Enhancing Open-Domain Table Question Answering via Syntax- and Structure-aware Dense Retrieval

Nengzheng Jin[1], Dongfang Li[1], Junying Chen[1], Joanna Siebert[1], Qingcai Chen[1,2*]
[1]Harbin Institute of Technology (Shenzhen), Shenzhen, China
[2]Peng Cheng Laboratory
{nengzhengjin,crazyofapple,junying.chen.cs,joannasiebert}@gmail.com
qingcai.chen@hit.edu.cn

Abstract

Open-domain table question answering aims to provide answers to a question by retrieving and extracting information from a large collection of tables. Existing studies of open-domain table QA either directly adopt text retrieval methods or consider the table structure only in the encoding layer for table retrieval, which may cause syntactical and structural information loss during table scoring. To address this issue, we propose a syntax- and structure-aware retrieval method for the open-domain table QA task. It provides syntactical representations for the question and uses the structural header and value representations for the tables to avoid the loss of fine-grained syntactical and structural information. Then, a syntactical-to-structural aggregator is used to obtain the matching score between the question and a candidate table by mimicking the human retrieval process. Experimental results show that our method achieves the state-of-the-art on the NQ-tables dataset and overwhelms strong baselines on a newly curated open-domain Text-to-SQL dataset[1].

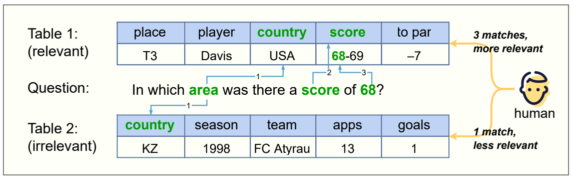

Figure 1: Illustration of fine-grained semantic matching when a human retrieves relevant tables for a question. If more fine-grained matches are found in a table, one will treat the table as more relevant.

1 Introduction

Open-domain table QA uncovers the necessity of table retrieval for practical applications. It slightly differs from most table QA tasks (e.g., table semantic parsing (Yin and Neubig, 2018; Wang et al., 2020) and end-to-end table QA (Müller et al., 2019; Eisenschlos et al., 2021)), which typically assume that the relevant tables of a question are provided at test time. This assumption can hardly hold when the user is asking questions through some open-domain natural language interface or querying large databases.

Hence, some recent studies (Herzig et al., 2021; Chen et al., 2022; Zhong et al., 2022) have started to explore open-domain table QA. Generally, these methods are based on a two-stage framework, namely the retriever-reader (parser) framework. In such frameworks, the retriever is utilized to obtain relevant tables of the given question and the reader (parser) provides answers to the question directly or parses the question into an SQL query.

Since retrieval is the first key factor for open-domain table QA, recent works have investigated different approaches for table retrieval, which could be broadly classified into two categories. One is directly adopting text retrieval methods for table retrieval. Such methods typically linearize the tables into text and apply sparse text retrieval (e.g., BM25) (Li et al., 2021) or dense passage retrieval with a Bi-encoder model (Oguz et al., 2022; Chen et al., 2021; Huang et al., 2022). Another direction is to follow the framework of text retrieval models while also considering the unique characteristics of tables. For instance, DTR (Herzig et al., 2021) follows the Bi-encoder framework but incorporates row/column features into the table encoder, aiming to specify the cell location and enhance table understanding. UTP (Chen et al., 2023) introduces pre-training and cross-model contrastive regularization for better tabular understanding.

Although some of these methods (Herzig et al., 2021; Chen et al., 2023) have modeled the characteristics of tables, they still have two shortages. First, the learned tabular semantics may be compromised when all token embeddings are combined

[*]Corresponding author.
[1]The data and processing code are publicly available at https://github.com/nzjin/ODTQA.

Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (Volume 2: Short Papers), pages 157–165
November 1–4, 2023.

into a single table representation in the Bi-encoder retrieval framework (Gillick et al., 2018). Second, they neglect that table retrieval is a fine-grained semantic matching problem. As illustrated in Figure 1, humans would consciously match each meaningful phrase in the question to the table columns and rank the candidate tables by the matching degree.

Motivated by this, we propose a syntax- and structural-aware dense retrieval method to mimic this fine-grained matching process. We first apply syntax analysis to extract all possible meaningful phrases. Then, a corresponding syntactical representation is generated for each meaningful phrase based on the phrase token embeddings. To obtain fine-grained structural representations, we provide one representation for each table header. Further, we observe that the semantics of a table header and its value may be different; for example, a header may be "age" but the corresponding values are numbers. Therefore, we also provide one representation for the values of a column to retain the structural semantics better. Finally, the matching score between the question and a candidate table is obtained by performing syntactical-to-structural aggregation over the fine-grained representations, wherein the aggregation is analogous to the human behavior of counting matches.

