Enhancing SQL Table Acquisition with Reverse Engineering for Text-to-SQL

Shixin Liu, Haoyu Xu, Yu Hong*
School of Computer Science and Technology, Soochow University, Suzhou, China
{sido.meet, xu.order.e, tianxianer}@gmail.com

Abstract

Text-to-SQL oriented table acquisition suffers from heterogeneous semantic gap. To address the issue, we propose a Reverse Engineering (RE) based optimization approach. Instead of forward table search using questions as queries, RE reversely generates potentially-matched question conditioned on table schemas, and promotes semantic consistency verification between homogeneous questions. We experiment on two benchmarks, including SpiderUnion and BirdUnion. The test results show that our approach yields substantial improvements compared to the Retrieval-Reranker (2R) baseline, and achieves competitive performance in both table acquisition and Text-to-SQL tasks.[1]

1 Introduction

We study SQL table acquisition for Text-to-SQL. Text-to-SQL is a task of generating the executable SQL query given a certain question (Yu et al., 2018; Kim et al., 2020; Li et al., 2023b; Katsogiannis-Meimarakis et al., 2023; Liu et al., 2025). Table acquisition is a crucial subtask of Text-to-SQL, which acquires the question-related SQL tables to support source-aware reasoning of SQL queries (Kothyari et al., 2023; Zhang et al., 2025). Figure 1 shows an example, where "✓" denotes the relevant table needs to be acquired from the SQL database.

The core of SQL table acquisition is relevance-centered text matching at the semantic level. Current studies struggle with the bottleneck of heterogeneous semantic gap. Specifically, a SQL table appears as the semi-structured data, whose schema is constituted with relation-unaware fragmented names (e.g., field names). By contrast, the question is a free text. Therefore, for a pair of mutually-aligned table and question, it is difficult to facilitate their semantics to exactly coincide with each other.

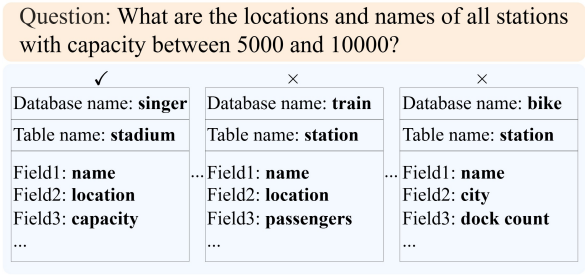

Figure 1: An example of SQL table acquisition.

This limits the implementation of anti-distraction relevance measurement.

To address the issue, we propose a Reverse Engineering (RE) based optimization approach. It leverages Large Language Model (LLM) to reversely generate potentially-matched question for every SQL table. On this basis, we conduct text matching between homogeneous data, i.e., the input question and the reversely-generated question. In our experiments, we use RE to strengthen Retrieval-Reranker (2R) based table acquisition model. Experiments on SpiderUnion (Kothyari et al., 2023) and BirdUnion (Kothyari et al., 2023) demonstrate that RE enables both the BGE-M3 (0.5B) based table acquisition and Text-to-SQL models to perform better. In addition, it achieves competitive performance compared to the strong models CRUSH (Kothyari et al., 2023) and MURRE (Zhang et al., 2025) that use a larger baseline SGPT (5.8B).

2 Approach

In general, SQL-table acquisition is performed in a Retriever-Reranker (2R) framework (Chen et al., 2024b; Kothyari et al., 2023), i.e., retrieving candidate SQL tables and re-ranking them according to question-oriented relevance. Top-k candidates will be adopted as the results. The 2R framework is formalized as follows.

$$T_r = f(q, T_o) \tag{2.1}$$

*Corresponding author

[1]Our code and data are available at https://github.com/sido-meet/Reverse-Engineering.

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 8034–8041
November 4-9, 2025

where, q is a question, T_o denotes all tables in the SQL database, T_r is the list of ranked SQL tables, while f serves as an 2R model.

We propose a reverse engineering approach to strengthen the 2R framework. It uses a LLM to reversely generate the **potentially-matched question** q_o for each SQL table. We specify q_o as a reference object, and integrate it into the 2R model to optimize relevance estimation. On this basis, we form a conditional 2R framework, which is formalized as follows.

$$T_r = f(q, T_o|Q_o) \tag{2.2}$$

where, Q_o ($q_o \in Q_o$) refers to the set of potentially-matched questions generated by LLM for all SQL tables, which serves as the crucial condition for optimizing relevance estimation.

