Bidirectional Semantic Enhancement for Schema Routing Across Large-Scale Databases

Yuyang Wu[1,2,4], **Xiaoliang Wang**[1,2], **Cam-Tu Nguyen**[†1,3]

[1]State Key Laboratory of Novel Software Technology, Nanjing University
[2]School of Computer Science, Nanjing University
[3]School of Artificial Intelligence, Nanjing University
[4]Guodian Nanjing Automation CO.,LTD.

Emails: wuyuyang@smail.nju.edu.cn; {waxili, ncamtu}@nju.edu.cn

Abstract

With the prevalence of Large Language Models (LLMs), Text-to-SQL has made significant progress, yet applying it to massive, real-world databases remains a challenge. While previous works adopt a retrieve-then-generate framework, they struggle with the profound *semantic gap* between user queries and vague schema definitions. Existing methods relying on unidirectional query expansion often fail to bridge lexical mismatches, while graph-based approaches struggle to navigate schemas when explicit structural links (e.g., foreign keys) are missing. To address this, we propose **Bi-SR**, a retrieval framework that bridges this gap through a bidirectional semantic enhancement strategy. We simultaneously enrich vague table schemas offline and perform online generative query expansion—specifically predicting potential schema structures—to align user intent. Crucially, we introduce a dual-augmented contrastive training objective for the dense retriever, which trains the dense retriever to recognize the semantic correspondence between the LLM-expanded query intent and the detailed schema descriptions. Experiments on massive schema routing benchmarks constructed from BIRD and Spider demonstrate that Bi-SR achieves state-of-the-art performance and significantly empowers smaller models for cost-effective deployment.

1 Introduction

The advent of Large Language Models (LLMs) has fundamentally transformed Text-to-SQL parsing, enabling intuitive natural language interfaces for complex databases. While state-of-the-art systems demonstrate impressive proficiency on standard benchmarks, they predominantly operate under an idealized assumption where the target database is known a priori and the schema fits within the context window (Li et al., 2024c; Lee et al., 2025). However, this premise fails to hold in real-world enterprise environments, such as data lakes and warehouses, which often encompass thousands of tables (Nargesian et al., 2019). In such massive scenarios, *Schema Routing* (Wang et al., 2025b), also known as large-scale schema linking, becomes the critical step for efficiently identifying a minimal, relevant subset of tables from a vast repository to ground subsequent generation.

The primary bottleneck in schema routing is not merely the scale of the search space, but the profound *semantic gap* between user intent and database definitions (Yu et al., 2019; Floratou et al., 2024). User queries typically employ colloquial terms (e.g., "profit"), whereas real-world schemas often utilize abbreviated, technical, or vague identifiers (e.g., `t_fin_05`) that lack explicit semantic cues (Gan et al., 2021). Standard retrieval methods, which rely on lexical overlap or surface-level embedding similarity, often fail to bridge this gap (Li et al., 2024a; Talaei et al., 2024). Furthermore, existing solutions typically attempt to mitigate this via unidirectional alignment—either by augmenting the query with hypothetical schemas or by traversing static schema graphs (Chen et al., 2024; Kothyari et al., 2023; Shi et al., 2025; Wang et al., 2025b; Toteja et al., 2025). These approaches often fall short when the underlying schema is inherently cryptic or when structural links are missing, leading to low recall and subsequent generation failures. As illustrated in Figure 1(A), user queries typically employ colloquial terms (e.g., "profit"), whereas real-world schemas often utilize abbreviated or vague identifiers (e.g., `t_fin_05`) that lack explicit semantic cues. Standard retrieval methods fail to bridge this gap due to lexical mismatches.

To address these challenges, we propose **Bi-SR** (Bidirectional Semantic Routing), a retrieval framework designed to robustly bridge the semantic gap in massive databases. As shown in Figure 1(B), Our method introduces a novel Two-Level Bidirec-

[†]Corresponding author.

Findings of the Association for Computational Linguistics: ACL 2026, pages 7495–7509

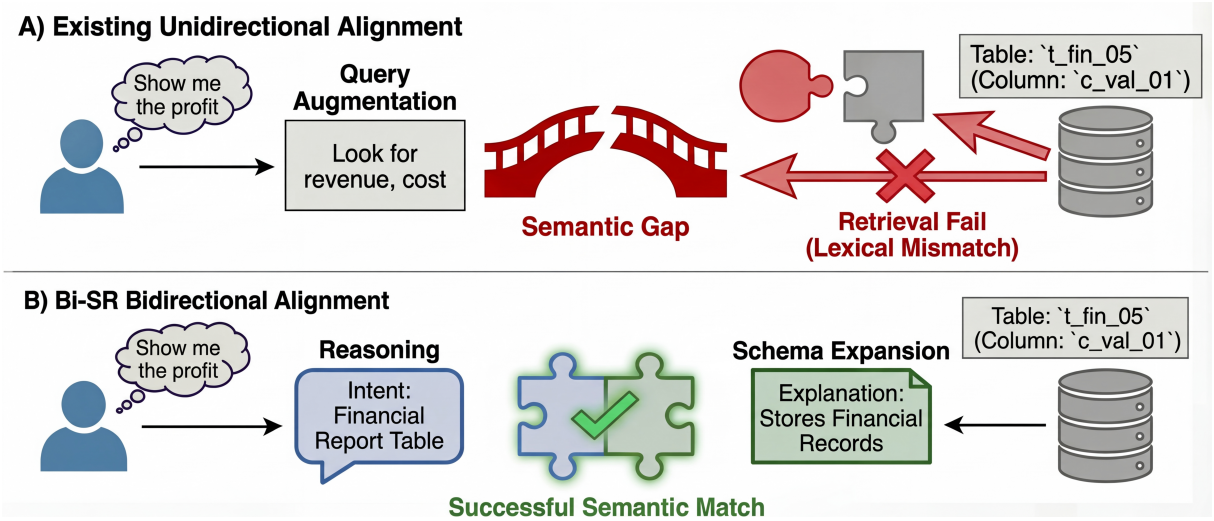

Figure 1: Motivation Analysis. (A) Existing unidirectional methods struggle to bridge the *semantic gap* when user queries (e.g.,"profit") lack lexical overlap with opaque schema definitions (e.g., `t_fin_05`), leading to retrieval failure. (B) **Bi-SR** employs a **bidirectional alignment** strategy: it expands schema semantics offline (right) and infers query intent online (left), creating a shared semantic space where the "puzzle pieces" perfectly align.

tional Semantic Enhancement strategy that aligns the semantic space of queries and schemas from two directions: offline schema-side expansion that enriches vague definitions with semantic explanations, and online query-side reasoning that deduces potential schema structures. To equip dense retrieval with the domain database information as well as the ability for targeted query expansion, we propose a dual-augmented training objective, that allows a dense retriever to mimic the semantic expansion patterns of a Large Language Model.

Comprehensive evaluations on massive schema routing benchmarks constructed from BIRD and Spider demonstrate that Bi-SR achieves state-of-the-art performance, substantially outperforming existing baselines in both retrieval recall and downstream execution accuracy. Further analysis confirms that the proposed strategy effectively mitigates the noise in massive search spaces, offering a highly robust solution for navigating vague and complex real-world database schema.