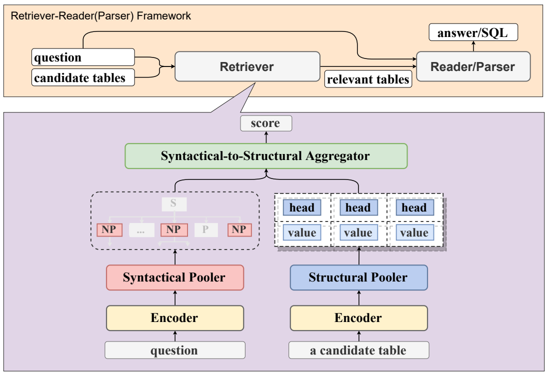

Figure 2: Illustration of the retriever-reader (parser) framework and our proposed retriever. Here, *head* and *value* denote the structural table representations. *NP* (noun phrase) denotes a syntactical question representation.

2 Methodology

2.1 Overview

As shown in Figure 2, our proposed retrieval model comprises of 1) *Syntactical representation module* that generates fine-grained syntactical representations for the question; 2) *Structural representation module* that generates a limited number of structural representations for a table; 3) *Syntactical-to-structural aggregator* that produces the matching score between the question and a candidate table.

2.2 Syntactical Question Representation

Explicit syntactical representations. We utilize an explicit syntax parser from the Natural Language Toolkit (NLTK) (Bird et al., 2009) to extract the noun phrases in the question. To generate the question representations, we first feed the question tokens of length L into the encoder and treat the hidden states of the last layer as question token embeddings $\{\mathbf{h}_l\}_{l=1}^{L}$. Then, we apply a mean pooling function over the token embeddings that belong to a phrase, thus obtaining one representation $\mathbf{q}_i$ for the ith phrase in the question:

$$\mathbf{q}_i = \mathrm{Pooling}(\mathbf{h}_{start_i}, ..., \mathbf{h}_{end_i}), \tag{1}$$

where $start_i$ and end_i denote the token span of the ith phrase. Applying the same operation for each phrase, we obtain a small group of fine-grained syntactical representations $\{\mathbf{q}_i\}_{i=1}^{n}$, where n is the number of the syntactical representations.

Implicit syntactical representations. Since explicit syntax parsing is associated with additional preprocessing time, we provide an implicit syntactical representation approach that does not require syntax parsing. It attempts to learn a static number of syntactical representations through an attentive learning mechanism based on the token embeddings $\{\mathbf{h}_l\}_{l=1}^{L}$. Specifically, we first define a set of randomly initialized learning embeddings $\{\mathbf{a}_i\}_{i=1}^{n}$. Then, we feed $\{\mathbf{a}_i\}_{i=1}^{n}$ and $\{\mathbf{h}_l\}_{l=1}^{L}$ into an attention layer to generate the final syntactical representations $\{\mathbf{q}_i\}_{i=1}^{n}$, where $\mathbf{q}_i$ is learned as in Eqn. 2 and Eqn. 3.

$$\mathbf{q}_i = \sum_l \mathbf{w}_l^i \mathbf{h}_l \tag{2}$$

$$(\mathbf{w}_1^i, .., \mathbf{w}_L^i) = \mathrm{softmax}(\mathbf{a}_i \cdot \mathbf{h}_1, .., \mathbf{a}_i \cdot \mathbf{h}_L). \tag{3}$$

Here, $\mathbf{a}_i$ is used as a seed to learn an implicit syntactical representation $\mathbf{q}_i$ during training.

2.3 Structural Table Representation

For each candidate table, we linearize the table into a sequence by concatenating the columns, wherein a column sequence consists of the column name (also referred to as the header) and column values.

