Cypher-RI: Reinforcement Learning for Integrating Schema Selection into Cypher Generation

Hanchen Su
Zhejiang University
hcsu@zju.edu.cn

Xuyuan Li
Zhejiang University
3220102970@zju.edu.cn

Yan Zhou
Createlink Technology
zhouyan@chuanglintech.com

Zhuoyi Lu
Zhejiang University
22451217@zju.edu.cn

Ziwei Chai
Zhejiang University
zwchai@zju.edu.cn

Haozheng Wang
Independent Researcher
haozheng.wang@ensea.fr

Chen Zhang
Createlink Technology
zhangchen@chuanglintech.com

Yang Yang *
Zhejiang University
yangya@zju.edu.cn

Abstract

The increasing utilization of graph databases across various fields stems from their capacity to represent intricate interconnections. Nonetheless, exploiting the full capabilities of graph databases continues to be a significant hurdle, largely because of the inherent difficulty in translating natural language into Cypher. Recognizing the critical role of schema selection in database query generation and drawing inspiration from recent progress in reasoning-augmented approaches trained through reinforcement learning to enhance inference capabilities and generalization, we introduce Cypher-RI, a specialized framework for the Text-to-Cypher task. Distinct from conventional approaches, our methodology seamlessly integrates schema selection within the Cypher generation pipeline, conceptualizing it as a critical element in the reasoning process. The schema selection mechanism is guided by textual context, with its outcomes recursively shaping subsequent inference processes. Impressively, our 7B-parameter model, trained through this RL paradigm, demonstrates superior performance compared to baselines, exhibiting a 9.41% accuracy improvement over GPT-4o on CypherBench. These results underscore the effectiveness of our proposed reinforcement learning framework, which integrates schema selection to enhance both the accuracy and reasoning capabilities in Text-to-Cypher tasks.

1 Introduction

Graph databases [28, 37] are commonly utilized to efficiently handle graph data, providing a powerful solution for representing and storing intricate, highly connected information. However, effectively utilizing them remains a formidable challenge, primarily due to the nuanced and complex syntax of graph query language-Cypher [9, 10, 26], which poses obstacles for general users seeking to use graph databases, particularly those unfamiliar with programming paradigms. Therefore, The development of a system for converting natural language into Cypher [11, 13] has gained significant importance.

*Corresponding authors.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

The automatic conversion of natural language questions into Cypher queries remains a persistent challenge in graph database retrieval [7, 14]. The emergence of large language models (LLMs) [21, 38, 40] has led to substantial progress. R^3-NL2GQL [41] combines small and large language models for various tasks including ranking, rewriting, and refinement. This methodology exploits the capabilities of finetuned small models for initial stages while harnessing the advanced generalization and query formulation abilities of big LLMs for the final transformation into Cypher. An alternative strategy proposed by Liang et al. [22] employs ChatGPT [2] to generate natural language-Cypher paired datasets through self-instruction, based on given graph databases. These datasets are subsequently used to fine-tune LLMs, facilitating alignment between LLMs and the graph databases. Nevertheless, this approach necessitates dataset construction and model retraining for each new graph database, incurring significant time and financial costs. Ozsoy et al. [30] curated and structured several publicly available datasets, facilitating effective fine-tuning. The models fine-tuned on this dataset demonstrated significant performance improvements.

Existing approaches predominantly depend on supervised fine-tuning [23] and few-shot prompting [15] to guide models in Cypher query generation. Nevertheless, when faced with intricate database schemas or ambiguous linguistic contexts that necessitate advanced reasoning, such models often fail to produce Cypher queries accurately reflecting users' intentions, due to their reliance on static generation patterns and previously observed data. Consequently, current systems exhibit limited robustness [5] in adapting to new domains and generalizing across diverse database environments. Moreover, the opaque reasoning mechanisms underlying Cypher generation restrict the applicability of these models in critical sectors like law and finance, where transparency and accountability are paramount.