2 Related Work

2.1 Schema Linking and Routing

Schema linking is a critical prerequisite for Text-to-SQL, aiming to identify database elements (tables and columns) relevant to a natural language question (Nargesian et al., 2019). Early approaches (Zhang and Ives, 2020; Fan et al., 2023) relied on rule-based matching or syntactic parsing, which often struggled with the lexical mismatch between user queries and database schemas (Guo et al., 2019; Bogin et al., 2019). With the advent of Large Language Models (LLMs), recent works have focused on leveraging LLMs to enhance schema linking (Chen et al., 2021; Cai et al., 2021). For instance, Solid-SQL (Liu et al., 2025) utilizes growth-based heuristic searches to filter irrelevant columns, while RSL-SQL (Cao et al., 2024) employs a bidirectional linking strategy to recall schema elements within a single database. Similarly, X-SQL (Peng, 2025) introduces X-Linking, an SFT-based module, to improve linking precision. However, these methods primarily address the schema filtering problem within a single or a small number of databases.

In real-world scenarios involving massive databases (e.g., data warehouses with thousands of tables), determining the target database and tables—defined as *Schema Routing*—becomes the primary bottleneck. DBCopilot (Wang et al., 2025b) pioneered this direction by constructing a schema graph to model relationships between tables and navigating through the graph to locate relevant schemas. More recently, LinkAlign (Wang et al., 2025c) proposed a framework for large-scale multi-database environments, utilizing multi-round semantic enhanced retrieval to filter irrelevant databases. Despite their effectiveness, methods like DBCopilot (Wang et al., 2025b) and In-Context RL (Toteja et al., 2025) rely on heavy graph construction or computationally expensive iterative interactions. In contrast, our work proposes a streamlined vector-based routing framework that

achieves superior scalability through efficient bidirectional semantic alignment.

2.2 Retrieval-Augmented Text-to-SQL

Retrieval-Augmented Generation (RAG) has been widely adopted to bridge the gap between user queries and external knowledge (Lewis et al., 2020). In Text-to-SQL, retrieval is often used to fetch relevant schema items or few-shot examples to fit within the LLM's context window (Shi et al., 2025; He et al., 2025). A key challenge in retrieval-based Text-to-SQL is the *semantic gap* between natural language questions and rigid database schemas (e.g., abbreviations or opaque column names). CRUSH4SQL (Kothyari et al., 2023) attempts to mitigate this by using an LLM to hallucinate a minimal schema, which is then used to retrieve the actual schema. Building on this, Gen-SQL (Shi et al., 2025) introduces a "Pseudo-Schema" generation step, prompting LLMs to predict potential table structures based on the query before retrieval.

However, these approaches typically treat the alignment process as one-way (Query $\rightarrow$ Predicted Schema). If the underlying database schema itself lacks semantic clarity (e.g., containing codes like $t_0 1$ instead of $transaction$), query-side augmentation alone is insufficient. OpenSearch-SQL (Xie et al., 2025) attempts to address hallucination through consistency alignment but does not explicitly tackle the bidirectional semantic gap in retrieval. Our approach advances this paradigm by introducing a Bidirectional Semantic Enhancement strategy. We not only augment the query with potential table descriptions but also enrich the database schema offline with LLM-generated explanations and aliases. This ensures that both the query and the schema meet in a shared, semantically rich embedding space.

3 Methodology

Preliminaries The problem of Text-to-SQL over massive databases, often referred to as Schema Routing, poses unique challenges compared to standard Text-to-SQL. Formally, let $\mathcal{D} = \{T_1, T_2, \ldots, T_N\}$ be a massive repository of database tables, where N can scale to thousands. Each table T_i consists of a table name t_i and a set of columns $\mathcal{C}_i = \{c_{i,1}, c_{i,2}, \ldots, c_{i,m}\}$. Given a natural language question q, the goal is to identify a minimal subset of relevant tables $\mathcal{S}^* \subset \mathcal{D}$ and their associated columns that contain the necessary information to answer q. Finally, a SQL generation model M_{gen} takes q and the routed schema $\mathcal{S}^*$ to generate the executable SQL query y.

3.1 Framework Overview

The proposed **Bi-SR** framework illustrated in Figure 2. We structure the framework into two distinct phases to ensure both retrieval precision and inference efficiency: an **Offline Phase** for schema enrichment and dual-augmented training, and an **Online Phase** for real-time semantic reasoning, retrieval, and fine-grained pruning.

3.2 Offline Phase: Semantic Expansion and Dense Retrieval Training

The offline phase focuses on constructing a semantically rich embedding space. This involves expanding opaque schema definitions into descriptive text and training a dense retriever to align user intent with these descriptions.

3.2.1 Schema-Side Semantic Expansion

Raw DDL (Data Definition Language) statements are often insufficient for semantic retrieval due to their sparse information (Codd, 1970). To enhance the document-side representation, we transform each raw table schema into a semantically rich textual description. For every table $T_i \in \mathcal{D}$, we construct a structured triplet representation $D_{T_i} = \langle \text{DatabaseID}, \text{TableID}, \text{Explanation} \rangle$. We utilize a Large Language Model (LLM) to generate the `Explanation` component based on the table's schema and sampled values. This generated content specifically encompasses a functional description of the table's utility and contextual keywords describing the target database domain. This expansion is performed once, converting the discrete schema space $\mathcal{D}$ into an indexed vector space $\mathcal{V}_{\mathcal{D}}$ ready for efficient retrieval.

3.2.2 Dual-Augmented Training

Core Intuition. Standard retrieval aligns a raw question q with a raw schema s. However, there is a mismatch: q implies an implicit *schema-aware intent*, while s contains hidden business logic. *Our key innovation is to explicitly "materialize" these hidden semantics on both sides.* We construct a "Dual-Augmented" objective where the model learns to match a *schema-aware expanded query* (augmented with potential schema descriptions) against a *semantic-rich schema* (augmented with functional explanations).

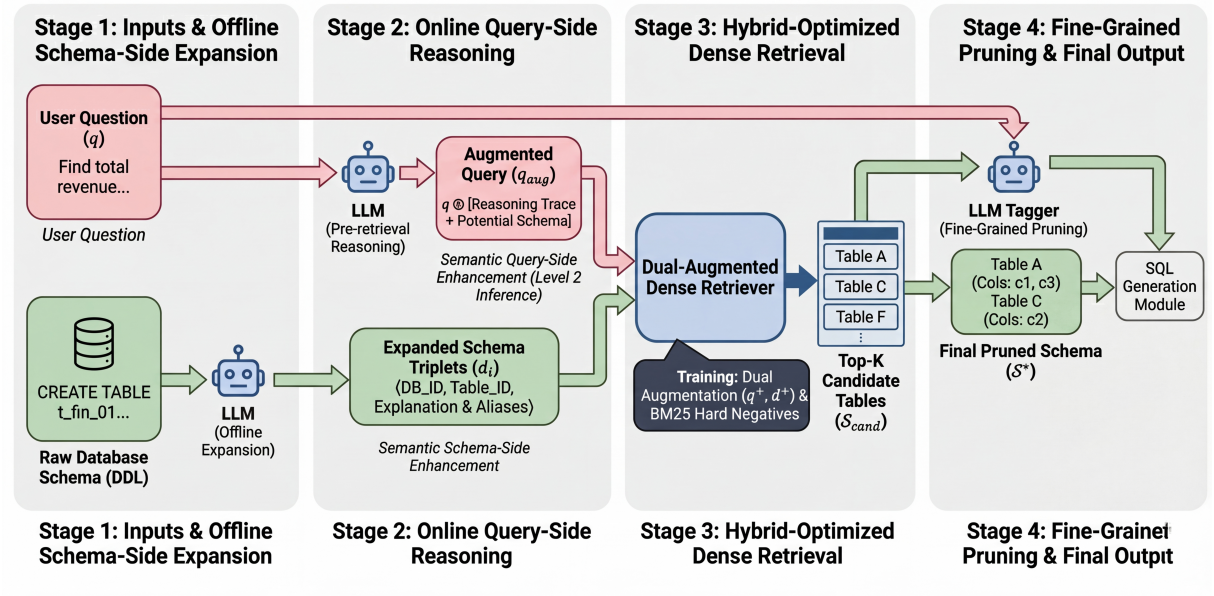

Figure 2: Overview of Bi-SR Framework. SQL Generation Module here means any LLM or generation model that Bi-SR gives schema input

