DMC-SQL: A Debate-Driven Multi-Agent Collaboration Framework for Mitigating Error Propagation in Text-to-SQL

Anonymous ACL submission

Abstract

Text-to-SQL aims to translate natural language questions into executable SQL queries, enabling users to access relational databases through natural language. In recent years, an increasing number of studies have adopted multi-stage or multi-agent frameworks to decompose the task and improve generation quality. However, these methods often assume that upstream intermediate results are reliable. Once an early-stage result is incorrect, the error may be inherited and amplified by subsequent stages. To reduce this risk, this paper proposes DMC-SQL, a debate-driven multi-agent collaboration framework for mitigating error propagation in Text-to-SQL. DMC-SQL organizes the Text-to-SQL process into a staged pipeline from schema pruning and structural planning to SQL generation and revision, and introduces multi-round debate during each stage to generate more reliable outputs through interactive comparison and iterative revision among multiple agents. In addition, to provide stable discussion units for debate, DMC-SQL further replaces natural language sub-question decomposition with SQL-structure-oriented decomposition, constraining intermediate results to explicit SQL components and making agent outputs easier to align and compare. On the Spider development set, DMC-SQL achieves an execution accuracy of 86.1% using a locally deployed `gpt-oss:20b`. Ablation studies further demonstrate the effectiveness of both the staged debate mechanism and SQL-structure-oriented decomposition.

1 Introduction

Text-to-SQL aims to translate natural language questions into executable SQL queries, enabling users without SQL programming skills to access relational databases (Deng et al., 2022). It has broad application value in business analytics, data management, and intelligent question answering (Liu et al., 2025). Real-world Text-to-SQL often involves cross-domain schema alignment, multi-table joins, aggregation, nested queries, and multiple constraints, making the task still challenging (Iacob et al., 2020).

Recent advances in large language models have significantly promoted Text-to-SQL research. Existing LLM-based methods improve SQL generation through prompt engineering, demonstration selection, reasoning-enhanced prompting, and task-specific generation strategies (Shi et al., 2024). To better handle complex queries, recent studies further organize Text-to-SQL as a multi-stage workflow, including schema selection, question decomposition, SQL generation, and revision (Wang et al., 2025a, 2026). Despite these advances, these methods still face stability challenges in complex query understanding, semantic disambiguation, large-schema scenarios, and multi-step reasoning (Liu et al., 2026).

A remaining limitation is that existing multi-stage or multi-agent Text-to-SQL frameworks usually lack reliable mechanisms for improving intermediate results. Once an intermediate-stage result deviates from the correct semantics, the error may propagate through the pipeline and become further amplified in subsequent generation. Existing studies have also shown that such intermediate stages are important sources of final errors (Heidari et al., 2025). Another concern is that relying on the same model for reflective self-correction may lead to degeneration-of-thought in complex reasoning tasks, where the model, once it forms an initial belief, struggles to produce new evidence for correction (Liang et al., 2024).

This suggests that introspective revision by a single model alone may be insufficient to consistently improve the quality of intermediate decisions. Instead, a more effective solution may require an external and interactive verification mechanism for intermediate results. Prior studies have also shown that multiple model instances can improve reasoning performance and factuality through multi-round

argumentation and interaction (Du et al., 2024). Motivated by this, this paper proposes DMC-SQL, a debate-driven multi-agent collaboration framework for mitigating error propagation in Text-to-SQL. DMC-SQL introduces debate-based multi-agent collaboration at intermediate stages to improve the reliability of intermediate results and reduce error accumulation in subsequent generation.

Natural language sub-question decomposition is also not inherently well suited to multi-agent debate, because its ordering and granularity are often highly flexible, making comparison across agents difficult. DMC-SQL therefore shifts the decomposition target from natural language sub-questions to intermediate representations oriented toward SQL structure, providing more stable units for comparison, debate, and convergence among multiple agents.

Beyond method design, this paper also emphasizes reproducibility and deployability. Many existing Text-to-SQL methods rely on closed-source models or additional fine-tuning, which can limit reproducibility due to model accessibility, data construction, and computational cost. To examine DMC-SQL under a more accessible setting, this paper evaluates it on a locally deployed open-weight quantized model.

In summary, the main contributions of this paper are as follows:

- This paper proposes DMC-SQL, a debate-driven multi-agent collaboration framework for Text-to-SQL, which introduces stage-level debate and adjudication to enhance intermediate-stage reliability and mitigate error propagation in the pipeline.

- This paper introduces a SQL-structure-oriented decomposition strategy to replace flexible natural language sub-question decomposition, providing more stable units for multi-agent debate and convergence.

- Experiments on the Spider development set with a locally deployed open-weight quantized model show that DMC-SQL achieves competitive performance under a low-cost and reproducible setting.