Recently, reinforcement learning (RL) [16, 32, 33] has emerged as a promising paradigm for enhancing the reasoning capabilities of LLMs. Unlike traditional fine-tuning, RL enables adaptive modification of decision policies through continuous interaction with dynamic environments, resulting in improved performance on complex inference tasks. Empirical studies have validated the efficacy of RL-based frameworks in strengthening reasoning proficiency and generalization in areas such as information retrieval [4, 20], mathematical problem solving [17, 39] and SQL generation [27, 31]. Furthermore, innovations in expansive reasoning models, exemplified by OpenAI o1 [19] and DeepSeek-R1 [12], demonstrate that allocating additional computational resources during inference to construct deliberate reasoning paths substantially elevates output quality.

Providing an LLM with concise structural metadata about a graph database helps it convert natural-language questions into precise query statements. Feeding complete schema dumps to the model, however, often injects irrelevant detail, provokes spurious outputs, and increases computational expenses. Schema selection [3, 24], which focuses on identifying and providing only the schema components pertinent to a given query context, serves as a key mechanism for enhancing the precision and overall performance of natural-language–to–database query generation models. By steering the model's attention toward relevant schema elements, it alleviates cognitive burden and significantly narrows the exploration space for LLMs. Motivated by advancements in training reasoning models through RL and recognizing the advantages of schema selection, we introduce Cypher-RI, a reinforcement learning-based framework that integrates schema selection into Cypher generation. This method promotes the articulation of explicit intermediate reasoning trajectories, thereby substantially improving the reasoning quality and final query accuracy of LLMs.

Unlike common reasoning process solely involves `<think>` and `<answer>`, our framework incorporates additional components: specifically, the identification of needed entities and relations (annotated within `<json>` tags) alongside the results of schema selection (enclosed within `<schema>` tags). We conceptualize schema selection as an integral element of the overall reasoning trajectory, enabling dynamic interactions between schema selection and textual deliberation. Notably, no manually annotated reasoning traces are provided to guide the language models. Instead, we employ reinforcement learning techniques to motivate LLMs to autonomously develop reasoning chains that integrate schema selection behavior.

We train Cypher-RI from scratch using Qwen2.5-Coder-7B and perform comprehensive evaluations on multiple Text-to-Cypher benchmarks. Notably, we validate the generalization capability of our framework by evaluating the trained models on graph databases distinct from those used during training. On CypherBench [8], our models achieve substantial absolute improvements ranging from

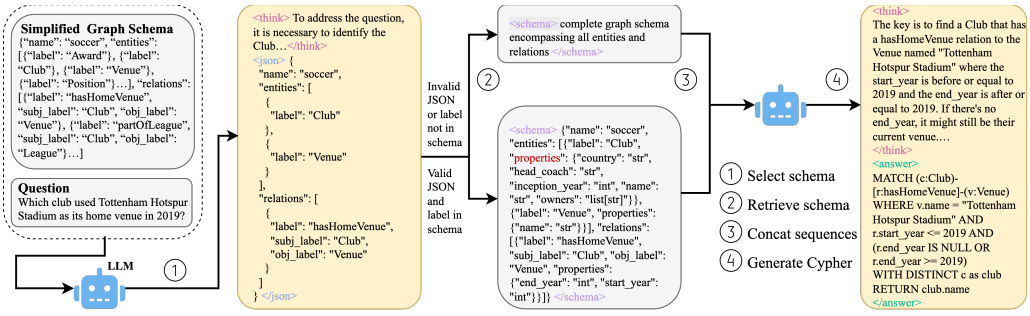

Figure 1: An overview of the Cypher-RI training process. During the rollout phase, the LLM integrates schema selection into Cypher generation. The two LLMs shown in the figure correspond to the same model. Only the text highlighted with a yellow background participates in backpropagation.

