each action is not fixed, so several potential query graphs will be generated. It is very important to select the correct action sequence and to determine the correct query graph. This will affect the correctness of the final result query graph because query graph candidates may contain intermediate relations and incorrect entities. Following our intuition mentioned in the first subsection, to enhance the generation of the query graph and to improve the accuracy of the intermediate relations, we employ the dependency structure of the input question.

3.2. Dependency Structure of a Question Based on a GCN

The dependency structure helps models capture the relations between words. First, we represent the input question as a dependency structure. An example is shown in Figure 4 (here, we set it as an undirected graph). We can see that the "film" is related to "starring", "first", "Chan", etc. The words of each neighboring node are related. The process is shown in Figure 5. First, we convert query graph g into a sequence of tokens g_t. We represent an input question $Q = \{q_i\}_{i=1}^{\{Q\}}$ as a sequence of word embeddings q_i. Then, we use BERT (language model) [25] to encode the concat of the question and query graph as h_q, which is the sequence of the hidden states.

GCN [26,27] is an adaptation of the convolutional neural network for encoding graphs. Given a graph with n nodes. We employ the convolution action to obtain the dependency trees. In a GCN with an l-layer, we represent the input vector of the i-th node of the l-th layer as $h_i^{(l-1)}$, and the output vector is expressed as h_i^l. In addition, a normalization operation is performed before the data are transferred into the nonlinear layer, and self-circulation is added to each node in the graph. The convolution action can be formulated as follows:

$$h_i^{(l)} = pool(\sigma(\sum_{j=1}^{n} \bar{A} W^{(l)} h_j^{(l-1)} / d_i + b^{(l)})) \quad (1)$$

This operation is superimposed onto Layer l to obtain a deep GCN network, where we set $h_1^{(0)}, ..., h_n^{(0)}$ to be the input word vector obtained by BERT and $h_1^{(L)}, ..., h_n^{(L)}$ as the output word representations. All operations can be efficiently applied through matrix multiplication, making the method suitable for batch computing and running on a GPU. Thus far, we have obtained the question representation containing the relation between words, which is used to affect the selection of the relations in the ranking of the query graph. In addition, the representation also captures the edge information needed by the selection relation.

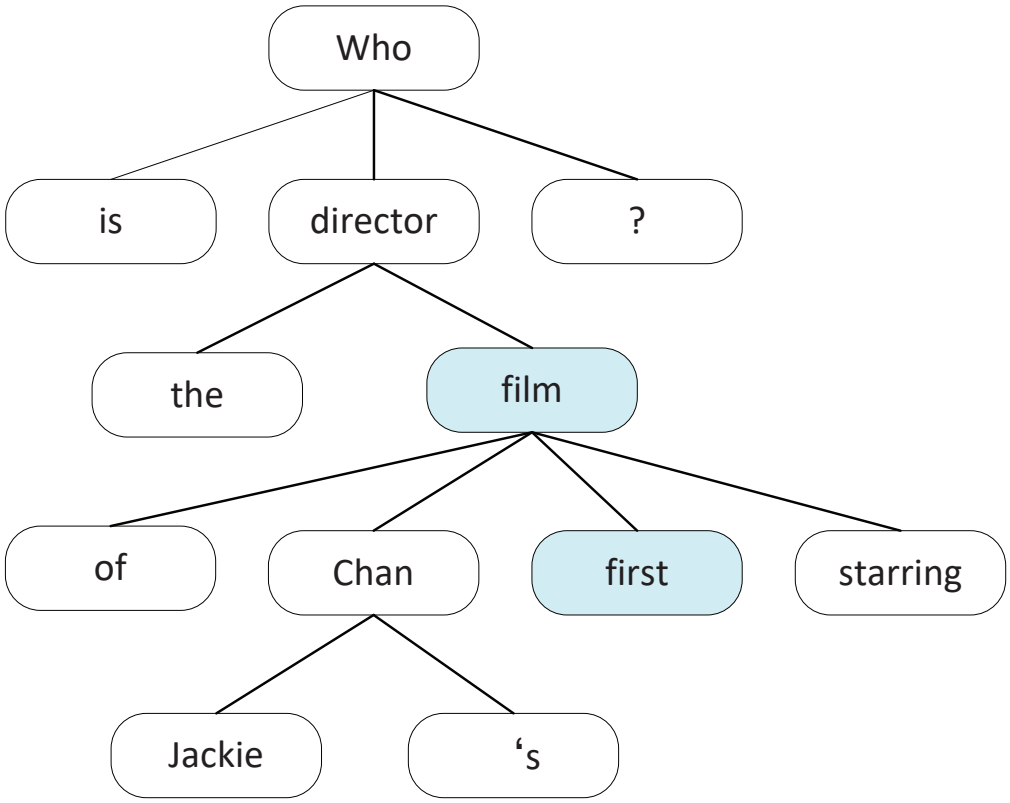

Figure 4. The dependency structure of the question “Who was the director of Jackie Chan’s first starring film?” We treat the dependency graph as undirected.