2 Related Work

2.1 Prompt-based Single-Stage SQL Generation

Recent Text-to-SQL research has increasingly adopted large language models as the main generation backbone. These methods often combine few-shot prompting (Brown et al., 2020), Chain-of-Thought (Wei et al., 2022), and example selection to improve generation quality. For example, ACT-SQL (Zhang et al., 2023) automatically constructs chain-of-thought demonstrations for prompt-based Text-to-SQL, while DAIL-SQL (Gao et al., 2024), C3 (Dong et al., 2023), and related studies show that question representation, demonstration organization, and prompt design can substantially affect final SQL generation.

However, these single-stage generation methods still suffer from limited stability in complex queries, long schemas, and cross-table reasoning, and their performance gains often rely on strong closed-source models. To address these limitations, subsequent LLM-based Text-to-SQL research has developed along two representative directions (Zhu et al., 2024; Liu et al., 2025): one improves controllability and stability through task decomposition and multi-stage generation, while the other explores the potential of open-source small models through supervised or decomposed fine-tuning.

2.2 Multi-Stage Generation and Multi-Agent Collaboration

To improve generation quality in complex query scenarios, subsequent studies have organized Text-to-SQL as a multi-stage generation or multi-agent collaboration process. DIN-SQL (Pourreza and Rafiei, 2023) decomposes the task into subtasks such as schema linking, classification, SQL generation, and self-correction to reduce the difficulty of direct generation. MAC-SQL (Wang et al., 2025a) further introduces different agents, including the Selector, Decomposer, and Refiner, for schema reduction, question decomposition, and SQL revision. QCMA-SQL (Shao et al., 2025) selects agents according to question difficulty and incorporates syntax validation and execution optimization, while SQL-of-Thought (Chaturvedi et al., 2025) introduces query planning and error-correction agents to support chained reasoning and dynamic correction. In addition, methods such as SQLFixAgent (Cen et al., 2025) specifically focus on the SQL repair stage, using multi-agent collaboration to improve

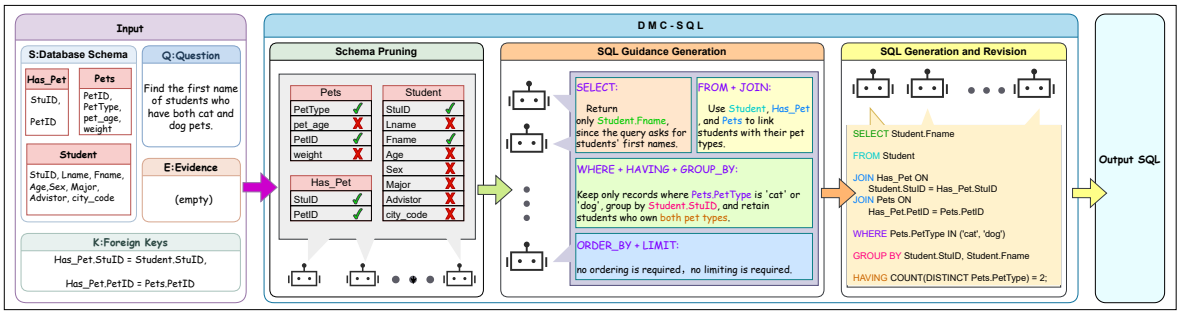

Figure 1: Overall architecture of the DMC-SQL framework.

the semantic consistency of generated results.

Although these methods improve performance through task decomposition, role collaboration, and post-generation revision, their verification is still mostly concentrated after final SQL generation or revision. Upstream results, such as schema selection, question decomposition, and intermediate planning, usually lack independent verification and consistency constraints. As a result, once a deviation occurs in one stage, the error may continue to propagate through the pipeline and become progressively amplified. To address this issue, this paper introduces debate-based multi-agent collaboration and stage-level adjudication into the intermediate stages of multi-stage Text-to-SQL, aiming to improve intermediate-result reliability and mitigate error accumulation.

2.3 Training and Fine-Tuning of Open-Source Small Models

Another line of work reduces reliance on strong closed-source models by improving open-source small models through specialized training or fine-tuning. DTS-SQL (Pourreza and Rafiei, 2024) decomposes complex tasks into easier-to-learn subtasks through two-stage fine-tuning. ROUTE (Qin et al., 2025) combines multi-task supervised fine-tuning with collaborative prompting to enhance open-source models on tasks such as schema linking, noise correction, and SQL generation. STaR-SQL (He et al., 2025) reformulates Text-to-SQL as a reasoning-driven process and improves complex-query performance through fine-tuning on high-quality reasoning trajectories and verifier-based test-time computation. In addition, methods such as SQLCritic (Chen et al., 2025) train dedicated critic models to improve the identification and correction of syntactic and semantic SQL errors.

Overall, these methods show that open-source small models can be substantially improved through targeted training. However, they usually require additional data construction, training pipelines, and computational resources, leading to higher barriers to reproduction and deployment. Different from these approaches, this paper does not rely on large-scale training or high-performance closed-source models, but evaluates the proposed framework on locally deployed open-weight quantized models under a low-cost and easily deployable setting.