Data augmentation. As discussed above, standard dense retrieval training often fails to capture the knowledge required to link ambiguous questions to specific table structures. To address this, we construct a Dual-Augmented training objective using the training set triplets $\langle q, y, \mathcal{D} \rangle$.

We first parse the gold SQL y to identify the ground truth table T_{gold}, establishing the positive document as its pre-computed explanation triplet $d^+ = d_{T_{gold}}$. Simultaneously, we construct an augmented query representation that combines the user's intent with explicit schema reasoning. Using the ground truth table T_{gold} and database metadata $\mathcal{D}$ as context, we prompt an LLM to generate a "Target Potential Schema." Specifically, the LLM is instructed to generate the table description and potential names for the table T_{gold} that answers question q. Let this generated content be P_{gold}. Formally, the progress above can be described as:

$$P_{gold} = \text{LLM}\left(\mathcal{I}_{gen}(q, T_{gold}, \mathcal{D})\right) \tag{1}$$

The instruction $\mathcal{I}_{gen}$ is explicitly designed to bridge the semantic gap by forcing the model to explain the function of the table. The augmented training query is defined as:

$$q^+ = q \oplus [\text{SEP}] \oplus P_{gold} \tag{2}$$

Training with Contrastive Learning To enhance the model's discriminative power, we adopt a hybrid hard negative mining strategy using BM25 (Robertson et al., 2009). For each training query q^+, we select the top-K retrieved tables that are *not* the ground truth table T_{gold} as the hard negative set $\mathcal{N}_{hard}$. We then fine-tune the embedding model using the InfoNCE (Oord et al., 2018) loss:

$$\mathcal{L} = -\log \frac{e^{\mathbf{e}_q \cdot \mathbf{e}_d / \tau}}{e^{\mathbf{e}_q \cdot \mathbf{e}_d / \tau} + \sum_{n \in \mathcal{N}_{neg}} e^{\mathbf{e}_q \cdot \mathbf{e}_n / \tau}} \tag{3}$$

where τ is a temperature hyperparameter, $\mathcal{N}_{neg}$ denotes the union of hard negatives and in-batch negatives. This objective optimizes the embedding space such that semantically aligned query-schema pairs are pulled closer while lexically similar but incorrect pairs are pushed apart.

3.3 Online Phase: Reasoning, Retrieval, and Pruning

During the online phase, the system processes a new query q through a pipeline designed to resemble the semantic alignment available in training.

Pre-retrieval Reasoning Since the ground truth table is unknown during inference, we employ a Pre-retrieval Reasoning step (Nogueira et al., 2019) to bridge the gap. We instruct the LLM to predict the schema required to answer the user's question q. This process involves a two-step Chain-of-Thought (CoT) (Wei et al., 2022) generation: the LLM first analyzes the question to identify key entities, relationships, and constraints (Reasoning

Stage	Raw Input	Augmented Representation
Schema Side (Offline)	**Raw DDL:** DatabaseID: `california_schools` TableID: `frpm`	**Expanded Triplet (d_i):** ***Database***: california_schools, ***Table***: frpm *"**Description***: School meal program eligibility and participation data in Education performance and demographics.
Query Side (Training)	**User Question (q):** "What is the highest eligible free rate for K-12 students in the schools in Alameda County?" **Gold Table:** `frpm` in Database `california_schools`	**Dual-Augmented Query (q^+):** ***Query***: What is the highest eligible free rate for K-12 students in the schools in Alameda County? ***Potential tables***: - frpm: Stores school meal program information including participation data, potential names:[Frpm, Frpm_Info, Frpm_Details, Educational_Institutions]
Query Side (Inference)	**User Question (q):** "Please list the lowest three eligible free rates for students aged 5-17 in continuation schools." *(Gold Table Unknown)*	**Pre-retrieval Reasoning Query (q_{aug}):** ***Query***: Please list the lowest three eligible free rates for students aged 5-17 in continuation schools. ***Potential tables***: - student_rates: Stores rate information for students including rate amount and eligibility criteria, potential names:[Rate, Rates, Student_Rates, Fee_Rates, Rate_Structures, Eligibility_Rates]... **Approximates the training distribution.*

Table 1: An example of **Bidirectional Semantic Enhancement**. Note that during training, we use the *Ground Truth Table* to construct an "Ideal Reasoning" trace, forcing the retriever to learn the mapping from user intent to table semantics. During inference, the LLM generates a similar trace.

Trace), and subsequently outputs a list of hypothetical table names and functional descriptions (Potential Schema Generation). Let the generated reasoning and potential schema be P_{pred}. The augmented inference query is formulated as:

$$q_{test} = q \oplus [\text{SEP}] \oplus P_{pred} \tag{4}$$

This mechanism transforms the retrieval task from a difficult "Raw Query $\rightarrow$ Schema" into a robust "Predicted Schema $\rightarrow$ Actual Schema" alignment. Since P_{pred} approximates the P_{gold} used in training, the domain shift is minimized.

Dense Retrieval The augmented query q_{test} is encoded by our fine-tuned dense retriever to obtain the query embedding. We then perform a maximum inner product search against the pre-computed offline index $\mathcal{V}_{\mathcal{D}}$ to recall the top-K candidate tables $\mathcal{S}_{cand}$. This step efficiently filters the massive search space down to a manageable subset of semantically relevant tables.

Fine-Grained Schema Pruning While retrieval ensures recall, the top-K candidates (e.g., $K = 5$) often introduce irrelevant columns that can confuse the SQL generator or exceed context limits. To mitigate this, we perform a fine-grained pruning step, inspired by the schema grounding phase in Gen-SQL (Shi et al., 2025). We define a pruning function $\mathcal{P}$ parameterized by an LLM, where the input consists of the original question q and the full DDL statements of the candidate tables $\mathcal{S}_{cand}$:

$$\mathcal{S}^* = \mathcal{P}_{LLM}(q, \mathcal{S}_{cand}) \tag{5}$$

The LLM acts as a filter, identifying and outputting only the "Important Columns" required to answer q. This helps SQL generation model focus on the important information but without losing other potential useful information. Finally, the refined schema $\mathcal{S}^*$ and the question q are fed into the SQL generation model to produce the final SQL query.