In the subsections, we first present the 2R-based baseline, and further detail our reverse engineering based approach.

2.1 "2R" Baseline

Retriever— The retriever in the 2R baseline is developed with BGE-M3 (Chen et al., 2024a), which is pretrained on 1.2 billions of multilingual data.

In BGE-M3, two BERTs (Devlin et al., 2019) are coupled with pooling layers, which form two separate encoding channel. We use one channel to encode the question q, and use the other to encode the schema s of SQL table. Note that q is embodied as a free text, though s appears as semi-structured data that comprises a SQL table name and its inner field names, as shown in (1).

(1) **Schema Pattern**: *Table Name* (*Field Name 1*, *Field Name 2*, ..., *Field Name* $\mathcal{N}$)
Example: *Retails.nation* (*Nation key*, *Nation key*, *Orientation*, *Territorial area*, *Population*)

By BGE-M3, we obtain the embeddings of q and s, which are referred to E_q and E_s respectively (E_q, $E_s \in \mathbb{R}^{1\times1,024}$). Thus, we estimate the relevance score by computing the Cosine similarity between E_q and E_s, i.e., r=Cosine(E_q, E_s). Given q, we estimate q-oriented relevance to the schemas of all SQL tables in the SQL database, and rank them in descending order of relevance. On this basis, we adopt top-$\mathcal{K}$ schemas, and regard them as the candidate results. During training, we follow previous works (Niu et al., 2023; Wang et al., 2024) to apply contrastive learning (Hadsell et al., 2006).

Reranker— The reranker serves to re-estimate the relevance scores of the $\mathcal{K}$ candidates for reranking, with the aim to squeeze out top-k (k<$\mathcal{K}$) most reliable results. We construct the reranker by coupling XLM-RoBERTa-large (Conneau et al., 2020) with a two-layer perceptron. Given a candidate schema $\dot{s}$, we concatenate it with q to form the input I (I=[CLS]q[SEP]$\dot{s}$[SEP]), and feed I into XLM-RoBERTa-large to produce hidden state $h \in \mathbb{R}^{1\times1,024}$ of [CLS]. Conditioned on h, the perceptron predicts the relevance score $\dot{r} \in \mathbb{R}^{1\times1}$. During training, we use cross-entropy to calculate loss based on the dataset we constructed.

2.2 Reverse Engineering

Within a reverse engineering process, we infer the potentially-matched question (viz., q_o) for each SQL table, where q_o is required to be a free text. The utilization of q_o in the 2R framework enables the conversion from heterogeneous semantic matching (between semi-structured schema and question) to homogeneous (between questions q and q_o).

We leverage Qwen2.5-7B-Instruct (Yang et al., 2024) to generate q_o for each SQL table. During generation, we feed the schema of each SQL table into Qwen2.5-7B-Instruct, and prompt it to generate q_o that aligns with the schema. To avoid hallucinations like off-topic q_o or free-style output structure, we fine-tune Qwen2.5-7B-Instruct using the training set. In the training set, the gold question aligning with a SQL table is observable. Fine-tuning is performed upon two 3090 GPUs (24G), where LoRA (Hu et al., 2022) is used for optimizing efficiency, and the loss between q_o and gold question is calculated with cross-entropy.

2.3 Conditional "2R" Model

We use the potentially-matched question q_o as an additional condition that constrains the relevance estimation. This constraint is imposed upon the retriever and reranker separately.

For retriever, we compute the similarity $\check{r}$ for the homogeneous questions q and q_o. The computation process is similar to that of baseline, where BGE-M3 is used to produce the embeddings E_q and E_{q_o} for q and q_o, and cosine similarity is calculated, i.e., $\check{r}$=Cosine(E_q, E_{q_o}). Further, we combine $\check{r}$ with the similarity r, i.e., the one obtained for heterogeneous schema s and question q:

$$\mathcal{R} = \alpha \cdot r + (1 - \alpha) \cdot \check{r} \tag{2.3}$$

Dataset	Model	UR@5	UR@10	UR@20	PR@5	PR@10	PR@20
SpiderUnion	Retriever v1 (SGPT 125M) (Zhang et al., 2025)	66.0	75.4	82.2	73.1	80.7	86.3
	Retriever v2 (SGPT 5.8B) (Zhang et al., 2025)	86.8	94.1	97.6	91.5	96.2	98.7
	Retriever v3 (BGE-M3 0.5B)	79.8	86.9	93.9	86.6	91.9	96.4
	Retriever v4 (BGE-M3-finetuned 0.5B)	92.6	97.1	98.6	95.4	98.4	99.3
	Retriever v1+RE	66.7 ↑	76.4 ↑	85.0 ↑	74.2 ↑	81.9 ↑	88.7 ↑
	Retriever v2+RE	86.2	94.7 ↑	97.6	91.4	96.8 ↑	98.8 ↑
	Retriever v3+RE	81.2 ↑	87.7 ↑	94.5 ↑	87.9 ↑	92.6 ↑	97.0 ↑
	Retriever v4+RE	92.9 ↑	96.8	98.5	95.6 ↑	98.3	99.3
BirdUnion	Retriever v1 (SGPT 125M) (Zhang et al., 2025)	50.3	62.1	70.9	63.2	73.3	80.9
	Retriever v2 (SGPT 5.8B) (Zhang et al., 2025)	67.3	79.4	86.4	80.8	88.6	92.8
	Retriever v3 (BGE-M3 0.5B)	56.3	71.7	82.7	72.7	83.0	90.1
	Retriever v4 (BGE-M3-finetuned 0.5B)	76.0	87.4	92.0	87.1	93.4	95.9
	Retriever v1+RE	51.8 ↑	63.1 ↑	72.5 ↑	65.2 ↑	74.9 ↑	82.4 ↑
	Retriever v2+RE	68.8 ↑	80.1 ↑	87.7 ↑	82.1 ↑	89.3 ↑	93.4 ↑
	Retriever v3+RE	60.6 ↑	74.9 ↑	85.3 ↑	76.1 ↑	85.6 ↑	92.2 ↑
	Retriever v4+RE	76.1 ↑	88.1 ↑	92.2 ↑	87.1	93.7 ↑	96.0 ↑

Table 1: Performance obtained when retriever is solely used for SQL table acquisition. The symbol "↑" denotes that the performance is improved when Reverse Engineering (RE) is additionally used to enhance the retriever.

where, α is a dynamic weight which is presented in Appendix A. We preliminarily rank all SQL tables in descending order of $\mathcal{R}$, and adopt $\mathcal{K}$ candidates.

For reranker, all its model structure, inner distributed representation process and training strategy remain unchanged. We only refine the input I, which is expanded with the potentially-matched question q_o, i.e., I=[CLS]q[SEP]$\dot{s}$[SEP]q_o[SEP]. According to the output relevance score $\dot{R}$, we rerank the $\mathcal{K}$ candidates, and adopt top-k (k<$\mathcal{K}$) highly ranked candidates as the final results.

3 Experimentation

3.1 Datasets, Evaluation and Parameters

Datasets— We experiment on SpiderUnion (Kothyari et al., 2023) and BIRDUnion (Kothyari et al., 2023). Both are constructed by merging SQL tables in Spider (Yu et al., 2018) and BIRD (Li et al., 2023b), containing 4,502 and 75 tables respectively, along with 658 and 1,534 questions. All the data in SpiderUnion and BIRDUnion are used for testing. We follow Kothyari et al. (2023) to construct the training set, which contains 18,087 samples, each of which is a self-harmonic tuple of {*question*, *table* and *schema*}.

Evaluation— We follow the previous work (Kothyari et al., 2023; Zhang et al., 2025) to evaluate SQL table acquisition models. Unbroken Recall (UR@k) and Partial Recall (PR@k) are used as the metrics. UR@k is specified as the proportion of positive samples in the top-k acquired results, where a positive sample denotes the one that holds the same unabridged components (table name and field names) as the gold table. The free-style sequential layout of the components is allowed. PR@k is calculated in the same manner, though it allows some of the gold components to be omitted. In a separate experiment, we evaluate the effect of unabridged Text-to-SQL (table acquisition plus SQL query generation), where Execution Accuracy (EX@k) (Yu et al., 2018) is used as the metric, where k indicates that top-k retrieved SQL tables are used as evidence for generating the SQL query.

Hyperparameter Settings— The details of all hyperparameters are presented in Appendix B.