3 Methodology

3.1 Overview of the Framework

DMC-SQL is a debate-driven multi-agent framework for Text-to-SQL, designed to enhance the reliability of intermediate results. As illustrated in Figure 1, it introduces multi-agent debate into key stages of the pipeline and decomposes the Text-to-SQL process into three core modules: Schema Pruning, SQL Guidance Generation, and SQL Generation and Revision. The overall generation process can be formulated as follows:

$$S' = f_{\text{prune}}(Q, S, K, E), \quad (1)$$

$$G = f_{\text{guide}}(Q, S', K, E), \quad (2)$$

$$Y = f_{\text{revise}}(Q, S', K, G, E). \quad (3)$$

Here, Q denotes the natural language question, S denotes the full database schema, K denotes foreign key information, and E denotes optional evidence. S' denotes the pruned schema, G denotes the SQL-structure-oriented guidance, and Y denotes the final executable SQL query. The functions $f_{\text{prune}}(\cdot)$, $f_{\text{guide}}(\cdot)$, and $f_{\text{revise}}(\cdot)$ correspond to Schema Pruning, SQL Guidance Generation, and SQL Generation and Revision, respectively.

The following subsections introduce the unified debate mechanism and the specific design of the three core modules described above.

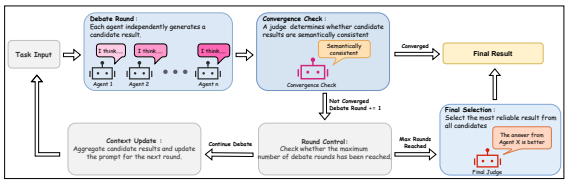

Figure 2: Overview of the multi-agent debate mechanism. Multiple agents generate and revise candidates through multi-round interaction. A convergence judge determines whether to stop early; otherwise, a final judge selects the best candidate after the maximum number of rounds is reached.

3.2 Unified Multi-Agent Debate Mechanism

To improve the stability and consistency of intermediate results, DMC-SQL applies a unified multi-agent debate mechanism to Schema Pruning, SQL Guidance Generation, and SQL Generation and Revision. As illustrated in Figure 2, the mechanism does not simply resample the same input, but lets multiple agents generate candidates in parallel and refine stage-level outputs through multi-round interaction.

For the current stage t, let $X^{(t)}$ denote its input, including the question, schema information, evidence, and previous-stage outputs. The system initializes n agents and conducts at most m debate rounds. Let $z_i^{(r)}$ denote the output of the i-th agent in round r, which contains a candidate result and a brief rationale. The first-round generation and subsequent update process are formulated as:

$$z_i^{(1)} = A_i\left(X^{(t)}\right), \quad i \in \{1, 2, \ldots, n\}, \tag{4}$$

$$z_i^{(r)} = A_i\left(X^{(t)}, \mathcal{Z}_{-i}^{(r-1)}\right), \quad r \geq 2, \tag{5}$$

where $A_i(\cdot)$ denotes the generation function of the i-th agent, and $\mathcal{Z}_{-i}^{(r-1)} = \{z_j^{(r-1)} \mid j \neq i\}$ denotes the outputs generated by all other agents in the previous round.

This update enables agents to incorporate complementary information from other candidates while keeping the original task conditions unchanged. Since different candidates may differ in local decisions such as table and column selection, structural planning, or SQL organization, multi-round interaction helps improve the consistency and stability of stage outputs (Wang et al., 2023; Li et al., 2024).

To avoid unconstrained cyclic interaction, DMC-SQL applies a convergence judge after each round and a final judge when convergence is not reached within m rounds:

$$c^{(r)} = J_{\text{cons}}\left(z_1^{(r)}, z_2^{(r)}, \ldots, z_n^{(r)}\right), \quad c^{(r)} \in \{0, 1\}, \tag{6}$$

$$z^{(t)} = J_{\text{final}}\left(X^{(t)}, z_1^{(m)}, z_2^{(m)}, \ldots, z_n^{(m)}\right). \tag{7}$$

Here, $J_{\text{cons}}(\cdot)$ denotes the convergence judge, which determines whether the current-round candidates are semantically consistent. $c^{(r)} = 1$ indicates that the candidates in round r have reached semantic agreement, in which case the stage terminates early. If convergence is not reached within m rounds, $J_{\text{final}}(\cdot)$ selects the most reliable candidate from the last round based on both the stage input $X^{(t)}$ and the candidate outputs.

Overall, the mechanism combines parallel generation, multi-round interaction, convergence judgment, and final selection. It makes stage outputs explicitly comparable and subject to adjudication, reducing the risk of passing local errors to downstream modules.