4 Experiments

This section evaluates Bi-SR through three research questions focused on: (1) the retrieval effectiveness of Bi-SR in course (table-level) and fine-grained (column-level) retrieval, (2) the impact of improved schema routing on Text-to-SQL execution accuracy, and (3) the robustness and efficiency of Bi-SR in out-of-distribution generalization.

4.1 Experimental Setup

Datasets and Setup We evaluate Bi-SR on two benchmarks, BIRD (Li et al., 2024b) and Spider (Yu et al., 2018). To simulate real-world large-scale environments, we adopt the Massive Schema Routing setting. Unlike standard evaluations that as-

sume known database identifiers, we discard identifiers and construct a unified retrieval pool comprising all available tables (597 for BIRD and 1020 for Spider). This setup requires the system to identify relevant schemas from a massive, mixed-domain repository without prior knowledge. We provide comprehensive dataset statistics and detailed experimental configurations in Appendix B.

Baselines To rigorously evaluate performance, we compare Bi-SR against three categories of representative methods: (1) Sparse and Dense Retrievers, including BM25, SXFMR (Reimers and Gurevych, 2019), and the zero-shot Qwen3-Embedding (Zhang et al., 2025); (2) LLM-Augmented Retrievers, including DTR (Herzig et al., 2021), CRUSH4SQL (Kothyari et al., 2023), and Gen-SQL (Shi et al., 2025); (3) Graph-based and Agentic Methods, specifically DBCopilot (Wang et al., 2025b) and ICRL (Toteja et al., 2025). Detailed descriptions and specific configurations for each baseline are provided in **Appendix B.4.**

Implementation Details We implement our framework using PyTorch and the HuggingFace Transformers library. All baselines are implemented by the same models as our method. For the retrieval module, we initialize the dense retriever with `Qwen3-embedding-4B` (Zhang et al., 2025), fine-tuning it for 3 epochs with a batch size of 32, while mining 5 hard negatives per query using BM25 to enhance discriminative power. We employ `DeepSeek-V3.2-Exp` (DeepSeek-AI, 2025) as the primary Large Language Model (LLM) to drive the Offline Schema Explanation, Online Pre-retrieval Reasoning, and Final SQL Generation modules, owing to its superior reasoning capabilities and cost-effectiveness. Furthermore, to evaluate the framework's adaptability in resource-constrained environments (as discussed in Section C), we conduct additional experiments utilizing the smaller `Qwen3-8B` (Yang et al., 2025) model for the bidirectional semantic enhancement tasks. Both `DeepSeek-V3.2-Exp` and `Qwen3-8B` are set to non-thinking mode.

Evaluation Metrics. We employ distinct metrics to assess both retrieval and generation performance. For schema routing, we report **Recall@K** (R@K), measuring the proportion of ground-truth tables covered within the top-K retrieved candidates. To rigorously evaluate ranking precision, we specifically introduce **R@e**, defined as the exact set match rate where the top-e retrieved tables must perfectly align with the ground-truth set of size e. For downstream SQL generation, we adhere to standard benchmark protocols: on Spider, we report Execution Accuracy (**EX**) and Exact Match (**EM**) to evaluate functional and structural correctness; on BIRD, we report EX and the Valid Efficiency Score (**VES**), which additionally accounts for the execution efficiency of valid SQL queries in large-scale database environments.

Method	Spider			BIRD		
	R@e	R@5	R@15	R@e	R@5	R@15
Sparse & Standard Dense Retrievers						
BM25	48.07	86.60	93.87	34.98	68.32	82.81
SXFMR	52.36	80.42	92.39	35.66	67.56	83.05
Qwen3-Embed	59.32	85.17	95.48	58.50	82.23	92.60
LLM-Augmented Methods						
DTR	52.67	76.27	93.18	56.97	76.24	91.96
CRUSH	59.06	87.91	95.06	40.16	68.37	87.82
Gen-SQL	60.62	86.54	95.64	60.17	82.56	92.85
Opensearch-SQL	57.02	84.91	93.27	58.17	82.70	87.15
Graph-based Methods						
DBCopilot	49.94	85.34	87.25	34.30	61.02	62.14
ICRL	56.92	86.42	88.34	51.63	82.40	92.24
Ours						
Bi-SR	**65.83**	**88.65**	**96.48**	**62.54**	**87.16**	**94.25**

Table 2: Schema routing performance (Table Retrieval) on Spider and BIRD dev datasets under the massive database setting. We report **R@e** (Exact Set Match at rank e, where e is the number of gold tables), **R@5**, and **R@15**. Note that R@e is a strict metric measuring if the top-e retrieved tables perfectly match the gold set.

4.2 Schema Routing Performance

Table Retrieval As presented in Table 2, Bi-SR achieves state-of-the-art performance across all metrics on both Spider and BIRD datasets. While Gen-SQL performs competitively by augmenting the query side with pseudo-schemas, Bi-SR outperforms it by 5.21% and 2.37% in R@e on Spider and BIRD respectively, demonstrating that offline schema-side expansion is critical for bridging the semantic gap that query augmentation alone cannot resolve. Additionally, graph-based methods like DBCopilot exhibit significant performance degradation on BIRD (dropping to 34.30% R@e) due to their reliance on explicit foreign keys which are often missing or implicit in real-world "noisy" databases. In contrast, Bi-SR maintains high robustness in large-scale schema routing where structural links may be incomplete.

Column Retrieval In addition to table-level routing, we evaluate the granularity of our system through Column Retrieval performance, as detailed

Method	Spider		BIRD	
	R@e	R@15	R@e	R@15
Sparse & Standard Dense Retrievers				
Qwen3-Embed	65.4	92.2	63.1	87.7
BM25	32.1	54.8	47.3	49.6
LLM-Augmented Methods				
CRUSH4SQL	51.2	88.7	53.8	90.1
Gen-SQL	70.6	95.2	68.7	94.3
Graph-based Methods				
DBCopilot	44.1	65.7	40.3	67.0
ICRL	43.3	63.9	37.2	65.2
Ours				
Bi-SR	**72.1**	**97.2**	**69.3**	**96.1**

Table 3: Column Retrieval Performance on Spider and BIRD dev datasets. For Bi-SR, this measures the recall of columns retained after the pruning stage.

Method	Spider		BIRD	
	EX	EM	EX	VES
LLM-based Schema Linking & Full Schema				
MAC-SQL(Wang et al., 2025a)	0.9	0.6	1.7	3.2
DIN-SQL(Pourreza and Rafiei, 2023)	8.3	7.1	7.2	2.4
Deepseek-V3.2 w/ Full Schema	13.4	6.7	3.2	14.2
LLM-Augmented Retriever				
DTR	54.4	23.5	19.5	18.5
CRUSH	55.6	28.1	17.6	19.2
Gen-SQL	67.1	21.7	28.4	35.1
Graph-based Methods				
DBCopilot	61.5	31.2	21.4	26.5
ICRL	69.6	33.5	20.7	27.7
Ours				
Bi-SR	**71.4**	**35.1**	**30.0**	**59.1**

Table 4: End-to-End Text-to-SQL Generation Performance on Spider and BIRD dev datasets under the massive database setting. All methods here use `DeepSeek-V3.2-Exp` as backbone LLM.