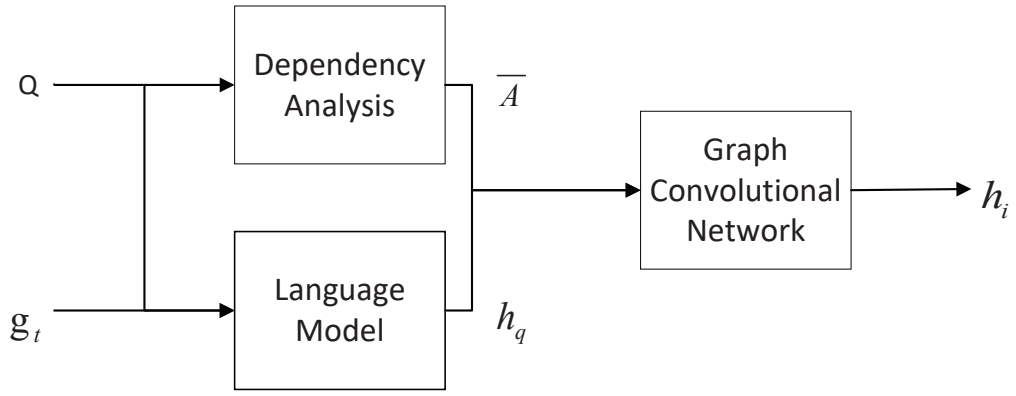

Figure 5. Overview of the dependency structure of a question based on a GCN.

3.3. Query Graph Ranking

After each query graph extension, we need to rank the candidate query graphs $g \in G_t'$, which follows the sequence of operations taken by the construction g. For example, the query graph (a) in Figure 3 is expressed as (Jackie Chan, starred, starring). Before the ranking, we integrate the information extracted from the graph convolution network into the question vector:

$$v_x = h_q + h^{(l)} \tag{2}$$

$$v_q = MLP(v_x) \tag{3}$$

where h_q is the question vector, $h^{(l)}$ is the output vector from the GCN, and MLP(·) denotes an MLP layer. Then, we derive a vector v_g for each graph and put it into FFN. Finally, we calculate the probability with softmax.

3.4. Learning

Without any correct query graph, we use question–answer pairs to train our model. Inspried by Das et al. (2018) [28], we use a RL (reinforcement learning) algorithm to obtain $p_\theta(g\|v_q)$ so that the query graph can fit the problem better. θ is the learnable parameters. As our focus is not on the model's optimization approach, but on a novel graph's application based method to KBQA, the procedure of model learning and RL exploration is not described in detail.

4. Experiments

4.1. Datasets and Settings

WebQuestionsSP(WQSP) [8] WebQuestionsSP includes 5810 train samples. WQSP annotates SPARQL query statements for each answer and removes some questions with ambiguities, unclear intentions or no clear answer. WebQuestionsSP was created for the task of question answering over structured data, specifically targeting Freebase, a large knowledge base. Each sample in WebQuestionsSP is associated with a SPARQL query statement that retrieves the answer from Freebase. To ensure the quality and clarity of the dataset, certain questions that had ambiguities, unclear intentions, or no clear answer were removed during the annotation process. This helps to maintain a reliable and focused dataset for training and evaluating question answering models. The statistic of WQSP is shown as Table 1.

Table 1. The QA pair distributions of WebQuestionsSP (WQSP) ComplexQuestions (CQ) dataset.

	WQSP	**CQ**
Total QA pairs	4737	2100
Training set QA pairs	3098	1300
Test set QA pairs	1639	800

ComplexQuestions(CQ) [6] is used to increase the complexity of the question. On the basis of webquestions, complex questions introduce the constraint types, explicit or implicit time constraints, multi-entity constraints, and aggregate class constraints (the sum of the maximum value) and provide the logical form of a query.

We need to first discover the entities in the query and then link them to the corresponding entities in the KB. We use the existing tools to learn the linking model by training questions and their answers. For superlative linking and temporal expressions, we apply a superlative word list and regular expressions simply. We use a pretrained BERT vector to initialize the word embedding with a size of 768. We set the dropout ratio and the size of the hidden layer to 0.1 and 768, respectively.