3.3 Schema Pruning Module

The main motivation for introducing this module is that, when the full schema is large, irrelevant tables and columns may increase the risk of selecting incorrect schema items in later stages. Therefore, pruning the original schema helps retain a question-relevant local structure and reduces the interference of irrelevant schema information during subsequent generation (Wang et al., 2025c; Yang et al., 2024).

As defined in Equation 1, the Schema Pruning module takes the question Q, the full schema S, foreign key information K, and optional evidence E as input, and outputs a compact schema subset S' for subsequent SQL guidance generation and SQL generation.

In this paper, the pruned schema skeleton is denoted as $S' = \{T', C'\}$, where $T' \subseteq T$ denotes the retained tables and $C' \subseteq C$ denotes the retained columns associated with these tables.

In implementation, multiple agents independently produce candidate schema subsets with brief rationales. These candidates are then processed by the unified debate mechanism in Section 3.2 to obtain the final pruned schema.

When producing the final S', the system explicitly preserves relevant primary key and foreign key information. This prevents the local schema

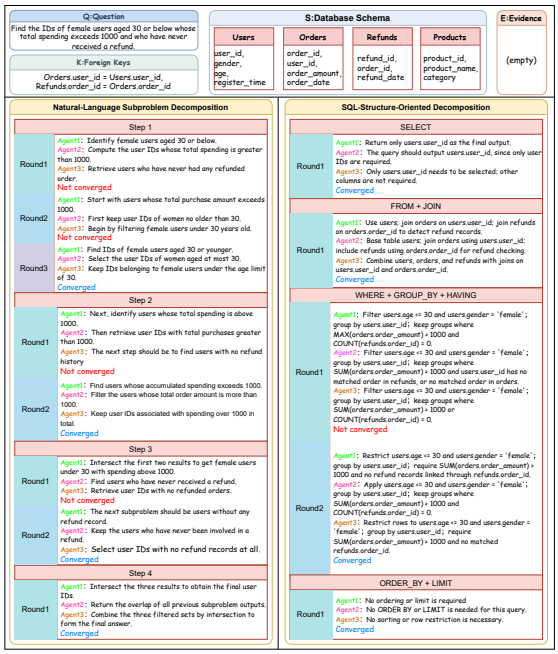

Figure 3: Comparison between natural language sub-question decomposition and SQL-structure-oriented decomposition. The latter organizes intermediate planning around fixed SQL components, making agent outputs easier to align, compare, and converge.

from losing necessary join constraints during compression, keeping the resulting schema skeleton compact and structurally complete.

3.4 SQL Guidance Generation Module

Existing studies have shown that decomposing the Text-to-SQL task into a series of intermediate subtasks or intermediate representations can improve the reasoning performance of large language models in complex query scenarios. Compared with directly generating the final SQL from natural language in a single step, this approach provides clearer intermediate constraints and a more controllable generation process (Guo et al., 2019; Dai et al., 2025; Wang et al., 2025b; Chaturvedi et al., 2025).

As defined in Equation 2, the SQL Guidance Generation module takes the question Q, the pruned schema S', foreign key information K, and optional evidence E as input, and outputs a SQL-structure-oriented planning result G. This guidance specifies the key structural decisions required to construct the final SQL.

Unlike natural language sub-question decomposition commonly used in existing methods, DMC-SQL performs explicit planning around the internal structure of SQL. Natural language sub-questions are often flexible in order, granularity, and linguistic form, so different agents may produce substantially different intermediate results even when their understanding is correct. This flexibility makes candidate outputs difficult to align, compare, and judge for semantic consistency in multi-agent debate. In contrast, SQL itself has clearer structural boundaries, making SQL-structure-oriented decomposition more suitable for comparison, revision, and convergence judgment. Figure 3 illustrates the difference between conventional natural language sub-question decomposition and the SQL-structure-oriented decomposition adopted in DMC-SQL. Section 4 further compares this design with a natural language sub-question decomposition variant through ablation experiments.

Based on the above considerations, this paper organizes intermediate planning into structured guidance units centered on key SQL components, rather than free-form natural language sub-questions. Let the output of this module be denoted as G:

$$G = \{g_{\text{target}}, g_{\text{source}}, g_{\text{constraint}}, g_{\text{presentation}}\} \quad (8)$$

Here, g_{target} describes the return target, g_{source} describes the data source and join relationships, $g_{\text{constraint}}$ describes filtering, grouping, and aggregation constraints, and $g_{\text{presentation}}$ describes result ordering and return scope. These guidance units capture key SQL construction decisions for subsequent stages.

3.5 SQL Generation and Revision Module

As defined in Equation 3, the SQL Generation and Revision module generates the final executable SQL query Y based on the question Q, the pruned schema S', SQL guidance G, foreign key information K, and optional evidence E. This module integrates candidate SQL generation, execution feedback, and revision into a unified process, continuously updating and refining the final SQL.