Variant	Table Retrieval		Generation	
	R@e	R@5	EX	EM
Bi-SR (Full)	**65.8**	**88.7**	**71.4**	**35.1**
w/o Query Augmentation	60.7	86.2	65.7	33.1
w/o Schema Expansion	59.4	83.1	68.2	32.6
w/o Column Pruning	N/A	N/A	66.2	31.7
w/o Training	60.1	85.3	71.4	35.1

Table 5: Ablation study of different components in Bi-SR on Spider dev set.

in Table 3. To ensure a fair and comprehensive comparison, we benchmark against all baseline methods in our experimental setup that explicitly support column-level granularity, excluding coarse-grained retrievers (e.g., BM25, standard Dense Retrievers) that lack fine-grained filtering mechanisms. For Bi-SR, this setting is to verify the effectiveness of the fine-grained pruning module.

As observed, Bi-SR achieves superior precision, recording the highest R@e of 72.1% on Spider. The contrast with graph-based methods is particularly stark: DBCopilot and ICRL struggle significantly at the column level (e.g., DBCopilot reaches only 44.1% R@e on Spider), as these methods primarily model inter-table relationships and lack the granular semantic understanding required to distinguish relevant attributes within a table.

4.3 Text-to-SQL Performance

This section reveals the critical necessity of schema routing in massive database settings and the advantage of our framework in the downstream task. We standardize the input context by providing the top-5 retrieved tables ($K = 5$) unless otherwise noted.

The experimental results are shown in Table 4. The most striking observation is the catastrophic performance degradation of traditional LLM-based methods that rely on full schema input; state-of-the-art systems like MAC-SQL and DIN-SQL fail to function effectively (e.g., MAC-SQL yields only 0.9% EX on Spider). This collapse confirms that simply scaling LLM context windows is insufficient for massive schema routing due to severe noise interference. In contrast, retrieval-augmented approaches successfully recover performance, with Bi-SR achieving the highest Execution Accuracy of 71.4% on Spider and 30.0% on BIRD. This significant lead over Gen-SQL (67.1% on Spider) validates that our bidirectional semantic enhancement provides a more accurate schema subset than unidirectional augmentation, directly translating higher recall into better SQL generation.

Furthermore, the results highlight Bi-SR's substantial advantage in handling realistic, complex environments. We observe a notable performance gap between the two datasets: BIRD yields significantly lower accuracy than Spider (30.0% vs. 71.4%) despite comprising fewer total tables in the merged pool. This counter-intuitive finding underscores that the primary challenge in real-world schema routing is not merely the quantity of tables, but the semantic ambiguity and "noisy" content inherent in complex databases.

4.4 Ablation Study

To rigorously assess the contribution of each module within our framework, we conduct an ablation study on the Spider development set, analyzing both retrieval recall (R@e, R@5) and downstream

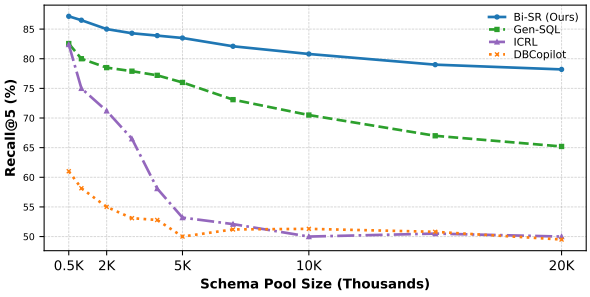

Figure 3: Scalability & OOD Generalization Analysis

generation quality (EX, EM), as summarized in Table 5. Note that for the "w/o Query Augmentation" variant, we re-trained the retriever using only raw (q, d) pairs to strictly isolate the impact of reasoning-based alignment.

The results reveal distinct roles for our bidirectional components. *Offline Schema Expansion proves most critical for retrieval precision*; removing it causes the sharpest drop in Exact Set Match (R@e falls from 65.8% to 59.4%). This confirms that without semantically rich descriptions, the retriever struggles to map queries to opaque table identifiers (e.g., `t_fin_01`). On the other hand, *Query Augmentation in Pre-retrieval Reasoning stage appears vital for bridging the gap to final SQL generation*. While its absence moderately impacts retrieval (R@e drops to 60.7%), it causes the most catastrophic degradation in Execution Accuracy (EX drops by 5.7% to 65.7%). This suggests that the "reasoning trace" not only aids in finding tables but also implicitly primes the system with a clearer understanding of user intent, which is crucial for the subsequent generation phase.

Finally, we investigate the impact of the Fine-Grained Column Pruning stage. As this module operates on the candidate set $\mathcal{S}_{cand}$ *after* the retrieval phase, its removal does not affect retrieval metrics (marked as "-" in Table 5). However, eliminating pruning results in a substantial decline in downstream performance (EX falls to 66.2%), comparable to removing core retrieval augmentations.

Further analyses of parameter setting and small model performance are given in Appendix C.

4.5 Scalability and Out-of-Distribution Generalization

To rigorously evaluate the framework's robustness in open-world scenarios, we conduct a large-scale generalization experiment where the retrieval search space is progressively expanded. In this setting, all models are trained exclusively on the BIRD dataset to ensure no prior exposure to the test domains. During evaluation, we incrementally introduce unseen databases from the Spider and WikiSQL (Zhong et al., 2017) datasets into the retrieval pool, expanding the total schema size from 0.5K to 20K tables. This setup challenges the models to not only scale to larger noise levels but also generalize to completely out-of-distribution schemas without additional fine-tuning.

Figure 3 reveals a fundamental divergence between semantic retrieval paradigms and generative or policy-based routing methods in open-world settings. As the schema pool expands from 0.5K to 20K tables with unseen domains, methods like DBCopilot and ICRL suffer a catastrophic performance collapse, plunging to near-random retrieval rates (approximately 50%). This failure underscores the inherent limitation of generative routing: such models rely on parametrically "memorizing" schema identifiers or overfitting generation policies to specific training domains. In contrast, retrieval-based approaches (Bi-SR and Gen-SQL) exhibit significantly stronger robustness, confirming that mapping user intent to semantic representations is the only viable pathway for handling the scale and diversity of realistic data warehouses.

Crucially, within the robust retrieval paradigm, Bi-SR demonstrates superior resilience compared to Gen-SQL, with the performance gap widening markedly as the noise level increases. While Gen-SQL's unidirectional query augmentation provides a baseline level of generalization, its reliance on the LLM to zero-shot generate potential table structures becomes fragile when facing the diverse and often opaque naming conventions of 20,000 heterogeneous tables. Bi-SR overcomes this bottleneck by establishing stable "semantic anchor points" through offline schema-side expansion. This allows Bi-SR to maintain a high Recall@5 of over 78% even in the most extreme 20K-table setting, effectively neutralizing the distraction posed by massive unseen schemas.

5 Conclusion

In this paper, we introduced Bi-SR, a robust framework designed to address the critical challenge of schema routing over massive databases. By implementing a novel bidirectional semantic enhancement strategy, we effectively bridged the semantic gap between colloquial user queries and opaque database schemas. Our dual-augmented training

objective helps the dense retrieval match a *schema-aware expanded query* against a *semantic-rich schema*, enabling deep semantic alignment. Extensive experiments on the BIRD and Spider benchmarks under rigorous open-world settings demonstrate that Bi-SR achieves state-of-the-art performance in both retrieval recall and downstream execution accuracy. Furthermore, our analysis demonstrates that Bi-SR maintains strong generalization on a noisy dataset of 20K tables, exhibiting only a minor performance drop.