4.2. Experimental Results and Comparison

Better results are achieved by our method on the two datasets, as shown in Table 2. The performance of our method on WQSP achieves an F1 of 74.8. Our method outperforms previous state-of-the-art methods significantly on the CQ by achieving an F1 of 44.2. It is important to note that our method is more effective when handling complicated KBs and questions. We compared our method with those of References [6–8], which have staged query graph generation methods that cannot handle the complex questions. Reference [29] focused on multihop relations; however, without the limitation of the method, the search space grows exponentially. Chen et al. (2019) [9] used a beam search to face the multihop questions, but it did not effectively handle the issue of constraints. We compared our method with that of Bhutani et al. (2019) [30], which constructs complex query patterns using a set of simple queries. We also compared our method with that of Ansari et al. (2019) [31], which generates query graphs token by token. The most important thing is to compare our method with Lan et al. (2020) [11], which allows longer relation paths to modify the graph generation method and uses the beam search to reduce the search space; however, in query graph generation, Lan et al. (2020) [11] is not optimal in the selection of the relations in each iteration because a longer relation path means more relation choices. Although effective for multihop questions, these methods sometimes ignore the constraints for the questions with constraints. Consequently, we apply GCN to effectively fuse the information on the dependency structure and to encode the dependency structure, which is helpful for relation selection. Additionally, the constraints in the question are easier to capture through an analysis of the dependency structure. Our method not only focuses on reducing the search space but also increases the relation accuracy selection in the query graph generation process, which affects the query graph ranking. Table 2 shows that our method not only works well on complex questions but also works well on the WQSP, which proves the robustness of our method.

Table 2. Results on different QA datasets.

Method	**Dataset**	
	WQSP (F1)	**CQ (F1)**
[8]	69.0	-
[6]	-	40.9
[7]	-	42.8
[29]	67.9	-
[9]	68.5	35.3
[30]	60.3	-
[31]	72.6	-
[11]	74.0	43.3
Our	74.8	44.2

4.3. Qualitative Analysis

The questions containing constraints are extracted from the CQ (approximately 25%) and WQSP (approximately 10%) test datasets to verify the effectiveness of our method for questions containing constraints. Table 3 shows the performance of the questions with constraints on the test dataset of CQ and WQSP. In Lan's [11] method, the relationship between words is not captured, and the constraints in some problems are omitted, leading to a lower accuracy on questions with constraints. Compared with Lan's [11] method, our method captures the relationship between words and has high sensitivity to the constraints in the question, so the accuracy of a question with constraints is higher. We also discuss the validity of the dependency structure of questions based on the GCN. By comparing the generated query graph, our method is proved to be effective.

Table 3. Performance of question with constraints on the test dataset of CQ and WQSP.

Method	CQ	WQSP
Lan et al. (2020) [11]	0.715	0.640
Our method	0.730	0.670

To summarize, our method not only affects the selection of relations in the graph generation process but also affects the ranking of the final query graph and even successfully captures some constraints that are difficult to capture. Therefore, our method is proven to be effective. Our method successfully affects the query graph generation process by convoluting the dependency structure of the question. In addition, the results show that our system performs stably and works well on not only multi-constraint questions but also on simple questions.

4.4. Error Analysis

We sampled 100 error cases randomly and obtained the following two types of errors. First, due to the query graph generation strategy, it is difficult to generate a query graph for some questions without predicate relations in the knowledge graph, which are approximately 63% of the questions. Second, the wrong query graph is generated due to the wrong entity or expression link, which is approximately 32% of the query graphs. For example, for the question "What guitar does Corey Taylor play?", there is no obvious constraint word in the question that leads to the wrong query graph.

5. Conclusions

In this paper, we proposed a graph convolution operation on a dependency structure of the question to obtain relation information between words and then integrated the relation information into the question vector to generate and rank the query graph. Our proposed methods have a dual objective of reducing the search space and improving the accuracy of relation selection during the query graph generation process. This, in turn, has a direct impact on the ranking of query graphs. Through experimentation, the results have demonstrates the effectiveness of our approach in addressing both complex questions and the WQSP dataset, thereby highlighting the robustness of our method. Notably, our method has shown a significant improvement over previous baseline methods.

Our methods also have its own weaknesses. One such weakness may be in the handling of certain types of questions or datasets that require specialized treatment or have unique characteristics. Additionally, there may be limitations in terms of scalability and efficiency when dealing with extremely large-scale datasets or in scenarios with real-time constraints. These weaknesses provide opportunities for future research and improvement. In future work, we plan to explore additional enhancements. One aspect we will focus on is pruning dependency structures to eliminate unnecessary information, which can help streamline the processing and improve efficiency. Furthermore, we aim to increase the accuracy of answer prediction, ensuring more precise and reliable responses. By continuously refining and expanding our approach, we anticipate further advancements in the field of question answering systems.