3.2 Main Results

We first evaluate the performance of different retrievers for SQL table acquisition, in which a retriever is solely used without cooperating with any reranker. During evaluation, we compare to the previous work (Zhang et al., 2025) which uses different sizes of SGPT (Muennighoff, 2022) to construct the retriever. Table 1 shows the performance, where RE denotes that Reverse Engineering (RE) used for enhancement. As indicated by the symbol "↑", RE improves the four retriever baselines in nearly all top-k (k=5, 10 and 20) results when BirdUnion is used for testing. Similar situation can be observed when SpiderUnion is used for testing, except the unstable effect on the BGE-M3 fine-tuned by contrastive learning. These test results demonstrate the reliability of RE in bridging semantic gaps across heterogeneous question and table schema.

In a separate experiment, we verify the effect of

Dataset	Model	UR@5	UR@10	UR@20	PR@5	PR@10	PR@20
SpiderUnion	1R Baseline (SGPT 125M) (Zhang et al., 2025)	66.0	75.4	82.2	73.1	80.7	86.3
	+CRUSH (Kothyari et al., 2023)	71.3	80.7	86.8	76.3	83.4	88.9
	+MURRE (Zhang et al., 2025)	74.2	81.0	85.3	77.5	82.3	86.9
	1R Baseline (SGPT 5.8B) (Zhang et al., 2025)	86.8	94.1	97.6	91.5	96.2	98.7
	+CRUSH (Kothyari et al., 2023)	80.1	88.4	92.2	85.1	91.2	94.5
	+MURRE (Zhang et al., 2025)	93.5	96.7	97.3	94.3	96.8	97.5
	2R Baseline (BGE-M3 0.5B)	81.9	91.2	95.6	88.7	94.8	97.8
	+RE+Conditional 2R (Ours)	**93.9**	**97.0**	**98.5**	**97.0**	**98.4**	**99.3**
BirdUnion	1R Baseline (SGPT 125M) (Zhang et al., 2025)	50.3	62.1	70.9	63.2	73.3	80.9
	+CRUSH (Kothyari et al., 2023)	56.1	70.2	77.7	70.0	79.5	86.1
	+MURRE (Zhang et al., 2025)	62.7	72.9	78.3	72.7	79.6	84.2
	1R Baseline (SGPT 5.8B) (Zhang et al., 2025)	67.3	79.4	86.4	80.8	88.6	92.8
	+CRUSH (Kothyari et al., 2023)	63.5	78.4	88.1	77.9	87.5	93.0
	+MURRE (Zhang et al., 2025)	**80.1**	88.7	92.7	87.6	92.6	95.4
	2R Baseline (BGE-M3 0.5B)	62.6	78.1	86.9	77.8	87.6	93.1
	+RE+Conditional 2R (Ours)	79.7	**89.7**	**94.4**	**89.4**	**94.7**	**97.1**

Table 2: Comparing SQL table acquisition performance among different models.

Model	SpiderUnion			BirdUnion		
	EX@5	EX@10	EX@20	EX@5	EX@10	EX@20
SGPT (125M)+CRUSH+GPT3.5 (Kothyari et al., 2023)	50.9	55.9	59.6	17.1	19.0	20.5
SGPT (5.8B)+CRUSH+GPT3.5 (Kothyari et al., 2023)	56.4	60.3	60.8	18.0	19.7	21.1
SGPT (125M)+MURRE+GPT3.5 (Zhang et al., 2025)	54.1	54.7	57.4	16.8	19.4	20.0
SGPT (5.8B)+MURRE+GPT3.5 (Zhang et al., 2025)	64.4	64.4	66.7	21.8	22.0	22.4
BGE-M3 (0.5M)+RE+Conditional 2R+GPT3.5 (Ours)	64.4	**64.6**	**66.9**	21.7	**22.8**	**23.3**
BGE-M3 (0.5B)+RE+Conditional 2R+QwenCoder (Ours)	**68.2**	64.4	63.5	**23.5**	22.0	22.2

Table 3: Comparing Text-to-SQL performance among different models, where EX@k is used for evaluation.

our unabridged approach, where the retriever and reranker in 2R baseline (Section 2.1) are mutually coupled with, and they are respectively enhanced by RE (Section 2.2) and conditional 2R framework (Section 2.3). In this experiment, we compare with the previous work CRUSH (Kothyari et al., 2023) and MURRE (Zhang et al., 2025). They are strong competitors due to their abilities in schema compaction and redundancy elimination. We present them and other related arts in Appendix C.