Acknowledgements

This work was supported in part by the Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China (No. JYB2025XDXM118) and NSFC 62532010, W2532049. It was also supported by Guodian Nanjing Automation CO., LTD. through the project "Research on Key Technologies for a Fully Domestic Power-Dedicated Large-Scale Model Toolchain."

Limitations

Despite these advancements, our framework operates under certain constraints. First, the quality of our offline schema expansion is inherently bound by the knowledge capacity of the backbone LLM used for data synthesis. In extremely niche or highly proprietary domains where the LLM lacks prior exposure, the generated descriptions may suffer from hallucinations, potentially misleading the retriever. Second, our current routing mechanism primarily focuses on schema-level semantics (table and column definitions) and does not explicitly index database cell values. Consequently, the system may struggle with queries that rely exclusively on exact value matching for disambiguation (e.g., distinguishing between tables based solely on specific row contents) without explicit schema cues. Future work will explore integrating value-aware retrieval mechanisms and lightweight domain adaptation techniques to further enhance robustness in such edge cases.

References

Ben Bogin, Jonathan Berant, and Matt Gardner. 2019. Representing schema structure with graph neural networks for text-to-SQL parsing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4560–4565.

Ruichu Cai, Jinjie Yuan, Boyan Xu, and Zhifeng Hao. 2021. Sadga: Structure-aware dual graph aggregation network for text-to-sql. *Advances in Neural Information Processing Systems*, 34:7664–7676.

Zhenbiao Cao, Yuanlei Zheng, Zhihao Fan, Xiaojin Zhang, Wei Chen, and Xiang Bai. 2024. Rsl-sql: Robust schema linking in text-to-sql generation. *arXiv preprint arXiv:2411.00073*.

Peter Baile Chen, Yi Zhang, and Dan Roth. 2024. 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.

Zhi Chen, Lu Chen, Yanbin Zhao, Ruisheng Cao, Zihan Xu, Su Zhu, and Kai Yu. 2021. ShadowGNN: Graph projection neural network for text-to-SQL parser. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5567–5577.

Edgar F. Codd. 1970. A relational model of data for large shared data banks. *Communications of the ACM*, 13:377–387.

DeepSeek-AI. 2025. Deepseek-v3.2-exp: Boosting long-context efficiency with deepseek sparse attention.

Grace Fan, Jin Wang, Yuliang Li, and Renée J. Miller. 2023. Table discovery in data lakes: State-of-the-art and future directions. In *Companion of the 2023 International Conference on Management of Data*, page 69–75.

Avrilia Floratou, Fotis Psallidas, Fuheng Zhao, Shaleen Deep, Gunther Hagleither, Wangda Tan, Joyce Cahoon, Rana Alotaibi, Jordan Henkel, Abhik Singla, and 1 others. 2024. Nl2sql is a solved problem... not! In *Proceedings of the 14th Conference on Innovative Data Systems Research*.

Yujian Gan, Xinyun Chen, Qiuping Huang, Matthew Purver, John R. Woodward, Jinxia Xie, and Pengsheng Huang. 2021. Towards robustness of text-to-SQL models against synonym substitution. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2505–2515.

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. *arXiv preprint arXiv:1905.08205*.

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.

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.

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*, pages 14054–14066.

Dongjun Lee, Choongwon Park, Jaehyuk Kim, and Heesoo Park. 2025. Mcs-sql: Leveraging multiple prompts and multiple-choice selection for text-to-sql generation. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 337–353.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Haoyang Li, Jing Zhang, Hanbing Liu, Ju Fan, Xiaokang Zhang, Jun Zhu, Renjie Wei, Hongyan Pan, Cuiping Li, and Hong Chen. 2024a. Codes: Towards building open-source language models for text-to-sql. *Proceedings of the ACM on Management of Data*, 2:1–28.

Jinyang Li, Binyuan Hui, Ge Qu, Jiaxi Yang, Binhua Li, Bowen Li, Bailin Wang, Bowen Qin, Ruiying Geng, Nan Huo, and 1 others. 2024b. Can llm already serve as a database interface? a big bench for large-scale database grounded text-to-sqls. *Advances in Neural Information Processing Systems*, 36.

Zhishuai Li, Xiang Wang, Jingjing Zhao, Sun Yang, Guoqing Du, Xiaoru Hu, Bin Zhang, Yuxiao Ye, Ziyue Li, Rui Zhao, and 1 others. 2024c. Petsql: A prompt-enhanced two-round refinement of text-to-sql with cross-consistency. *arXiv preprint arXiv:2403.09732*.

Geling Liu, Yunzhi Tan, Ruichao Zhong, Yuanzhen Xie, Lingchen Zhao, Qian Wang, Bo Hu, and Zang Li. 2025. Solid-SQL: Enhanced schema-linking based in-context learning for robust text-to-SQL. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 9793–9803.

Fatemeh Nargesian, Erkang Zhu, Renée J. Miller, Ken Q. Pu, and Patricia C. Arocena. 2019. Data lake management: challenges and opportunities. *Proceedings of the VLDB Endowment*, 12:1986–1989.

Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019. Document expansion by query prediction. *arXiv preprint arXiv:1904.08375*.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Dazhi Peng. 2025. X-sql: Expert schema linking and understanding of text-to-sql with multi-llms. *arXiv preprint arXiv:2509.05899*.

Mohammadreza Pourreza and Davood Rafiei. 2023. Din-sql: Decomposed in-context learning of text-to-sql with self-correction. *arXiv preprint arXiv:2304.11015*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. 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)*, pages 3982–3992.

Stephen Robertson, Hugo Zaragoza, and 1 others. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3:333–389.

Jie Shi, Bo Xu, Jiaqing Liang, Yanghua Xiao, Jia Chen, Chenhao Xie, Peng Wang, and Wei Wang. 2025. Gen-SQL: Efficient text-to-SQL by bridging natural language question and database schema with pseudo-schema. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 3794–3807.

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

Rishit Toteja, Arindam Sarkar, and Prakash Mandayam Comar. 2025. In-context reinforcement learning with retrieval-augmented generation for text-to-SQL. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 10390–10397.

Bing Wang, Changyu Ren, Jian Yang, Xinnian Liang, Jiaqi Bai, Linzheng Chai, Zhao Yan, Qian-Wen Zhang, Di Yin, Xing Sun, and 1 others. 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.

Tianshu Wang, Xiaoyang Chen, Hongyu Lin, Xianpei Han, Le Sun, Hao Wang, and Zhenyu Zeng. 2025b. Dbcopilot: Natural language querying over massive databases via schema routing. In *Proceedings 28th International Conference on Extending Database Technology*, pages 707–721.

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.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Xiangjin Xie, Guangwei Xu, Lingyan Zhao, and Ruijie Guo. 2025. Opensearch-sql: Enhancing text-to-sql with dynamic few-shot and consistency alignment. *arXiv preprint arXiv:2502.14913*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Tao Yu, Rui Zhang, Heyang Er, Suyi Li, Eric Xue, Bo Pang, Xi Victoria Lin, Yi Chern Tan, Tianze Shi, Zihan Li, and 1 others. 2019. Cosql: A conversational text-to-sql challenge towards cross-domain natural language interfaces to databases. 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)*, pages 1962–1979.

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*.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, and 1 others. 2025. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*.