Then, we use the same encoder to encode the table sequence of length T. After obtaining table token embeddings $\{\mathbf{s}_l\}_{l=1}^{T}$, we apply a mean pooling function over the embeddings of each column name to generate a header representation $\mathbf{c}_j^{head}$. Additionally, we perform the same pooling function over the column value to generate a value representation $\mathbf{c}_j^{val}$. The pooling is only applied to the first column value rather than all values as it proved to be more effective and efficient in our preliminary experiments. After that, we obtain two representations for the jth column as follows:

$$\mathbf{c}_j^{head} = \text{Pooling}(\mathbf{s}_{start_j^{head}}, .., \mathbf{s}_{end_j^{head}}), \quad (4)$$

$$\mathbf{c}_j^{val} = \text{Pooling}(\mathbf{s}_{start_j^{val}}, .., \mathbf{s}_{end_j^{val}}). \quad (5)$$

Here, $start_j^{head/val}$ and $end_j^{head/val}$ represent the token span corresponding to the header (value) of the jth column.

2.4 Syntactical-to-Structural Aggregator

We obtain syntactical question representations $\{\mathbf{q}_i\}_{i=1}^{n}$ and structural table representations $\{\mathbf{c}_j\}_{j=1}^{2m}$ after representation generation, where $\{\mathbf{c}_j\}_{j=1}^{2m}$ includes $\{\mathbf{c}_j^{head}\}_{j=1}^{m}$ and $\{\mathbf{c}_j^{val}\}_{j=1}^{m}$, and m is the number of columns. Then, we perform a *maxsim* operation (Khattab and Zaharia, 2020; Luan et al., 2021a) over the syntactical and structural representations to retrieve tables. Specifically, we calculate the dot product similarity between each syntactical representation $\mathbf{q}_i$ and each structural representation $\mathbf{c}_j$ to obtain a fine-grained matching score w_{ij} as follows:

$$w_{ij} = \mathbf{q}_i \cdot \mathbf{c}_j \quad (6)$$

$$Score = \sum_{i}^{n} \max_{j \in [1,2m]} w_{ij}. \quad (7)$$

Subsequently, we select the fine-grained score of the most matched column for each syntactical representation and sum up all the greatest fine-grained scores as the final matching score between the question and a candidate table. This is analogous to the process of a human finding a match for each phrase and counting the number of matches when retrieving tables.

3 Experiments

3.1 Experimental Data and Settings

Datasets. We conducted experiments on two datasets: **NQ-TABLES** (Herzig et al., 2021), **WikiSQL** (Zhong et al., 2017). NQ-TABLES is an open-domain table QA database constructed from the Natural Question dataset (Kwiatkowski et al., 2019). In the original WikiSQL dataset, the relevant tables are given by humans. To simulate a realistic situation, we remove the table labeling of the questions and introduce table retrieval. Further details of the dataset modification can be seen in Appendix A. Concurrently with this work, (Kweon et al., 2023) also introduce an open-domain setting to the WikiSQL dataset but does not process those same-header database tables.

Model	R@1	R@10	R@50	EM
BM25	16.77	40.06	58.39	21.46
DTR-Schema	34.36	74.24	88.37	32.75
DTR	36.24	76.02	90.25	35.50
UTP	38.45	79.03	92.21	-
Ours(ex)	45.15	83.73	93.12	37.73
Ours(im)	**47.03**	**84.76**	**94.89**	**37.98**
With hard negatives training				
DTR	42.42	81.13	92.56	37.69
UTP	50.39	85.40	94.31	-
Tri-encoder	-	86.4	-	-
Ours(ex)	53.39	88.11	95.09	39.47
Ours(im)	**54.12**	**90.41**	**97.18**	**39.72**

Table 1: Experimental results on the NQ-TABLES dataset. R is short for recall. Here, ex/im denotes using explicit or implicit syntactical representations.

Settings. The uncased BERT-base (Devlin et al., 2019) is employed as the encoder. The number of implicit syntactical representations is set to 3 and Adam is utilized (Kingma and Ba, 2015) as the optimizer. To evaluate the performance of the retrievers, the recall@K and exact match accuracy (EM) of the final answers are utilized as the metrics. Since the focus of this work is to enhance open-domain table QA via improved retrievers, TAPAS (Herzig et al., 2020) and HydraNet (Lyu et al., 2020) are directly used as the reader (parser) on the NQ-TABLES and WikiSQL datasets, respectively.