In each generation round, multiple agents generate candidate SQL queries in parallel with brief rationales. For the i-th candidate SQL y_i, its execution feedback is denoted as $e_i = \text{Exec}(y_i)$, where $\text{Exec}(\cdot)$ denotes the feedback returned by executing the candidate SQL on the target database, including execution results, error messages, empty-result signals, or other database feedback useful for revision.

The system feeds both candidate SQL queries and their execution feedback into subsequent debate rounds. In this way, agents revise their outputs not only according to linguistic self-consistency, but also according to actual execution outcomes from the database (Mao et al., 2024).

During revision, agents can use execution feedback to adjust schema usage, join paths, filtering conditions, aggregation constraints, or result limits. After each round, the system applies the convergence judgment in Equation 6. If the candidates have reached agreement, the revision process terminates early; otherwise, when the maximum number of rounds is reached, the final judge in Equation 7 selects the final SQL by considering both candidate queries and their execution feedback.

4 Experiments

4.1 Experimental Setup

4.1.1 Dataset

Experiments are conducted on the Spider dataset (Yu et al., 2018), a representative cross-domain Text-to-SQL benchmark. Spider contains 10,181 natural language questions and 5,693 unique SQL queries over 200 databases across 138 domains. Since its train, development, and test splits use non-overlapping databases, it is widely used to evaluate cross-domain generalization on unseen schemas. This paper follows the standard Spider setting and reports results on the development set.

4.1.2 Evaluation Metric

Execution Accuracy (EX) from the official Spider evaluation protocol is used as the evaluation metric. Since a natural language question may correspond to multiple semantically equivalent SQL queries, surface-form or structure-level matching may not fully reflect query correctness. EX instead compares the execution results of the predicted SQL and the gold SQL, making it more suitable for evaluating practical Text-to-SQL performance.

4.1.3 Baselines

To evaluate the effectiveness of DMC-SQL, experiments compare it with two groups of representative baselines. The first group consists of single-model direct generation baselines, including GPT-4 (OpenAI, 2023) and `gpt-oss:20b` (OpenAI, 2025), which generate SQL through zero-shot prompting without multi-agent collaboration or explicit intermediate planning. The second group includes non-fine-tuned LLM-based Text-to-SQL methods, including ACT-SQL (Zhang et al., 2023), DIN-SQL (Pourreza and Rafiei, 2023), DAIL-SQL (Gao et al., 2024), MAC-SQL (Wang et al., 2025a), and SQLFixAgent (Cen et al., 2025). These methods improve SQL generation through in-context reasoning, task decomposition, demonstration retrieval, multi-agent collaboration, or SQL repair.

Model	Method	EX
GPT-4	Zero-Shot	74.2
	DIN-SQL	82.8
	ACT-SQL	82.9
	DAIL-SQL	83.1
	MAC-SQL	**86.8**
`gpt-oss:20b`	Zero-Shot	72.2
	DIN-SQL	77.9
	ACT-SQL	78.6
	MAC-SQL	79.7
	DAIL-SQL	82.1
	SQLFixAgent + CodeS:3B	85.6
	DMC-SQL	**86.1**

Table 1: Main results of different methods on the Spider dev set under different backbone models.

4.1.4 Implementation Details

The proposed method is evaluated on a machine equipped with a single NVIDIA GeForce RTX 3090 GPU with 24 GB of GPU memory. The quantized models used in the experiments are deployed locally through Ollama. Unless otherwise specified, the generation temperature is uniformly set to 0.8. For the multi-agent debate framework proposed in this paper, the debate-stage configuration is kept consistent: each round involves three agents in parallel, and at most three rounds of debate are conducted.

4.2 Main Results

Table 1 presents the execution accuracy results of different methods on the Spider development set under different backbone models. The results show that, within the locally deployed `gpt-oss:20b` setting, DMC-SQL achieves the highest execution accuracy among the compared methods. This indicates that the proposed framework can effectively improve Text-to-SQL generation performance under a relatively low deployment cost.

In addition, although DMC-SQL is not evaluated with GPT-4 in this paper due to cost constraints, its execution accuracy on `gpt-oss:20b` is already close to, and in some cases slightly higher

Method	Easy	Medium	Hard	Extra	All
DMC-SQL + `gpt-oss:20b`	**95.2**	**85.9**	**86.8**	**72.3**	**86.1**
w/ NL sub-question decomposition	91.9	81.6	78.2	65.1	80.9
w/o debate in Schema Pruning	94.4	82.7	78.7	65.1	82.0
w/o debate in SQL Guidance Generation	93.1	84.3	80.5	62.0	82.2
w/o debate in SQL Generation and Revision	93.1	83.9	76.4	63.9	81.6

Table 2: Results of the ablation study of DMC-SQL on the Spider dev set.