13.12% to 69.59% over the base model, and outperform the strong GPT-4o baseline by up to 9.41%. Our principal contributions are as follows:

- This study offers new insights into leveraging reinforcement learning to enhance the Cypher generation capabilities of LLMs, marking the first RL-driven framework designed to autonomously optimize the reasoning process for the Text-to-Cypher task.
- We introduce Cypher-RI, a novel approach that integrates schema selection directly into the Cypher generation during training. Cypher-RI enables LLMs to learn reasoning strategies from scratch without relying on any supervised annotations for intermediate reasoning steps.
- Extensive empirical evaluations validate the effectiveness of our method consistently achieving substantial improvements over baseline approaches, and even outperforming the strong GPT-4o model while being much more cost effective.

2 Method

Schema selection is fundamental to the accuracy and effectiveness of Text-to-Cypher systems. Providing the entire schema as input to an LLM can cause information overload, hindering the model's ability to distinguish relevant information from irrelevant details. Schema selection mitigates this issue by directing the model's focus toward pertinent schema components. By filtering out irrelevant data, schema selection enables more accurate mapping of natural language queries to corresponding nodes and edges within the graph schema, thereby improving the accuracy of the generated Cypher queries. For questions requiring traversal across multiple nodes or involving complex aggregations, correctly selecting the relevant nodes and edges is crucial. Schema selection identifies the necessary components for constructing appropriate queries, effectively reducing the search space for LLMs. In essence, schema selection serves as the semantic grounding step, interpreting the user's intent within the specific context of the target database.

Considering the benefits of schema selection and drawing inspiration from the success of the o-series models developed by OpenAI and DeepSeek-R1 in learning to reason, we integrate schema selection into the reasoning process, as illustrated in Figure 1. We train LLMs from scratch using reinforcement learning without any labeled data on reasoning chains, enabling LLMs to learn schema selection and Cypher generation in a unified manner.

The training methodology for Cypher-RI is presented first with an in-depth exploration of GRPO and the implementation of rollout strategies that incorporate schema selection during reinforcement learning. Subsequently, we describe the design of a training template that steers LLMs to produce rollouts in a predefined format. Lastly, we outline the reward design, which plays a pivotal role in guiding the optimization process within the reinforcement learning framework.

2.1 Reinforcement Learning

When generating database query language, accurate schema selection plays a pivotal role in enhancing precision. However, obtaining labeled reasoning data that incorporates schema selection for supervised fine-tuning of LLMs remains a significant challenge, especially when aiming to replicate reasoning patterns with schema selection. A promising solution lies in reinforcement learning, which has demonstrated remarkable efficacy in training LLMs to perform reasoning tasks, even in the absence of labeled data. The core idea of reinforcement learning in our method is to sample diverse reasoning processes through schema selection and Cypher generation chains, and to optimize the LLMs to maximize the likelihood of producing rollouts that achieve higher rewards.

Unlike traditional reasoning models, which typically involve only the <think> and <answer> phases, the Cypher-RI rollout incorporates an additional schema selection step. In this process, the question and a simplified version of the graph schema containing only the names of entities and relations are provided as input to the LLM. The model first identifies the schema relevant to the question and generates a JSON-formatted string to retrieve the complete version of the selected schema, which includes the properties of entities and relations. These instructions are detailed in the training templates, which will be explained in subsequent sections. The rollout procedure follows a sequence of reasoning steps: thinking, selecting the schema, retrieving the complete schema, further reasoning, and generating the answer, as depicted in Figure 1. More specifically, the rollout process terminates upon encountering the `</json>` tag. If the string between the `<json>` and `</json>` tags can be successfully parsed and the parsed labels exist in the schema, the label will be used as a key to retrieve the corresponding complete schema. If the string fails to conform to the JSON format or the label is not found in the schema, the complete schema encompassing all entities and relations will be returned. This complete schema is enclosed within `<schema>` and `</schema>` tags and appended to the rollout sequence. The updated rollout, now containing the complete selected schema, is then used as input for the next generation step until the end-of-sentence tag is reached.