3.2 Comparison Models

We compared the proposed retriever with the following baselines. 1) Sparse retrievers: **TF-IDF** and **BM25** (Robertson and Walker, 1994). 2) Dense retrievers using one representation, such as **Bi-encoder** (Gillick et al., 2018) with representation vectors from word2vec (w2v) and BERT, as well as **DTR** (Herzig et al., 2021) and **UTP** (Chen et al., 2023), which use a table-oriented pre-trained model as encoder. 3) Dense retrievers using multiple representations: **Tri-encoder** (Kostic et al.,

Model	R@1	R@5	R@20	EM
TF-IDF	8.93	22.86	44.05	7.24
BM25	28.19	44.71	61.22	26.18
Bi-encoder(w2v)	10.01	20.14	30.13	7.92
Bi-encoder(BERT)	48.89	73.15	86.64	41.26
MEBERT	49.05	73.30	87.34	41.32
PolyEncoder	49.33	73.61	87.78	41.63
Ours(ex)	**54.16**	**77.63**	89.81	**45.42**
Ours(im)	53.19	77.57	**90.02**	44.71

Table 2: Experimental results on the WikiSQL dataset.

Model	R@1	R@5	R@20	EM
Ours(ex)	54.16	77.63	89.81	45.42
w/o $\mathcal{S}^1$	50.08	74.36	88.19	42.28
w/o $\mathcal{S}^2$ $(head)$	52.93	77.08	89.91	44.51
w/o $\mathcal{S}^2$ $(value)$	53.18	77.16	89.50	44.75
w/o $\mathcal{S}^2$	48.31	72.45	86.75	40.71
w/o $\mathcal{S}^1 + \mathcal{S}^2$	48.87	72.57	86.79	41.17

Table 3: Ablation sturdy on the WikiSQL dataset. $\mathcal{S}^1$ and $\mathcal{S}^2$ denote syntactical and structural representations.

Model	LAT.	Model	LAT.
TF-IDF	2.74	MEBERT	0.40
BM25	6.95	PolyEncoder	0.41
Bi-encoder(w2v)	0.32	Ours(ex)	$0.42 + 0.6^*$
Bi-encoder(BERT)	0.36	Ours(im)	0.47

Table 4: The retrieval latency (LAT.) per question (in milliseconds) on the WikiSQL dataset. $*$ denotes the latency of syntax parsing.

2021), **PolyEncoder** (Humeau et al., 2020) and **MEBERT** (Luan et al., 2021a). We set the representation number of PolyEncoder and MEBERT to the average representation number used in our model.

3.3 Experimental Results and Analysis

Table 1 demonstrates the experimental results on the NQ-TABLES dataset. It shows that our proposed model outperforms the previous state-of-the-art model by a considerable margin with or without hard negative training (Gillick et al., 2019). Specifically, our model surpasses previous models by approximately 8 and 2 points in terms of recall@1 and EM accuracy. Furthermore, as Table 2 illustrates, our method consistently outperforms strong text retrievers on the WikiSQL dataset. This verifies the effectiveness of the proposed syntax- and structure-aware dense retrieval method.

The experimental results in Table 1 and 2 also indicate that the implicit syntactical question representations yield better performance compared to the explicit ones in most cases. The underlying reason may be that the attention layer in the implicit syntactical pooler can learn more delicate semantic information compared to simple pooling.

Ablation Study. To analyze the impact of different components of our proposed method, we conducted an ablation study in Table 3. 1) w/o $\mathcal{S}^1$ variant eliminates syntactical question representations and takes the embedding of the $[CLS]$ token in the question as the representation. The result of w/o $\mathcal{S}^1$ shows the importance of syntactical representations. Moreover, our method still outperforms text retrieval baselines in this case, which also verifies the effectiveness of structural table representations compared with multiple contextualized representations (Humeau et al., 2020; Luan et al., 2021b). 2) w/o $\mathcal{S}^2$ variant uses the embedding of the $[CLS]$ token in the table sequence as the table representation. The results of w/o $\mathcal{S}^2$, w/o $\mathcal{S}^2(head)$, and w/o $\mathcal{S}^2(value)$ indicate that structural header and value representations both contribute to better representations of a table. 3) w/o $\mathcal{S}^1 + \mathcal{S}^2$ variant leads to inferior performance but is slightly better than the w/o $\mathcal{S}^2$ variant, this suggests that the use of syntactical representations alone does not improve performance.

Latency Analysis. We compared the retrieval latency of the retrievers on the WikiSQL dataset. As Table 4 illustrates, our model has an acceptable latency increase compared to the baselines but makes considerable progress in retrieval performance.