Max Debate Rounds	Easy	Medium	Hard	Extra	All
0	94.4	84.1	79.3	61.4	82.1
1	93.5	84.1	81.6	67.5	83.3
2	95.2	85.2	**87.4**	69.9	85.5
3	**95.2**	**85.9**	86.8	**72.3**	**86.1**

Table 3: Results of DMC-SQL on the Spider dev set under different numbers of debate rounds.

than, the results of several representative GPT-4-based methods. This suggests that, even when using a smaller quantized model that can be deployed locally, DMC-SQL is still able to achieve performance comparable to, or even slightly better than, some GPT-4-based approaches, thereby narrowing the gap with strong closed-source baselines.

4.3 Ablation Study

Table 2 presents the ablation results of DMC-SQL on `gpt-oss:20b`. The experiments compare the full framework with several variants, including replacing SQL-structure-oriented decomposition with natural language sub-question decomposition and removing the debate mechanism from different modules.

The results first show that replacing SQL-structure-oriented decomposition with natural language sub-question decomposition leads to a clear performance drop. As discussed in Section 3.4, natural language sub-questions are highly flexible in ordering, granularity, and wording, making them harder to align and compare during multi-agent debate. Such flexibility may introduce unnecessary variation into the debate process, whereas SQL-structure-oriented decomposition provides more regular planning units for comparison, revision, and convergence.

To further examine the role of debate in improving stage-level reliability, this paper removes the debate mechanism from each core module separately. Removing debate from Schema Pruning mainly affects medium, hard, and extra-hard questions, suggesting that complex queries are more sensitive to schema selection errors. Removing debate from SQL Guidance Generation causes clear declines on hard and extra-hard examples, which indicates that structural planning benefits from multi-agent comparison when the query logic becomes more complex. Removing debate from SQL Generation and Revision also weakens performance, especially on hard examples, showing that final SQL generation still requires iterative checking and correction. These results demonstrate that debate is not only useful at the final revision stage, but also helps stabilize upstream schema selection and SQL-structure planning.

Overall, the ablation results show that both SQL-structure-oriented decomposition and the multi-stage debate mechanism are important to DMC-SQL, since replacing the decomposition strategy or removing debate from any module consistently degrades performance.

4.4 Effect of Maximum Debate Rounds

In addition to component-level ablations, this paper further analyzes the effect of maximum debate rounds. Table 3 reports the execution accuracy of DMC-SQL on the Spider development set under different maximum debate-round settings. Overall, increasing the maximum number of rounds from 0 to 3 leads to a steady improvement in overall performance, suggesting that multi-round debate helps agents refine intermediate results through interaction. The best result is obtained when the maximum number of debate rounds is set to 3, which is therefore used as the default setting in this paper.

To examine whether the debate mechanism helps mitigate error propagation from intermediate stages to final SQL generation, this paper analyzes the outputs of Schema Pruning and SQL Guidance Generation on all 1,034 Spider development examples. The outputs are categorized into success, neutral, and fail cases according to whether they can effectively support the generation of the final SQL, and the number of final SQL errors is counted within each category.

Figure 4 shows that, as the maximum number

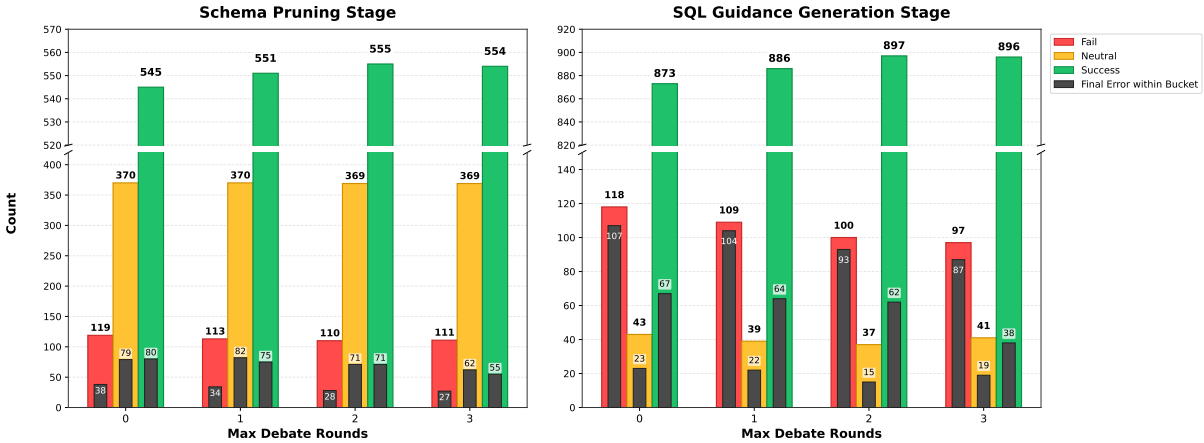

Figure 4: Effects of different maximum debate-round settings on intermediate-stage results. Colored bars show fail, neutral, and success cases in Schema Pruning and SQL Guidance Generation, while black bars indicate final errors within each category.

of debate rounds increases, the number of fail cases generally decreases, while the number of success cases increases in both stages. Compared with the setting without debate, setting the maximum number of debate rounds to 3 reduces the number of fail cases by 6.7% in Schema Pruning and by 17.8% in SQL Guidance Generation. More importantly, debate also improves the downstream reliability of intermediate results. Among the cases categorized as success, the final error rate decreases by 32.4% in Schema Pruning and by 44.7% in SQL Guidance Generation. These results indicate that the debate mechanism not only improves the stage-level correctness of intermediate outputs, but also reduces the risk of error propagation to subsequent SQL generation.