Yi Zhang and Zachary G. Ives. 2020. Finding related tables in data lakes for interactive data science. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, page 1951–1966.

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

A Key Prompts

The Bi-SR method comprises two phases: offline schema semantic augmentation and online query intent expansion.

A.1 Prompts for Schema Augmentation

These prompts are utilized to transform raw DDL statements into natural language descriptions rich in business semantics, which are then used to construct vector indexes. The specific designs are illustrated in Figure 4.

A.2 Query Intent Expansion

This prompt is employed in the online phase to guide the LLM in analyzing the user's question and inferring potential table or column names. It generates a "pseudo-query" to bridge the semantic gap during retrieval. The specific prompt content is shown in Figure 5.

A.3 Dual-Augmented Training Augmentation

This prompt is employed in training data processing phase to align user's query to expanded targeted database schema. The specific prompt content is shown in Figure 6.

B Dataset Details and Experimental Setting

B.1 Datasets

We conduct experiments on two authoritative Text-to-SQL benchmarks:

- **BIRD** (Li et al., 2024b): A large-scale cross-domain dataset known for its high difficulty, emphasizing "dirty" database values and complex reasoning. It contains 12,751 question-SQL pairs across 95 databases.
- **Spider** (Yu et al., 2018): A widely adopted cross-domain semantic parsing benchmark containing 10,181 questions and 5,693 unique SQL queries across 200 databases.

B.2 Massive Schema Routing Setting

Standard evaluations on these datasets typically assume the target database is known (Oracle Mode), isolating the schema linking task to a small scope. To rigorously evaluate our Schema Routing capability in a realistic setting, we modify the experimental setup following recent works (Wang et al., 2025b; Shi et al., 2025).

Specifically, we discard the ground-truth database identifiers for all test instances. Instead, we construct a unified schema pool comprising all tables from both the training and development sets of the respective datasets. For each user query, the system must identify the correct tables from

Prompt 1: Data Table Schema Augmentation

```
Instruction: Given a database table name and its column names, provide a concise description (5-8
words) of what this table represents or stores.
Input:
Table name: {table_name_original}
Normalized table name: {table_name}
Column names: {columns_list}
Constraint: Please return only a brief description that explains the table's purpose or what kind
of data it stores.
Output: You can think step by step, and output the answer:
```

Prompt 2: Database Schema Augmentation

```
Instruction: Given a list of database table names, provide a concise database description (3-5
words) that summarizes its main function or domain.
Input:
Table names: {table_list}
Constraint: Please return only the description, no other content.
Output: You can think step by step, and output the answer:
```

Figure 4: The prompt templates used for offline schema semantic augmentation. The model generates concise natural language descriptions for tables and the entire database to enhance vector retrieval.

Prompt 3: Query Intent Expansion

```
System Prompt: You are a helpful assistant in database domain.
User Prompt:
Your job is to generate the tables needed to answer the question. You need to describe tables
needed to answer the question, and generate the table description with the table name, with other
potential tables names.
<Example>
{Example}
</Example>
Question: {question}
You can think step by step, and output the answer:
```

Figure 5: The prompt template for online query intent expansion. The model predicts potential table and column names based on the user question to bridge the semantic gap.

this massive pool (597 tables in BIRD and 1020 in Spider) without any prior knowledge of the target database domain. This setting poses a significantly harder challenge, requiring the retriever to distinguish between semantically similar tables from completely different databases.

B.3 Data Synthesis for Small Model Distillation

To bridge the reasoning capability gap between the massive DeepSeek-V3.2 teacher and the compact Qwen3-8B student, we employed a generative knowledge distillation framework rooted in synthetic reasoning trajectories. We treat the original BIRD training set not merely as a collection of static question-SQL pairs, but as seed contexts for synthesizing high-fidelity semantic alignment data.

The core objective of this data generation process is to externalize the implicit routing logic of the teacher model into explicit, learnable natural language tokens. For every training instance in the BIRD corpus, we leverage the superior instruction-following capabilities of DeepSeek-V3.2. By conditioning the teacher model on the user query, the database metadata, and crucially, the ground-truth target tables, we prompt it to generate an "idealized pre-retrieval reasoning trace" (denoted as P_{gold} in our methodology). This synthetic trace explicitly articulates the logical steps required to bridge the user's colloquial intent with the specific schema definitions, effectively deconstructing the "black box" of semantic matching.

We subsequently compile these synthetic traces into a supervised fine-tuning (SFT) dataset, where

Prompt 4: Training Data Augmentation

```
System Prompt: You are a helpful assistant in database domain.
User Prompt:
Your job is to align the question with the tables given. You need to describe tables needed to
answer the question, and align the table description with the table name, with other potential
tables names.
<Example>
{Example}
</Example>
Question: {question}
Tables: {tables}
You can think step by step, and output the answer:
```

Figure 6: The prompt template for training data augmentation. The model aligns user's query to expanded targeted database schema

the input is the raw user question and the target is the teacher-generated reasoning path and potential schema description. By fine-tuning the Qwen3-8B model on this enriched corpus, we force the student model to internalize the teacher's semantic alignment patterns. This process allows the smaller model to emulate the sophisticated "hallucination" capabilities of the larger model during online inference, thereby achieving high-precision schema routing with significantly reduced computational overhead.

B.4 Baseline Implementation Details

We provide detailed descriptions of the baseline methods used in our comparative analysis.

1) Sparse and Dense Retrievers. These methods perform retrieval based on direct similarity without complex schema augmentation.

- **BM25**: The standard sparse retrieval algorithm based on exact keyword matching. It serves as a baseline for lexical overlap performance.
- **SXFMR** (Reimers and Gurevych, 2019): A widely used dense retriever that utilizes Siamese BERT networks to generate sentence embeddings.
- **Qwen3-Embedding** (Zhang et al., 2025): The base model of our retriever. We evaluate it in a zero-shot setting (with our dual-augmented fine-tuning) to verify the contribution of our training strategy.

2) LLM-Augmented Retrievers. These methods leverage PLMs or LLMs to bridge the gap between queries and tables.

- **DTR** (Herzig et al., 2021): A dense retrieval framework specifically designed for open-domain Table QA. It utilizes TAPAS-based encoders and hard negative mining to handle tabular structures.
- **CRUSH4SQL** (Kothyari et al., 2023): A two-stage method that first prompts an LLM to hallucinate a minimal schema (tables and columns) based on the user query, and then uses this hallucinated schema to retrieve the actual database schema.
- **Gen-SQL** (Shi et al., 2025): A retrieval-augmented method that generates a "Pseudo-Schema" (Query-side augmentation) to align with the database. Note that Gen-SQL primarily focuses on query-side enhancement, whereas our Bi-SR performs bidirectional alignment.

3) Graph-based and Agentic Methods. These methods rely on structural graphs or iterative agents to navigate massive schemas.

- **DBCopilot** (Wang et al., 2025b): A pioneering schema routing framework that constructs a compact schema graph to abstract relationships and uses a router to navigate through massive databases.
- **ICRL** (Toteja et al., 2025): An agentic framework that uses reinforcement learning to iteratively refine the retrieval policy and generated queries, optimizing for complex reasoning paths.