4 Related Works

Open-domain table QA (ODTQA) is an extension of the closed-domain table QA task (Pasupat and Liang, 2015; Zhong et al., 2017; Yin et al., 2020; Chen et al., 2021). Traditional closed-domain table QA does not require table retrieval and can be addressed through two primary methodologies: employing end-to-end models or utilizing semantic parsing approaches. End-to-end models take both the questions and relevant tables as input, and then directly generate answers (Müller et al., 2019; Zhu et al., 2021; Nararatwong et al., 2022; Nan et al., 2022). Differently, semantic parsing approaches transform the question into a logical form (e.g., an SQL query), and then retrieve the answers by executing the logical form (Zhong et al., 2017; Yu et al., 2018; Wang et al., 2020; Zhu et al., 2021).

ODTQA is also closely related to open-domain text-based QA (Kwiatkowski et al., 2019; Khashabi et al., 2021) and information retrieval (Luan et al., 2021a; Humeau et al., 2020; Tang et al., 2021). Compared to the text retrieval, the tabular characteristics need to be considered in ODTQA (Herzig et al., 2021; Chen et al., 2023; Kostic et al., 2021). Another task that ODTQA shares some degree of similarity with is keyword-based web table search (Zhang and Balog, 2018; Sun et al., 2019; Chen et al., 2020; Wang et al., 2021; Trabelsi et al., 2022). Compared with keyword-based web table search, table retrieval in open-domain table QA needs to process complex questions that may contain superfluous information rather than to process informative keyword queries. Hence, our work incorporates syntax analysis to extract useful syntactical representation, as well as uses simple yet effective structural representation and scoring mechanism for retrieval efficiency.

5 Conclusion

In this paper, we present a syntax- and structure-aware dense retrieval method for open-domain table QA. Specifically, our method mimics the human retrieval process by utilizing fine-grained syntactical and structural representations, as well as a syntactical-to-structural aggregator. Experimental results on two datasets demonstrate that our model surpasses the strong baselines while preserving a reasonable computation overhead.

References

Bird, Steven, Edward Loper, and Ewan Klein. 2009. *Natural Language Processing with Python*. O'Reilly Media Inc.

Nuo Chen, Linjun Shou, Ming Gong, Jian Pei, Chenyu You, Jianhui Chang, Daxin Jiang, and Jia Li. 2023. Bridge the gap between language models and tabular understanding. *CoRR*, abs/2302.09302.

Siqin Chen, Yubo Liu, Jiehui Wu, and Mengshu Hou. 2022. Retrieval augmented via execution guidance in open-domain table qa. *Proceedings of the 2022 5th International Conference on Algorithms, Computing and Artificial Intelligence*.

Wenhu Chen, Ming-Wei Chang, Eva Schlinger, William Yang Wang, and William W. Cohen. 2021. Open question answering over tables and text. In *9th International Conference on Learning Representations*, Virtual Event, Austria. OpenReview.net.

Zhiyu Chen, Mohamed Trabelsi, Jeff Heflin, Yinan Xu, and Brian D. Davison. 2020. Table search using a deep contextualized language model. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 589–598. ACM.

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 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT*, pages 4171–4186, Minneapolis, MN, USA. Association for Computational Linguistics.

Julian Eisenschlos, Maharshi Gor, Thomas Müller, and William W. Cohen. 2021. MATE: multi-view attention for table transformer efficiency. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 7606–7619. Association for Computational Linguistics.

Daniel Gillick, Sayali Kulkarni, Larry Lansing, Alessandro Presta, Jason Baldridge, Eugene Ie, and Diego García-Olano. 2019. Learning dense representations for entity retrieval. In *Proceedings of the 23rd Conference on Computational Natural Language Learning*, pages 528–537, Hong Kong, China. Association for Computational Linguistics.

Daniel Gillick, Alessandro Presta, and Gaurav Singh Tomar. 2018. End-to-end retrieval in continuous space. *CoRR*, abs/1811.08008.

Jonathan Herzig, Thomas Müller, Syrine Krichene, and Julian Eisenschlos. 2021. Open domain question answering over tables via dense retrieval. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 512–519, Online. Association for Computational Linguistics.

Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Martin Eisenschlos. 2020. Tapas: Weakly supervised table parsing via pre-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4320–4333, Online. Association for Computational Linguistics.

Junjie Huang, Wanjun Zhong, Qian Liu, Ming Gong, Daxin Jiang, and Nan Duan. 2022. Mixed-modality representation learning and pre-training for joint table-and-text retrieval in openqa. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 4117–4129, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Samuel Humeau, Kurt Shuster, Marie-Anne Lachaux, and Jason Weston. 2020. Poly-encoders: Architectures and pre-training strategies for fast and accurate multi-sentence scoring. In *8th International Conference on Learning Representations (ICLR)*, Addis Ababa, Ethiopia. OpenReview.net.