To enhance the stability of policy optimization and eliminate the necessity for an auxiliary value function approximation, Deepseek propose Group Relative Policy Optimization(GRPO). Unlike Proximal Policy Optimization (PPO), GRPO utilizes the mean reward derived from a set of sampled outputs as a baseline, instead of depending on a learned value function. Specifically, given an input query x, GRPO samples a batch of responses $\{y_1, y_2, \ldots, y_G\}$ from the policy model π_θ. The policy model is subsequently optimized by maximizing the following objective function:

$$\mathcal{J}(\theta) = \mathbb{E}_{x\sim\mathcal{D},\{y_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}(\cdot|x)} \tag{1}$$

$$\frac{1}{G}\sum_{i=1}^{G}\left[\frac{1}{\sum_{t=1}^{T}M_{i,t}}\sum_{t=1}^{T}\left(\min\left(\frac{\pi_\theta(y_i|x)}{\pi_{\theta_{\text{old}}}(y_i|x)}A_i, \text{clip}\left(\frac{\pi_\theta(y_i|x)}{\pi_{\theta_{\text{old}}}(y_i|x)}, 1-\epsilon, 1+\epsilon\right)A_i\right) - \beta\mathbb{D}_{\text{KL}}\left(\pi_\theta||\pi_{\theta_{\text{ref}}}\right)\right)M_{i,t}\right], \tag{2}$$

$$A_i = \frac{r_i - \text{mean}(\{r_j\}_{j=1}^{G})}{\text{std}(\{r_j\}_{j=1}^{G})} \tag{3}$$

where π_{ref} is the reference model, π_{old} is the old policy model, ϵ and β are hyperparameters, while A_i represents the advantage, calculated from the relative rewards r_i of outputs within each group. This method eliminates the need for additional complexity when computing A_i. Rather than introducing the Kullback-Leibler (KL) divergence as a penalty in the reward function, GRPO regularizes the model by directly incorporating the KL divergence between the learned policy and the reference policy into the loss function.

Additionally, we multiply $M_{i,t}$ by each token when computing the GRPO loss. The value of $M_{i,t}$ is either 0 or 1. During the reinforcement learning process of LLMs, token-level losses are calculated across the entire rollout sequence. In Cypher-RI, this sequence comprises both tokens generated by the LLM and tokens representing the full schema retrieved from an external dictionary. While optimizing the LLM-generated tokens enhances the model's ability to perform reasoning, selection, and generation, applying the same optimization to the retrieved tokens could inadvertently introduce undesired learning behaviors. To mitigate this issue, we propose the use of loss masking for the retrieved tokens, ensuring that the policy gradient objective is computed solely based on the LLM-generated tokens and excluding the retrieved tokens from the optimization. This method contributes to the stabilization of the training process while consolidating schema selection and Cypher generation into a cohesive framework.

Table 1: Training template for Cypher-RI. During training and inference, the placeholders question and schema will be replaced with specific instances.

A conversation between a User and an Assistant. The user asks the Assistant to translate the question to Cypher query based on a simplified version of the schema of a graph database. The Assistant first identifies the necessary schema components required to answer the User's question within <think> </think> tags. Next, the Assistant outputs the needed schema in JSON format within <json> </json> tags. If the selected schema is valid, the Assistant is provided with the complete version of the selected schema enclosed within <schema> tags, or it is provided with the complete version of the full schema. After that, the Assistant explains the reasoning process for generating the Cypher query within <think> </think> tags. Finally, the Assistant provides the Cypher query enclosed within <answer> </answer> tags. For example, <think> This is the reasoning process. </think> <json> {"name": "", "entities": [], "relations": []} </json> <schema> This is the complete version of the schema. </schema> <think> This is the reasoning process. </think> <answer> This is the Cypher query </answer>. Graph schema: {schema} User: {question} Assistant:

2.2 Training Template

The training of Cypher-RI begins with the creation of a straightforward template designed to guide the initial LLM in adhering to predefined instructions. As illustrated in Table 1, this framework organizes the output into five iterative components: an initial reasoning phase identifying the needed schema, followed by a JSON-formatted string representing the selected schema, the complete version of schema, a subsequent reasoning phase focused on Cypher query generation, and ultimately, the final response. By guiding LLMs to comprehend the rollout format, we incorporate schema selection into the training process, enabling the model to jointly learn schema selection and Cypher generation in a unified framework.