Backbone LLM	Method	R@e	R@5	R@15
Qwen3-8B	Gen-SQL	37.54	61.30	87.40
	Bi-SR (Ours)	**47.38**	**73.03**	**89.95**
	Improvement	*+9.84*	*+11.73*	*+2.55*
DeepSeek-V3.2-Exp	Gen-SQL	60.17	82.56	92.58
	Bi-SR (Ours)	**62.54**	**87.16**	**94.25**
	Improvement	*+2.37*	*+4.60*	*+1.67*

Table 6: Impact of Backbone LLM Size on Schema Routing Performance (BIRD Dataset).

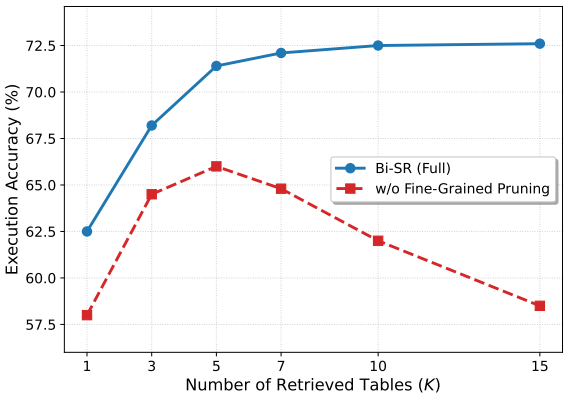

Figure 7: Impact of the number of retrieved tables (K) on execution accuracy on Spider dev dataset.

C Other Analysis

C.1 Empowering Smaller Models

To investigate the framework's adaptability to resource-constrained environments, we replace the backbone LLM with a fine-tuned `Qwen3-8B`. Crucially, this model was fine-tuned on the BIRD training set augmented by our DeepSeek-driven bidirectional semantic data, effectively transferring the reasoning capabilities of the larger teacher model to the smaller student, we share our detail in Appendix B.3. As detailed in Table 6, this strategy yields a remarkable performance leap: Bi-SR boosts the R@e of the 8B model by nearly 10% over the Gen-SQL baseline (47.38% vs. 37.54%). This improvement margin is substantially larger than that observed with the DeepSeek backbone (+2.37%), indicating that while large models can inherently bridge some semantic gaps via strong zero-shot reasoning, smaller models benefit disproportionately from our structured alignment strategy. By explicitly exposing the small model to "ideal reasoning traces" during training, Bi-SR effectively compensates for its weaker inherent hallucination capabilities, enabling it to perform schema routing with a precision that far exceeds standard prompting approaches.

Furthermore, this experiment highlights the significant cost-effectiveness of our framework. Although a performance gap naturally persists between the 8B model and the DeepSeek-V3.2-Exp giant (47.38% vs. 62.54% R@e), Bi-SR enables the lightweight Qwen3-8B to achieve a robust Recall@5 of 73.03%, making it a viable candidate for preliminary filtering in privacy-sensitive or low-latency local deployments. This result confirms that the heavy lifting in our framework—the semantic alignment—is largely offloaded to the offline training stage. Consequently, Bi-SR allows practitioners to deploy efficient, compact models during inference without suffering the catastrophic performance drop typically associated with scaling down reasoning engines in complex schema routing tasks. where massive cloud LLMs are unavailable.

C.2 Impact of Retrieval Count (K) and Pruning

We investigate the impact of the retrieval hyperparameter K (number of candidate tables) on downstream generation performance. As shown in Figure 7, we compare the full Bi-SR framework against a variant w/o Fine-Grained Pruning.

The results reveal a classic trade-off between *Recall* and *Noise*. For the variant without pruning (dashed red line), performance peaks at $K = 5$ but degrades significantly as K increases further (e.g., dropping from 66% at $K = 5$ to 58% at $K = 15$). This "inverted-U" curve indicates that while retrieving more tables improves recall, the excessive irrelevant schema information (Noise) overwhelms the generator's attention mechanism, leading to hallucinations.

In contrast, our full Bi-SR framework (solid blue line) exhibits a robust, non-decreasing performance trend. Even at large retrieval counts ($K = 15$), the accuracy remains stable or improves slightly. This confirms that our Fine-Grained Schema Pruning module effectively acts as a "noise filter," allowing the system to benefit from the high recall of a larger K while shielding the downstream LLM from distraction. This robustness relieves the need for precise K tuning, making Bi-SR highly adaptable to diverse database scales.

C.3 Inference Efficiency Analysis

To assess the feasibility of Bi-SR for real-time applications, we conducted a rigorous latency evaluation against representative baselines. All experiments were performed on a standardized high-

Method	Latency (s)	Throughput (Q/s)	Speedup
DBCopilot	0.82	1.22	1.00×
ICRL	0.79	1.26	1.03×
Gen-SQL	0.10	10.04	8.22×
Bi-SR (Ours)	0.11	9.09	7.45×

Table 7: Inference latency comparison on the Spider dev set. We report the average latency per query (in seconds) and throughput (Queries per second).

performance server equipped with one NVIDIA RTX Pro 6000 Blackwell GPU. To ensure fair comparison, all LLM-augmented methods utilized Qwen-8B as the generative backbone and Qwen-Embedding-4B as the dense retriever. The results are summarized in Table 7.

Structure-heavy approaches like DBCopilot (0.82s) and ICRL (0.79s) exhibit significantly higher latency. This is inherent to their design: they require constructing complex schema graphs and performing iterative traversals or multi-turn reasoning to identify the correct linking path. In contrast, Bi-SR achieves a 7.45× speedup over DBCopilot, demonstrating that our "offline expansion + online dense retrieval" paradigm effectively bypasses the computational bottleneck of graph navigation.

Among the high-efficiency methods, Gen-SQL achieves the lowest latency (0.10s). Our Bi-SR follows closely at 0.11s. The marginal difference (0.01s) is practically negligible in real-world user interactions but reflects the additional computational cost of our richer bidirectional semantic processing. However, this slight trade-off yields substantial returns: as evidenced in Table 2, Bi-SR significantly outperforms Gen-SQL in retrieval recall (e.g., +5.21% R@e on Spider). This confirms that Bi-SR strikes a superior balance, delivering state-of-the-art accuracy while maintaining millisecond-level latency suitable for large-scale deployment.

C.4 Robustness to Metadata Obfuscation

In real-world enterprise databases, schemas frequently employ cryptic abbreviations (e.g., `T_n` instead of `Team_name`). To evaluate robustness against such metadata obfuscation, we construct a "Hard Subset" of 200 tables sampled from the BIRD dataset, where all table and column names are systematically abbreviated.

While baseline methods struggle with opaque identifiers, Bi-SR leverages *value-informed schema expansion*. By incorporating sampled database values (e.g., "Lakers", "Warriors") during the offline phase, Bi-SR successfully infers the underlying semantic context despite the abstract metadata.

Method	Recall@e (%)	Recall@5 (%)
GenSQL	3.2	8.6
Bi-SR (Ours)	**14.2**	**28.1**

Table 8: Retrieval performance on the abstract metadata "Hard Subset" (200 tables from BIRD).

As shown in Table 8, extreme metadata abstraction severely degrades GenSQL to a mere 3.2% in Recall@e. In contrast, Bi-SR achieves 14.2% Recall@e and 28.1% Recall@5. This 4.4× improvement in exact recall empirically demonstrates that our value-grounded bidirectional strategy effectively pierces through cryptic schemas, maintaining robust routing performance where traditional semantic enhancements fail.