Author Contributions: Conceptualization, C.Z.; methodology, C.Z.; software, D.Z.; validation, D.Z., L.W. and C.Z.; formal analysis, N.M.; investigation, C.Z.; resources, C.Z.; writing—original draft preparation, C.Y., C.Z.; writing—review and editing, C.Z., B.W., F.X. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Social Science Foundation under Award 19BYY076; in part by the Key R & D project of Shandong Province 2019 JZZY010129; in part by the Shandong Natural Science Foundation under Award ZR2021MF064, Award ZR2021MF064, and Award ZR2021QG041; and in part by the Shandong Provincial Social Science Planning Project under Award 19BJCJ51, Award 18CXWJ01, and Award 18BJYJ04. This project is also supported by Major Science and Technology Demonstration Projects: Intelligent Perception Technology in

Complex Dynamic Scenes and IT Application Demonstration in Emergency Management and Social Governance, No. 2021SFGC0102).

Data Availability Statement: The data presented in this study are openly available in [6,8].

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Bordes, A.; Usunier, N.; Chopra, S.; Weston, J. Large-scale simple question answering with memory networks. *arXiv* **2015**, arXiv:1506.02075. [CrossRef].
2. Cai, Q.; Alexander, Y. Large-scale semantic parsing via schema matching and lexicon extension. In Proceedings of the Annual Meeting of the Association for Computational Linguistics, Sofia, Bulgaria, 4–9 August 2013; pp. 423–433.
3. Krishnamurthy, J.; Mitchel, T.M. Weakly supervised training of semantic parsers. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, Jeju Island, Republic of Korea, 12–14 July 2012; pp. 754–765.
4. Abujabal, A.; Yahya, M.; Riedewald, M.; Weikum, G. Automated template generation for question answering over knowledge graphs. In Proceedings of the 26th International Conference on World Wide Web, Perth, Australia, 3–7 April 2017; pp. 1191–1200. [CrossRef]
5. Hu, S.; Zou, L.; Yu, J.X.; Wang, H.; Zhao, D. Answering natural language questions by subgraph matching over knowledge graphs. *IEEE Trans. Knowl. Data Eng.* **2017**, *30*, 824–837. [CrossRef]
6. Bao, J.; Duan, N.; Yan, Z.; Zhou, M.; Zhao, T. Constraint-based question answering with knowledge graph. In Proceedings of the COLING, Osaka, Japan, 11–16 December 2016; pp. 2503–2514.
7. Luo, K.; Lin, F.; Luo, X.; Zhu, K.Q. Knowledge base question answering via encoding of complex query graphs. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 2185–2194. [CrossRef]
8. Yih, W.-T.; Chang, M.-W.; He, X.; Gao, J. Semantic Parsing via Staged Query Graph Generation: Question Answering with Knowledge Base. In Proceedings of the Annual Meeting of the Association for Computational Linguistics, Beijing, China, 26–31 July 2015.
9. Chen, Z.-Y.; Chang, C.-H.; Chen, Y.-P.; Nayak, J.; Ku, L.-W. UHop: An unrestricted-hop relation extraction framework for knowledge-based question answering. *arXiv* **2019**, arXiv:1904.01246. [CrossRef].
10. Lan, Y.; Wang, S.; Jiang, J. Multi-hop knowledge base question answering with an iterative sequence matching model. In Proceedings of the IEEE International Conference on Data Mining (ICDM), Beijing, China, 8–11 November 2019; pp. 359–368. [CrossRef]
11. Lan, Y.; Jiang, J. Query graph generation for answering multi-hop complex questions from knowledge bases. In Proceedings of the Annual Meeting of the Association for Computational Linguistics, Online, 5–10 July 2020. [CrossRef]
12. Miwa, M.; Bansal, M. End-to-End Relation Extraction using LSTMs on Sequences and Tree Structures. In Proceedings of the Annual Meeting of the Association for Computational Linguistics, Berlin, Germany, 7–12 August 2016. [CrossRef]
13. Xu, K.; Feng, Y.; Huang, S.; Zhao, D. Semantic Relation Classification via Convolutional Neural Networks with Simple Negative Sampling. *Comput. Sci.* **2015**, *71*, 941–949. [CrossRef]
14. Youcef, D.; Gautam, S.; Wei, L.J.C. Fast and accurate convolution neural network for detecting manufacturing data. *IEEE Trans. Ind. Inform.* **2020**, *17*, 2947–2955. [CrossRef]
15. Peng, H.; Chang, M.; Yih, W.T. Maximum margin reward networks for learning from explicit and implicit supervision. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, Copenhagen, Denmark, 7–11 September 2017; pp. 2368–2378. [CrossRef]
16. Sorokin, D.; Gurevych, I. Modeling semantics with gated graph neural networks for knowledge base question answering. In Proceedings of the 27th International Conference on Computational Linguistics, Association for Computational Linguistics, Santa Fe, NM, USA, 20–26 August 2018; pp. 3306–3317. [CrossRef]
17. Iyyer, M.; Yih, W.-T.; Chang, M.-W. Search-based neural structured learning for sequential question answering. In Proceedings of the Annual Meeting of the Association for Computational Linguistics, Vancouver, BC, Canada, 30 July–4 August 2017; pp. 1821–1831. [CrossRef]
18. Krishnamurthy, J.; Dasigi, P.; Gardner, M. Neural semantic parsing with type constraints for semi-structured tables. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, Copenhagen, Denmark, 7–11 September 2017; pp. 1516–1526. [CrossRef]
19. Moqurrab, S.A.; Ayub, U.; Anjum, A.; Asghar, S.; Srivastava, G. An accurate deep learning model for clinical entity recognition from clinical notes. *IEEE J. Biomed. Health Inform.* **2021**, *25*, 3804–3811. [CrossRef] [PubMed]
20. Wang, F.; Wu, W.; Li, Z.; Zhou, M. Named entity disambiguation for questions in community question answering. *Knowl.-Based Syst.* **2017**, *126*, 68–77. [CrossRef]
21. Bast, H.; Haussmann, E. More accurate question answering on freebase. In Proceedings of the 24th ACM International Conference on Information and Knowledge Management, Melbourne, Australia, 18–23 October 2015; pp. 1431–1440. [CrossRef]
22. Chakraborty, N.; Lukovnikov, D.; Maheshwari, G.; Trivedi, P.; Lehmann, J.; Fischer, A. Introduction to neural network based approaches for question answering over knowledge graphs. *arXiv* **2019**, arXiv:1907.09361. [CrossRef].