Table 2 show the test results, where the best performance is highlighted with bold font. It can be observed that our approach (RE plus Conditional 2R) yields substantial improvements compared to the 2R baselines, demonstrating its positive effects (See ablation studies in Appendix D). In addition, our approach outperforms CRUSH and MURRE in most cases. Note that our 2R baseline (0.5B) obtains worse performance than the large version (5.8B) of 1R baseline on two datasets. This indirectly exhibits the considerable benefits from RE and Conditional 2R.

Finally, we compare with CRUSH and MURRE in the complete Text-to-SQL scenario, where the acquired SQL tables are used as evidence for LLM-based SQL query generation. In this experiment, we employ GPT-3.5-turbo (175B) (Ouyang et al., 2022) for query generation, the same as CRUSH and MURRE. Furthermore, we also apply QwenCoder (7B) (Gao et al., 2024) for this task. Table 3 shows the test results (EX@k). It can be observed that our approach enables QwenCoder to achieve better performance in EX@5 rather than EX@20, with overall improved performance demonstrated by our method. This illustrates that the relieving of heterogeneous semantic gap by RE enables a light-weight SQL generator to perform well, conditioned on a smaller number of retrieval results.

4 Conclusion

We propose a reverse engineering based approach to relieve the semantic gaps between heterogeneous questions and SQL tables. Experiments show that our approach achieves substantial improvements. More importantly, it enables a light-weight table acquisition model to effectively support Text-to-SQL using a smaller number of search results.

Concrete SQL contents (i.e., values), as demonstrated previously, contribute to table acquisition. However, the utilization of SQL contents still encounters heterogeneity problem. Therefore, we will generate hybrid groups of SQL schemas and values in the future, and study graph-based reverse engineering for table acquisition and Text-to-SQL.

Limitations

The current version of reverse engineering proceeds in the one-to-one generation manner. Specifically, we leverage LLM to generate the potentially-matched question for a sole table in terms of its schema. However, in some cases, the gold SQL query induced by a question actually aligns with multiple SQL tables. This implies the requirement of decomposing a question into subquestions, and carrying out multi-directional SQL table acquisition. As a result, for such cases, it is difficult to maintain the symmetric information amount between the input content-rich question and the generated exclusive question of a SQL table during text matching. This reveals the requirement of further studying multi-table reverse engineering, where automatically discovering inherently-related tables for information fusion is an indispensable step.

Acknowledgment

The research is supported by National Science Foundation of China (62376182).

References

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024a. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *Preprint*, arXiv:2402.03216.

Peter Baile Chen, Yi Zhang, and Dan Roth. 2024b. Is table retrieval a solved problem? exploring join-aware multi-table retrieval. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2687–2699, Bangkok, Thailand. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

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, volume 1 (long and short papers)*, pages 4171–4186.

Yingqi Gao, Yifu Liu, Xiaoxia Li, Xiaorong Shi, Yin Zhu, Yiming Wang, Shiqi Li, Wei Li, Yuntao Hong, Zhiling Luo, Jinyang Gao, Liyu Mou, and Yu Li. 2024. A preview of xiyan-sql: A multi-generator ensemble framework for text-to-sql. *arXiv preprint arXiv:2411.08599*.

Raia Hadsell, Sumit Chopra, and Yann LeCun. 2006. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE computer society conference on computer vision and pattern recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

George Katsogiannis-Meimarakis, Mike Xydas, and Georgia Koutrika. 2023. Natural language interfaces for databases with deep learning. *Proc. VLDB Endow.*, 16(12):3878–3881.

Hyeonji Kim, Byeong-Hoon So, Wook-Shin Han, and Hongrae Lee. 2020. Natural language to sql: where are we today? *Proc. VLDB Endow.*, 13(10):1737–1750.

Mayank Kothyari, Dhruva Dhingra, Sunita Sarawagi, and Soumen Chakrabarti. 2023. CRUSH4SQL: Collective retrieval using schema hallucination for Text2SQL. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Haoyang Li, Jing Zhang, Cuiping Li, and Hong Chen. 2023a. Resdsql: decoupling schema linking and skeleton parsing for text-to-sql. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'23/IAAI'23/EAAI'23. AAAI Press.