Daniel Khashabi, Amos Ng, Tushar Khot, Ashish Sabharwal, Hannaneh Hajishirzi, and Chris Callison-Burch. 2021. Gooaq: Open question answering with diverse answer types. In *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 421–433. Association for Computational Linguistics.

Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over BERT. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 39–48, Virtual Event, China. ACM.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations (ICLR)*, San Diego, CA, USA.

Bogdan Kostic, Julian Risch, and Timo Möller. 2021. Multi-modal retrieval of tables and texts using tri-encoder models. *CoRR*, abs/2108.04049.

Sunjun Kweon, Yeonsu Kwon, Seonhee Cho, Yohan Jo, and Edward Choi. 2023. Open-wikitable: Dataset for open domain question answering with complex reasoning over table. *CoRR*, abs/2305.07288.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur P. Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: a benchmark for question answering research. *Trans. Assoc. Comput. Linguistics*, 7:452–466.

Alexander Hanbo Li, Patrick Ng, Peng Xu, Henghui Zhu, Zhiguo Wang, and Bing Xiang. 2021. Dual reader-parser on hybrid textual and tabular evidence for open domain question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL/IJCNLP) (Volume 1: Long Papers)*, pages 4078–4088, Virtual Event. Association for Computational Linguistics.

Yi Luan, Jacob Eisenstein, Kristina Toutanova, and Michael Collins. 2021a. Sparse, dense, and attentional representations for text retrieval. *Trans. Assoc. Comput. Linguistics*, 9:329–345.

Yi Luan, Jacob Eisenstein, Kristina Toutanova, and Michael Collins. 2021b. Sparse, dense, and attentional representations for text retrieval. *Transactions of the Association for Computational Linguistics*, 9:329–345.

Qin Lyu, Kaushik Chakrabarti, Shobhit Hathi, Souvik Kundu, Jianwen Zhang, and Zheng Chen. 2020. Hybrid ranking network for text-to-sql. *CoRR*, abs/2008.04759.

Thomas Müller, Francesco Piccinno, Peter Shaw, Massimo Nicosia, and Yasemin Altun. 2019. Answering conversational questions on structured data without logical forms. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 5901–5909. Association for Computational Linguistics.

Linyong Nan, Chiachun Hsieh, Ziming Mao, Xi Victoria Lin, Neha Verma, Rui Zhang, Wojciech Kryściński, Hailey Schoelkopf, Riley Kong, Xiangru Tang, Mutethia Mutuma, Ben Rosand, Isabel Trindade, Renusree Bandaru, Jacob Cunningham, Caiming Xiong, Dragomir Radev, and Dragomir Radev. 2022. FeTaQA: Free-form table question answering. *Transactions of the Association for Computational Linguistics*, 10:35–49.

Rungsiman Nararatwong, Natthawut Kertkeidkachorn, and Ryutaro Ichise. 2022. Enhancing financial table and text question answering with tabular graph and numerical reasoning. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing, AACL/IJCNLP 2022 - Volume 1: Long Papers, Online Only, November 20-23, 2022*, pages 991–1000. Association for Computational Linguistics.

Barlas Oguz, Xilun Chen, Vladimir Karpukhin, Stan Peshterliev, Dmytro Okhonko, Michael Sejr Schlichtkrull, Sonal Gupta, Yashar Mehdad, and Scott Yih. 2022. Unik-qa: Unified representations of structured and unstructured knowledge for open-domain question answering. In *Findings of the Association for Computational Linguistics (NAACL)*, pages 1535–1546, Seattle, WA, United States. Association for Computational Linguistics.

Panupong Pasupat and Percy Liang. 2015. Compositional semantic parsing on semi-structured tables. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing (Volume 1: Long Papers)*, pages 1470–1480, Beijing, China. The Association for Computer Linguistics.

Stephen E. Robertson and Steve Walker. 1994. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *Proceedings of the 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval.*, pages 232–241, Dublin, Ireland. ACM/Springer.

Yibo Sun, Zhao Yan, Duyu Tang, Nan Duan, and Bing Qin. 2019. Content-based table retrieval for web queries. *Neurocomputing*, 349:183–189.

Hongyin Tang, Xingwu Sun, Beihong Jin, Jingang Wang, Fuzheng Zhang, and Wei Wu. 2021. Improving document representations by generating pseudo query embeddings for dense retrieval. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL/IJCNLP) (Volume 1: Long Papers)*, pages 5054–5064, Virtual Event. Association for Computational Linguistics.