2.3 Reward design

In the reinforcement learning process of Cypher-RI, no supervised reasoning data is utilized, and optimization of LLMs is guided solely through a simplified reward mechanism applied to rollouts. Empirically, a rule-based reward function is sufficient to effectively elicit the reasoning capabilities that integrate schema selection and Cypher generation within LLMs. The reward function is composed of three key parts: format reward, schema selection reward, and Cypher execution reward.

To compute the format reward, we first specify the correct format as follows:

- The model's reasoning process and final answer should be enclosed within the <think>...</think> and <answer>...</answer> tags.
- The <json> </json> and <answer> </answer> tags must appear once, while the <think> </think> tags must appear twice.
- The order of the tags must conform to <think> </think> <json> </json> <think> </think> <answer> </answer>.

According to the aforementioned format requirements, the format reward is defined as follows:

$$r_{format} = \begin{cases} 1.0, & \text{if the format is correct} \\ -1.0, & \text{if the format is incorrect} \end{cases} \tag{4}$$

For the schema selection reward, we first parse the string generated within the <json> tags and then compare the parsed selected schema against the gold answer.

$$r_{selection} = \begin{cases} 2.0, & \text{if the selected schema matches the gold answer} \\ -1.5, & \text{if the selected schema does not match the gold answer} \\ -2.0, & \text{if the selected schema is unparseable} \end{cases} \tag{5}$$

For the Cypher execution reward, we extract the generated Cypher query enclosed within the <answer> tags, then execute both the extracted Cypher and the gold answer on the graph database.

$$r_{execution} = \begin{cases} 2.0, & \text{if the generated Cypher executes consistently with the gold answer} \\ -1.5, & \text{if the generated Cypher executes inconsistently with the gold answer} \\ -2.0, & \text{if the generated Cypher fails to execute} \end{cases} \quad (6)$$

The final reward is the sum of the format reward, schema selection reward, and Cypher execution reward.

$$r = r_{format} + r_{selection} + r_{execution} \quad (7)$$

3 Experiment

3.1 Training Data

GRPO algorithm encounters an issue related to vanishing gradients when certain prompts achieve perfect accuracy. If all generated outputs for a given question are correct and receive a reward of 1, the computed advantage for that group becomes zero. A zero advantage provides no meaningful gradient signal for policy updates, which severely limits sample efficiency. The situation in which all outputs are incorrect remains the same. To this end, we perform data selection by generating eight Cypher queries per prompt using our method.

Specifically, we select training instances from the CypherBench [8] training set . We employ the Qwen-2.5-Coder-7B model to first perform schema selection, then generate the corresponding Cypher queries, and subsequently execute them within the database. Prompts with a mean selection accuracy of 1 and a mean execution accuracy of 1, as well as those with a mean selection accuracy of 0 and a mean execution accuracy of 0, are subsequently filtered out. Finally, we obtain 8,295 samples from four distinct graph databases: Art, Terrorist Attack, Soccer, and Biology.

3.2 Evaluation Datasets and Metrics

We use two datasets to evaluate the conversion of natural language into Cypher queries: CypherBench, Neo4j-Text2Cypher. Specifically, CypherBench is constructed across seven different property graphs, including Fictional Character, Company, Flight Accident, Geography, Movie, NBA and Politics. Neo4j-Text2Cypher is a synthetic dataset developed by Neo4j, comprising 15 graph databases.

To ensure a fair comparison, we adopt the evaluation standards established in previous benchmarks. Specifically, we assess model performance using Execution Accuracy (EX), Provenance Subgraph Jaccard Similarity (PSJS), executable percentage (Exec.) and Google-BLEU. Execution Accuracy evaluates whether the database results executed by the predicted query align with those produced by the ground-truth query. The provenance subgraph is defined as the portion of the graph retrieved by executing the MATCH clause in combination with RETURN *. The PSJS metric quantifies query similarity by computing the Jaccard similarity between provenance subgraphs of generated and ground-truth Cypher queries. Exec. assesses whether the generated Cypher queries can be executed in graph databases without compiler errors. Google-BLEU metric quantifies textual correspondence by analyzing n-gram matches between the generated Cypher and the gold answer.