Jinyang Li, Binyuan Hui, Ge Qu, Jiaxi Yang, Binhua Li, Bowen Li, Bailin Wang, Bowen Qin, Ruiying Geng, Nan Huo, Xuanhe Zhou, Chenhao Ma, Guoliang Li, Kevin C.C. Chang, Fei Huang, Reynold Cheng, and Yongbin Li. 2023b. Can llm already serve as a database interface? a big bench for large-scale database grounded text-to-sqls. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA. Curran Associates Inc.

Xinyu Liu, Shuyu Shen, Boyan Li, Peixian Ma, Runzhi Jiang, Yuxin Zhang, Ju Fan, Guoliang Li, Nan Tang, and Yuyu Luo. 2025. A survey of nl2sql with large language models: Where are we, and where are we going? *Preprint*, arXiv:2408.05109.

Niklas Muennighoff. 2022. Sgpt: Gpt sentence embeddings for semantic search. *Preprint*, arXiv:2202.08904.

Kunying Niu, Yifan Wu, Yaohang Li, and Min Li. 2023. Retrieve and rerank for automated icd coding via contrastive learning. *Journal of Biomedical Informatics*, 143:104396.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, and 1 others. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc.

Mohammadreza Pourreza and Davood Rafiei. 2023. Din-sql: decomposed in-context learning of text-to-sql with self-correction. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA. Curran Associates Inc.

Mohammadreza Pourreza and Davood Rafiei. 2024. DTS-SQL: Decomposed text-to-SQL with small large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 8212–8220, Miami, Florida, USA. Association for Computational Linguistics.

Shayan Talaei, Mohammadreza Pourreza, Yu-Chen Chang, Azalia Mirhoseini, and Amin Saberi. 2024. Chess: Contextual harnessing for efficient sql synthesis. *Preprint*, arXiv:2405.16755.

Tianshu Wang, Xiaoyang Chen, Hongyu Lin, Xianpei Han, Le Sun, Hao Wang, and Zhenyu Zeng. 2025. Dbcopilot: Natural language querying over massive databases via schema routing. *Preprint*, arXiv:2312.03463.

Xindi Wang, Robert Mercer, and Frank Rudzicz. 2024. Multi-stage retrieve and re-rank model for automatic medical coding recommendation. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4881–4891, Mexico City, Mexico. Association for Computational Linguistics.

Yuanzhen Xie, Xinzhou Jin, Tao Xie, Matrixmxlin Matrixmxlin, Liang Chen, Chenyun Yu, Cheng Lei, Chengxiang Zhuo, Bo Hu, and Zang Li. 2024. Decomposition for enhancing attention: Improving LLM-based text-to-SQL through workflow paradigm. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 10796–10816, Bangkok, Thailand. Association for Computational Linguistics.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 22 others. 2024. Qwen2.5 technical report. *CoRR*, abs/2412.15115.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir 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*. Association for Computational Linguistics.

Xuanliang Zhang, Dingzirui Wang, Longxu Dou, Qingfu Zhu, and Wanxiang Che. 2025. MURRE: Multi-hop table retrieval with removal for open-domain text-to-SQL. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 5789–5806, Abu Dhabi, UAE. Association for Computational Linguistics.

Dataset	Method	UR@5	UR@10	UR@20	PR@5	PR@10	PR@20
SpiderUnion	Ours	**93.9**	**97.0**	**98.5**	**97.0**	**98.4**	**99.3**
	w/o RE	93.6 ↓	97.0	98.3 ↓	96.7 ↓	98.4	99.2 ↓
	w/ original BGE-M3	86.3 ↓	94.4 ↓	98.2 ↓	91.8 ↓	96.9 ↓	99.1 ↓
	w/o Reranker	92.9 ↓	96.8 ↓	98.5	95.6 ↓	98.3 ↓	99.3
	w/ original Reranker	87.8 ↓	94.1 ↓	96.2 ↓	93.2 ↓	96.8 ↓	98.2 ↓
BirdUnion	Ours	**79.7**	**89.7**	**94.4**	**89.4**	**94.7**	**97.1**
	w/o RE	79.5 ↓	89.5 ↓	94.1 ↓	89.2 ↓	94.5 ↓	97.0 ↓
	w/ original BGE-M3	71.0 ↓	83.2 ↓	89.8 ↓	83.4 ↓	91.0 ↓	94.8 ↓
	w/o Reranker	76.1 ↓	88.1 ↓	92.2 ↓	87.1 ↓	93.7 ↓	96.0 ↓
	w/ original Reranker	66.2 ↓	78.4 ↓	87.1 ↓	80.8 ↓	88.3 ↓	93.3 ↓