Mohamed Trabelsi, Zhiyu Chen, Shuo Zhang, Brian D. Davison, and Jeff Heflin. 2022. Strubert: Structure-aware BERT for table search and matching. In *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*, pages 442–451. ACM.

Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. 2020. RAT-SQL: relation-aware schema encoding and linking for text-to-sql parsers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7567–7578. Association for Computational Linguistics.

Fei Wang, Kexuan Sun, Muhao Chen, Jay Pujara, and Pedro A. Szekely. 2021. Retrieving complex tables with multi-granular graph representation learning. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, pages 1472–1482. ACM.

Pengcheng Yin and Graham Neubig. 2018. TRANX: A transition-based neural abstract syntax parser for semantic parsing and code generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018: System Demonstrations, Brussels, Belgium, October 31 - November 4, 2018*, pages 7–12. Association for Computational Linguistics.

Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. 2020. Tabert: Pretraining for joint understanding of textual and tabular data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 8413–8426. Association for Computational Linguistics.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir R. Radev. 2018. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3911–3921, Brussels, Belgium. Association for Computational Linguistics.

Shuo Zhang and Krisztian Balog. 2018. Ad hoc table retrieval using semantic similarity. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, pages 1553–1562. ACM.

Victor Zhong, Caiming Xiong, and Richard Socher. 2017. Seq2sql: Generating structured queries from natural language using reinforcement learning. *CoRR*, abs/1709.00103.

Wanjun Zhong, Junjie Huang, Qian Liu, Ming Zhou, Jiahai Wang, Jian Yin, and Nan Duan. 2022. Reasoning over hybrid chain for table-and-text open domain QA. *CoRR*, abs/2201.05880.

Fengbin Zhu, Wenqiang Lei, Youcheng Huang, Chao Wang, Shuo Zhang, Jiancheng Lv, Fuli Feng, and Tat-Seng Chua. 2021. TAT-QA: A question answering benchmark on a hybrid of tabular and textual content in finance. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 3277–3287. Association for Computational Linguistics.

A Dataset Modification

In the original WikiSQL dataset, tables relevant to each question are manually provided. To simulate a realistic situation, we removed the table labeling of the questions, introducing the task of table retrieval. Moreover, we merged the tables with the same headers into one distinct table, as it is implausible for an application database to accommodate two tables with identical headers. Furthermore, we created a mapping between the original tables and the distinct tables to enable using all samples for table retrieval and downstream tasks.

In consideration of the limited input length of a BERT-based encoder, we randomly sampled 5 rows of values for over-size tables, thereby obtaining smaller candidate tables during retrieval. If the resulting table sequence still exceeded the prescribed length limitation, we dynamically trimmed an equal number of values from each column until the specified limitation was met. The statistics of the processed WikiSQL dataset and the original NQ-TABLES dataset are reported in Table 5.

Dataset	Train	Dev	Test	# Tables
NQ-TABLES	9594	1068	966	169898
WikiSQL	56355	8421	15878	9898

Table 5: Statistics of NQ-TABLES and WikiSQL datasets.

B Experimental Setup

Our proposed retrieval model uses a training batch size of 144 and 128 on the NQ-TABLES dataset and WikiSQL dataset, respectively. The uncased BERT-base model is used as the encoder on two datasets. Following Tri-encoder (Kostic et al., 2021) and UTP (Chen et al., 2023), no down-projection is used for the final representations on the NQ-TABLES dataset. However, for a comprehensive evaluation, we also reported the performance of the proposed model with projection in Table 7. We trained the model for a maximum of 150 epochs on two datasets and adopted early stopping according to the recall numbers on the dev set. For efficiency, we only used the tables that appear in the dev set as the candidate pool for the early stopping, aligning with the method outlined in (Herzig et al., 2021).

We trained all models on 4 Nvidia Tesla A100 80GB GPUs. We tested each model on a single GPU. Detailed hyper-parameters for training the proposed retrieval model can be found in Table 6.

parameter	value
learning rate	2e-5
weight decay	0.01
warmup up ratio	0.05

Table 6: Hyper-parameters for training the proposed retrieval model.

Model	Size	Dim.	Bs.	R@1	R@10	R@50
DTR	large	256	256	36.24	76.02	90.25
Ours(ex)	base	256	144	41.29	82.06	91.86
Ours(im)	base	256	144	41.92	83.42	93.84

Table 7: Experimental results of the proposed method with down-projection on the NQ-TABLES dataset. Here, Size denotes the model size, Dim. denotes the dimensionality of each representation after down-projection, Bs. denotes the training batch size.