23. Chen, H.-C.; Chen, Z.-Y.; Huang, S.-Y.; Ku, L.-W.; Chiu, Y.-S.; Yang, W.-J. Relation extraction in knowledge base question answering: From general-domain to the catering industry. In Proceedings of the International Conference on HCI in Business, Government, and Organizations, Las Vegas, NV, USA, 15 July 2018; pp. 26–41. [CrossRef]
24. Yang, Z.; Garg, H.; Li, J.; Srivastava, G.; Cao, Z. Investigation of multiple heterogeneous relationships using a q-rung orthopair fuzzy multi-criteria decision algorithm. *Neural Comput. Appl.* **2021**, *33*, 10771–10786. [CrossRef]
25. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2019**, arXiv:1810.04805. [CrossRef].
26. Kipf, T.N.; Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv* **2016**, arXiv:1609.02907. [CrossRef].
27. Marcheggiani, D.; Ivan, T. Encoding Sentences with Graph Convolutional Networks for Semantic Role Labeling. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Copenhagen, Denmark, 7–11 September 2017; pp. 1506–1515. [CrossRef]
28. Das, R.; Dhuliawala, S.; Zaheer, M.; Vilnis, L.; Durugkar, I.; Krishnamurthy, A.; Smola, A.; McCallum, A. Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning. *arXiv* **2018**, arXiv:1711.05851.
29. Lan, Y.; Wang, S.; Jiang, J. Knowledge base question answering with topic units. In Proceedings of the International Joint Conference on Artificial Intelligence, Macao, China, 10–16 August 2019; pp. 5046–5052. [CrossRef]
30. Bhutani, N.; Suhara, Y.; Tan, W.-C.; Halevy, A.Y.; Jagadis, H.V. Open Information Extraction from Question-Answer Pairs. In Proceedings of the 17th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2019), Minneapolis, MN, USA, 3–5 June 2019; pp. 2294–2305.
31. Ahmed, G.A.; Saha, A.; Kumar, V.; Bhambhani, M.; Sankaranarayanan, K.; Chakrabarti, S. Neural Program Induction for KBQA Without Gold Programs or Query Annotations. In Proceedings of the International Joint Conference on Artificial Intelligence, Macao, China, 10–16 August 2019; pp. 4890–4896. [CrossRef]