Table 4: The ablation results on evaluating our method, compared without Reverse Engineering (RE) (denoted as *w/o RE*), using the BGE-M3 without task-specific fine-tuning (denoted as *w/ original BGE-M3*), without Reranker (*w/o Reranker*), and using the Reranking model without task-specific fine-tuning (denoted as *w/ original Reranker*).

A Dynamic weight for retriever

In this section, we elaborate on constructing the dynamic weight α. A fixed hyperparameter cannot adapt to differences because datasets and retrievers vary. Therefore, we propose a dynamic weight scheme. First, we add E_s and E_{q_o} as joint embedding E_c. Then, we estimate the relevance score between E_q and E_c by compute the Cosine similarity, i.e., r_c=Cosine(E_q, E_c). In retrieval, r_c is oriented by table semantics, with potentially-matched question playing a supplementary role. When r_c is low, it signifies that the table schema and the question are irrelevant, so $1 - r_c$ should be allocated the r. When r_c is high, indicating a relatively strong relevance between the table schema and the question, r_c is assigned to the r. Therefore, the dynamic weight α is formatted as follows:

$$\alpha = max(r_c, 1 - r_c) \tag{A.1}$$

To verify the effectiveness of dynamic weight, we conducted experiments with both dynamic α and various fixed α values on two datasets using four models. The results are shown in the figure 2. The experiments on α show that different models need different α settings on different datasets. In the figure 2, the best α for BGE-M3 is 0.7 on BIRD, but 0.9 on the Spider dataset. However, the dynamic α setting is always optimal or nearly optimal, which shows that it works well. It is also more practical for real-world applications.

B Hyperparameter settings

For the retriever, we set the candidates $\mathcal{K}$ as 40. For the embedding model (BGE-M3), we set the number of negative samples to 25, which are randomly selected from 5-200 top-similarity irrelevant tables. For the reranking model (BGE-ranker-v2-M3), we set the number of negative samples to 10, which are randomly selected from 5-20 top-similarity irrelevant tables. To fine-tune both the embedding model and the reranking model, we use the `FlagEmbedding`[2] framework. For the Reverse Engineering, we adopt the `LLaMA-Factory`[3] framework with LoRA (Hu et al., 2022) technique to fine-tune Qwen2.5-7B-Instruct model (Yang et al., 2024), while setting the LoRA rank to 12.

C Related work

Text-to-SQL plays a crucial role for optimizing data accessibility of SQL (Kim et al., 2020). Recently, Text-to-SQL performance has been increasingly improved due to the emergence of Large Language Models (LLMs) (Liu et al., 2025).

The current studies (Pourreza and Rafiei, 2023; Li et al., 2023a; Xie et al., 2024; Talaei et al., 2024) have explored the approaches of task decomposition, where different data acquisition subtasks are obtained. This allows LLMs to generate relatively-precise solutions by separately handling each sub-task. For example, DTS-SQL (Pourreza and Rafiei, 2024) employs a two-step pipeline, including schema linking and SQL generation.

However, the existing models encounter a new challenge in open-domain Text-to-SQL, where a model struggles with the retrieval of relevant SQL tables from a massive number of tables. The lack of out-of-redundancy and anti-distraction techniques limits the progressive development of open-domain Text-to-SQL. To address the issue, CRUSH (Kothyari et al., 2023) uses an LLM to generate a minimal schema, with the aim to narrow the scope