C Further Discussion

C.1 Impact of Different Pooling Functions on Performance

In the main experiments, we used the **mean pooling function** to retrieve explicit syntactical and structural representations. This function outputs the average of all input embeddings. Here, we explored the effect of two other pooling functions, namely the **max pooling function**, which selects the largest of the input embeddings as the output, and the **attentive pooling function**, which generates the output as a weighted sum of the input embeddings, with weights determined by a linear layer. As shown in Figure 8, the attentive pooling function achieves the best performance in most cases. This is likely due to the fact that the attentive pooling function is equipped with extra parameters to learn for pooling.

Model	R@1	R@5	R@20
Ours(mean pooling)	54.72	78.15	90.21
Ours(max pooling)	54.22	77.23	89.66
Ours(attentive pooling)	54.79	78.14	90.25

Table 8: Experimental results of the proposed method with different pooling functions on the WikiSQL dataset.

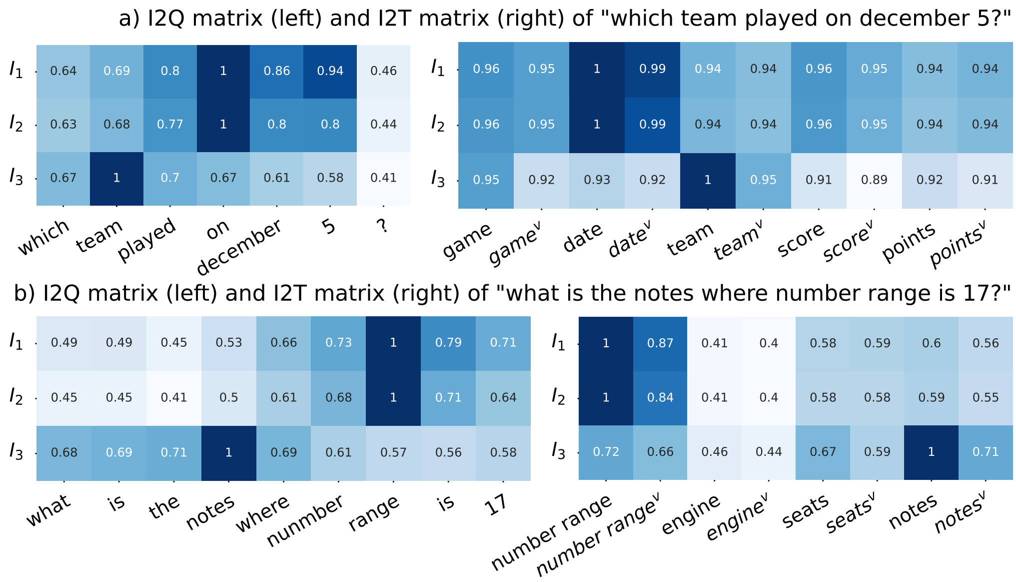

Figure 3: The coherence matrices of **I**mplicit syntactical representations ($\{I_j\}_{j=1}^{3}$) to **Q**uestion token embeddings (**I2Q**) and **T**able header/value representations (**I2T**). The matrices depict whether I_j learns syntactical information and matches the correct column. As shown in a), I_1 and I_2 learn information of "on December 5" and match the column "date" with the highest score. Here, $\mathbf{header^v}$ denotes value representations.

C.2 Coherence Matrices of Implicit Syntactical Representations

To investigate whether implicit syntactical representations effectively support fine-grained semantic matching akin to explicit representations, we visualized the normalized similarity matrix between implicit syntactical representations and question token embeddings (I2Q), as well as structural table representations (I2T) in Figure 3. The I2Q matrix indicates which tokens an implicit representation focuses on. Inspecting the I2Q matrix in Figure 3(a), it is clear that I_3 has the closest relationship with the phrase "*team*", while I_1 and I_2 both focus on the phrase "*on December 5*", as there are only two phrases in the question. We observe that different implicit representations usually focus on different syntactical phrases, which is similar to explicit representations. Then, carrying different syntactical information, I_j can perform neural fine-grained matching with structural header/value representations. As the I2T matrix in Figure 3(a) shows, I_3 matches the header "*team*" with the highest score, whereas I_1 and I_2 match the header and value of "*date*". Hence, the behavior of implicit representations is consistent with our design ideas.