5 Conclusion

In summary, this paper proposes DMC-SQL, a debate-driven multi-agent collaboration framework for mitigating error propagation in Text-to-SQL. The proposed method introduces a multi-agent debate mechanism into key stages of the pipeline, including Schema Pruning, SQL Guidance Generation, and SQL Generation and Revision. It also replaces highly flexible natural language subquestion decomposition with an SQL-structure-oriented decomposition strategy, providing more stable units for debate, comparison, and convergence. By improving the reliability of intermediate outputs, DMC-SQL reduces error accumulation across the multi-stage generation process. Experimental results show that DMC-SQL achieves an execution accuracy of 86.1 on the Spider development set with locally deployed gpt-oss:20b, demonstrating strong effectiveness under a low-cost local deployment setting. Further intermediate-stage analysis shows that debate reduces stage-level failures and lowers the risk of final SQL errors caused by unreliable intermediate results. Future work will further examine the generalization ability of the proposed method on stronger backbone models and broader evaluation scenarios.

Limitations

This paper still has several limitations. First, the experiments are mainly conducted on a locally deployable open-weight quantized model, and equivalent evaluations on frontier closed-source models are not performed, so comparisons with some GPT-4-style methods under the same conditions remain insufficient. Second, multi-agent debate substantially increases the number of model calls and the context length, which leads to higher token cost and inference latency. Third, when the backbone model has weak reasoning or verification ability, debate may amplify errors or produce erroneous consensus. Fourth, the SQL-structure-oriented decomposition used in this paper is mainly designed for data retrieval queries. It does not explicitly cover broader SQL operations such as table creation, data insertion, data update, or deletion. Therefore, although the proposed decomposition strategy improves the stability of query-oriented SQL generation, its applicability to more diverse database interaction scenarios remains to be further explored.

References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA. Curran Associates Inc.

Jipeng Cen, Jiaxin Liu, Zhixu Li, and Jingjing Wang. 2025. Sqlfixagent: towards semantic-accurate text-to-sql parsing via consistency-enhanced multi-agent collaboration. In *Proceedings of the Thirty-Ninth AAAI Conference on Artificial Intelligence and Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence and Fifteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'25/IAAI'25/EAAI'25. AAAI Press.

Saumya Chaturvedi, Aman Chadha, and Laurent Bindschaedler. 2025. Sql-of-thought: Multi-agentic text-to-sql with guided error correction. Accepted at NeurIPS 2025 DL4C "Deep Learning for Code" Workshop.

Jikai Chen, Leilei Gan, Ziyu Zhao, Zechuan Wang, Dong Wang, and Chenyi Zhuang. 2025. Sqlcritic: Correcting text-to-sql generation via clause-wise critic. *Preprint*, arXiv:2503.07996.

Yaxun Dai, Haiqin Yang, Mou Hao, and Pingfu Chao. 2025. PARSQL: Enhancing text-to-SQL through SQL parsing and reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 661–681, Vienna, Austria. Association for Computational Linguistics.

Naihao Deng, Yulong Chen, and Yue Zhang. 2022. Recent advances in text-to-SQL: A survey of what we have and what we expect. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 2166–2187, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Xuemei Dong, Chao Zhang, Yuhang Ge, Yuren Mao, Yunjun Gao, lu Chen, Jinshu Lin, and Dongfang Lou. 2023. C3: Zero-shot text-to-sql with chatgpt. *Preprint*, arXiv:2307.07306.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. 2024. Improving factuality and reasoning in language models through multiagent debate. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Dawei Gao, Haibin Wang, Yaliang Li, Xiuyu Sun, Yichen Qian, Bolin Ding, and Jingren Zhou. 2024. Text-to-sql empowered by large language models: A benchmark evaluation. *Proc. VLDB Endow.*, 17(5):1132–1145.

Jiaqi Guo, Zecheng Zhan, Yan Gao, Yan Xiao, Jian-Guang Lou, Ting Liu, and Dongmei Zhang. 2019. Towards complex text-to-SQL in cross-domain database with intermediate representation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4524–4535, Florence, Italy. Association for Computational Linguistics.