[2]Document for `FlagEmbedding`
[3]Document for `LLaMA-Factory`

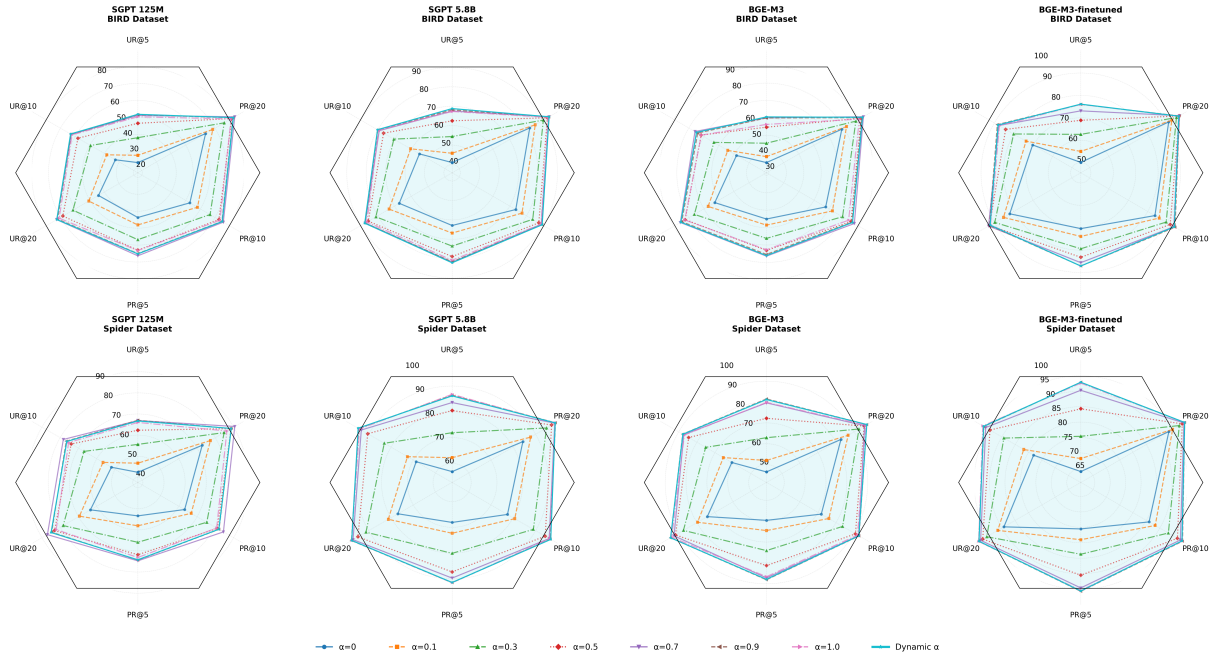

Figure 2: The ablation results on dynamic weight α compared with fixed alpha values across four models and two datasets.

of candidate SQL tables that are potentially relevant to the input question. DBCopilot (Wang et al., 2025) leverages a light-weight differentiable index for relation-aware retrieval of relevant tables. Chen et al. (2024b) proposes a relevance re-ranking approach to optimize the automatic selection of reliable SQL tables, where the dense retrieval models are used and fine-tuned. MURRE (Zhang et al., 2025) conduct iterative redundancy elimination within the multi-hop SQL table retrieval process, where LLM is used to recognize redundant table schemas. This effectively improves the diversity of qualified table schemas within a short ranking list.

D Ablation studies

In this section, we conduct ablation experiments on SpiderUnion and BirdUnion to validate the effectiveness of key components in our approach. As shown in Table 4, removing any module leads to performance degradation, demonstrating their necessity and effectiveness.

D.1 The effectiveness of Reverse Engineering

We evaluate the impact of Reverse Engineering (RE) by comparing our method with a variant that excludes RE. As shown in Table 4, removing RE leads to performance drops across both datasets. These consistent declines demonstrate that RE effectively bridges the semantic gaps across heterogeneous question and table schema.

D.2 The effectiveness of Fine-tuning

We verify the impact of task-specific fine-tuning by comparing our method with variants using original BGE-M3 and Reranker. Using original BGE-M3 leads UR@5 and PR@5 drop by 7.6% and 5.2% on SpiderUnion, and 8.7% and 6.0% on BirdUnion. Similarly, using the original Reranker leads to even larger declines, with UR@5 decreasing by 6.1% on SpiderUnion and 13.5% on BirdUnion. These results demonstrate that task-specific fine-tuning is important and necessary.

D.3 The effectiveness of Reranker

We analyze the role of the Reranker by comparing our method with two variants: one without Reranker and another using the original Reranker. Removing the Reranker causes UR@5 and PR@5 to drop by 1.0% and 1.4% on SpiderUnion, and 3.6% and 2.3% on BirdUnion. Using the original Reranker leads to even larger declines. These results confirm that both the inclusion of the reranker and its task-specific optimization are essential for filtering out noisy candidates and enhancing final accuracy.