3.3 Baselines

To evaluate the effectiveness of Cypher-RI, we compare it with a range of baseline approaches: (1) General Purpose Methods: including Qwen-2.5-Coder-7B[18], Llama3.1-8B[6], Gemma-2-9B[35], GPT-4o-mini, Gemini-1.5-Pro[34], and GPT-4o[1]; and (2) Text-to-Cypher Specific Methods: R^3-NLGQL[41], Text2Cypher-Gemma-3-27B[30, 36]. These baselines collectively represent both domain-agnostic and domain-specialized strategies, enabling a thorough and multifaceted evaluation of the proposed method.

3.4 Implementation Details

In our Cypher-RI framework, we adopt Qwen-2.5-Coder-7B as the base model for training. The dataset consists of 8,295 instances sourced from the CypherBench training split. During the training

Table 2: Main results of different methods on CypherBench and Neo4j-Text2Cypher. The best performance is set in bold.

Methods	CypherBench			Neo4j-Text2Cypher	
	EX (%)	PSJS (%)	Exec. (%)	EX (%)	Google-BLEU (%)
Qwen2.5-Coder-7B	13.12	22.65	62.95	12.06	44.37
Gemma-2-9B	18.61	30.67	68.57	14.49	51.04
Llama3.1-8B	18.82	30.98	90.67	11.86	41.30
R^3-NLGQL	23.94	38.12	64.83	19.27	47.91
GPT-4o-mini	31.43	45.91	87.39	25.82	59.62
Text2Cypher-Gemma-3-27B	38.71	55.53	92.97	29.84	50.10
Gemini-1.5-Pro	39.95	57.70	86.03	23.07	58.45
GPT-4o	60.18	**76.87**	94.90	**31.73**	**62.93**
Cypher-RI(Ours)	**69.59**	75.21	**99.28**	30.61	60.13

process, each instance is sampled through 8 rollouts with the sampling temperature maintained at 1.0. We employ a training batch size of 128 and set the rollout batch size to 128 as well. The learning rate is fixed at 1×10^{-6}. Further training details are provided in Appendix B. In R^3-NLGQL, Qwen-2.5-Coder-7B is employed as the foundation model across all components.

3.5 Main Results

Table 2 reports the comprehensive evaluation results of Cypher-RI and several strong baseline models across two benchmarks. Cypher-RI demonstrates state-of-the-art performance in crucial aspects of Cypher query generation, achieving the highest Execution Accuracy at 69.59%. This represents a significant margin over other models, notably outperforming the strong GPT-4o model, which scored 60.18% by 9.41 percentage points. Furthermore, Cypher-RI also sets a new standard for generating syntactically valid queries, attaining an exceptional Executable Percentage of 99.28%. This near-perfect rate in producing executable queries is the highest among all evaluated models, surpassing GPT-4o and indicating the high reliability and robustness of our approach. In terms of PSJS, GPT-4o records the highest score at 76.87%, with Cypher-RI achieving a very competitive second place at 75.21%. While GPT-4o shows a slight edge in aligning more closely with the exact structural representation of the ground truth provenance subgraph, Cypher-RI's leading EX score suggests that it effectively identifies and retrieves the correct data, potentially through equally valid but structurally slightly different query patterns. This also implies that while PSJS is an important measure of structural understanding, Cypher-RI excels in translating this understanding into executable and correct queries.

When compared to its base model, Qwen2.5-Coder-7B, which scored 13.12% EX, 22.65% PSJS, and 62.95% Exec., Cypher-RI exhibits a remarkable improvement across all evaluated metrics. Specifically, Cypher-RI boosts the Execution Accuracy by an absolute 56.47 percentage points, the Provenance Subgraph Jaccard Similarity by 52.56 percentage points, and the Executable Percentage by 36.33 percentage points over its foundation. This substantial leap in performance clearly demonstrates the profound efficacy of our training methodology in refining and specializing the base model for complex Cypher query generation.