Mingqian He, Yongliang Shen, Wenqi Zhang, Qiuying Peng, Jun Wang, and Weiming Lu. 2025. STaR-SQL: Self-taught reasoner for text-to-SQL. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 24365–24375, Vienna, Austria. Association for Computational Linguistics.

Omid Reza Heidari, Siobhan Reid, and Yassine Yaakoubi. 2025. AGENTIQL: An agent-inspired multi-expert framework for text-to-sql generation. *arXiv preprint arXiv:2510.10661*.

Radu Cristian Alexandru Iacob, Florin Brad, Elena-Simona Apostol, Ciprian-Octavian Truică, Ionel Alexandru Hosu, and Traian Rebedea. 2020. Neural approaches for natural language interfaces to databases: A survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 381–395, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Yunxuan Li, Yibing Du, Jiageng Zhang, Le Hou, Peter Grabowski, Yeqing Li, and Eugene Ie. 2024. Improving multi-agent debate with sparse communication topology. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 7281–7294, Miami, Florida, USA. Association for Computational Linguistics.

Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Shuming Shi, and Zhaopeng Tu. 2024. Encouraging divergent thinking in large language models through multi-agent debate. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 17889–17904, Miami, Florida, USA. Association for Computational Linguistics.

Mengyi Liu, Xieyang Wang, Jianqiu Xu, Weijia Yi, and Ouri Wolfson. 2026. A systematic review of natural language interfaces for databases. *Front. Comput. Sci.*, 20(11).

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 text-to-sql in the era of llms: Where are we, and where are we going? *IEEE Transactions on Knowledge and Data Engineering*, 37(10):5735–5754.

Wenxin Mao, Ruiqi Wang, Jiyu Guo, Jichuan Zeng, Cuiyun Gao, Peiyi Han, and Chuanyi Liu. 2024. Enhancing text-to-SQL parsing through question rewriting and execution-guided refinement. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 2009–2024, Bangkok, Thailand. Association for Computational Linguistics.

OpenAI. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

OpenAI. 2025. gpt-oss-120b & gpt-oss-20b model card. https://openai.com/index/gpt-oss-model-card/. Model card, published August 5, 2025.

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.

Yang Qin, Chao Chen, Zhihang Fu, Ze Chen, Dezhong Peng, Peng Hu, and Jieping Ye. 2025. Route: Robust multitask tuning and collaboration for text-to-sql.

Zhihui Shao, Shubin Cai, Rongsheng Lin, and Zhong Ming. 2025. Enhancing text-to-SQL with question classification and multi-agent collaboration. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 4340–4349, Albuquerque, New Mexico. Association for Computational Linguistics.

Liang Shi, Zhengju Tang, Nan Zhang, Xiaotong Zhang, and Zhi Yang. 2024. A survey on employing large language models for text-to-sql tasks. *ACM Computing Surveys*, 58:1 – 37.

Bing Wang, Changyu Ren, Jian Yang, Xinnian Liang, Jiaqi Bai, LinZheng Chai, Zhao Yan, Qian-Wen Zhang, Di Yin, Xing Sun, and Zhoujun Li. 2025a. MAC-SQL: A multi-agent collaborative framework for text-to-SQL. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 540–557, Abu Dhabi, UAE. Association for Computational Linguistics.

Dingzirui Wang, Longxu Dou, Xuanliang Zhang, Qingfu Zhu, and Wanxiang Che. 2025b. DAC: Decomposed automation correction for text-to-SQL. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 385–402, Suzhou, China. Association for Computational Linguistics.

Qin Wang, Youhuan Li, Suixi Lin, Zhou Tang, Kenli Li, Peng Peng, Quanqing Xu, and Chuanhui Yang. 2026. MCI-SQL: Text-to-sql with metadata-complete context and intermediate correction. *arXiv preprint arXiv:2603.13390*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *ICLR 2023*.

Yihan Wang, Peiyu Liu, and Xin Yang. 2025c. LinkAlign: Scalable schema linking for real-world large-scale multi-database text-to-SQL. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 977–991, Suzhou, China. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

Sun Yang, Qiong Su, Zhishuai Li, Ziyue Li, Hangyu Mao, Chenxi Liu, and Rui Zhao. 2024. Sql-to-schema enhances schema linking in text-to-sql. In *Database and Expert Systems Applications*, pages 139–145, Cham. Springer Nature Switzerland.

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*, pages 3911–3921, Brussels, Belgium. Association for Computational Linguistics.

Hanchong Zhang, Ruisheng Cao, Lu Chen, Hongshen Xu, and Kai Yu. 2023. ACT-SQL: In-context learning for text-to-SQL with automatically-generated chain-of-thought. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 3501–3532, Singapore. Association for Computational Linguistics.

Xiaohu Zhu, Qian Li, Lizhen Cui, and Yongkang Liu. 2024. Large language model enhanced text-to-sql generation: A survey. *Preprint*, arXiv:2410.06011.