In terms of PSJS, Cypher-RI achieving a very competitive second place at 75.21%. While GPT-4o shows a slight edge in aligning more closely with the exact structural representation of the ground truth provenance subgraph, Cypher-RI's leading EX score suggests that it generates valid but structurally slightly different query. This observation is consistent with our training reward, which is defined in terms of execution accuracy.

In the Neo4j-Text2Cypher evaluation, Cypher-RI also exhibits highly competitive results. It achieves an Execution Accuracy of 30.61%, closely matching GPT-4o's 31.73%, and outperforms all other baselines. Regarding Google-BLEU, which measures n-gram level similarity between generated queries and references, Cypher-RI achieves 60.13%, again ranking just behind GPT-4o.

A particularly noteworthy finding from our experiments is the efficiency and effectiveness of Cypher-RI relative to model scale. Despite being developed upon a 7B parameter model, Cypher-RI consistently outperforms significantly larger and more general-purpose models in the specialized domain of Cypher generation. For instance, it surpasses GPT-4o in both execution accuracy and the ability

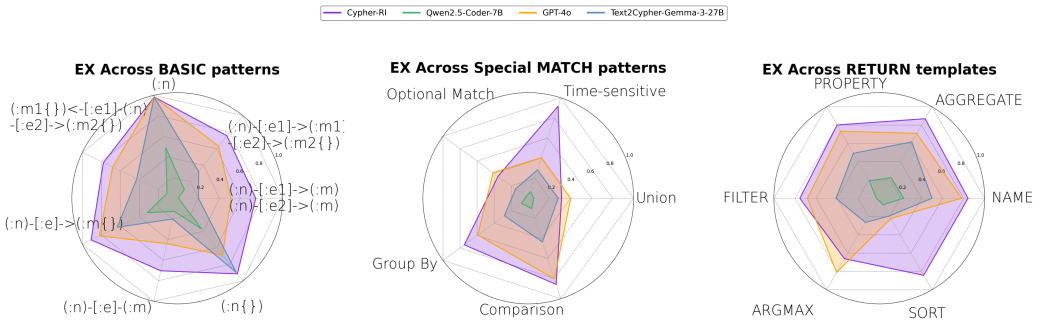

Figure 2: Performance evaluation of four models across distinct BASIC patterns, MATCH Patterns and RETURN templates.

to generate executable queries in CypherBench. Similarly, Text2Cypher-Gemma-3-27B, a 27B parameter model specifically finetuned for Text-to-Cypher tasks, also lags behind Cypher-RI with an EX of 38.71%. This highlights Cypher-RI's advanced capabilities and its potential for providing highly accurate Cypher generation without necessitating the substantial computational resources typically associated with much larger models.

3.6 Analysis

In this section, we provide a detailed analysis of the performance breakdown across multiple dimensions. As illustrated in Figure 2, the left chart presents the execution accuracy across different basic Cypher patterns. Here, "(:n)" denotes a node, "(:n{})" indicates a node with specified properties, and "-[:e]->" represents an edge. Notably, the performance of Qwen-2.5-Coder-7B is significantly lower than that of the other models, underscoring a substantial capability gap. In contrast, our RL-trained model achieves highly competitive, and frequently superior, performance compared to the strong baseline of GPT-4o. While GPT-4o exhibits robust general capabilities, Cypher-RI demonstrates a distinct advantage on several more complex Cypher structures, including two-hop paths and bidirectional patterns. Cypher-RI not only achieves high peak performance but also demonstrates greater consistency across the range of patterns. Its performance floor is significantly higher than the other models, especially the base version. This robustness is critical for practical applications where a wide variety of query types might be encountered. The high accuracy on complex patterns suggests an ability to handle disjunctive or conjunctive relational conditions effectively.

The mid chart in Figure 2 edisplays execution accuracy across the special patterns. Compared with GPT-4o, Cypher-RI showcased competitive and, in several categories, superior performance. Notably, our model achieved higher execution accuracy in "Time-sensitive", "Comparison" and "Group By" queries. While GPT-4o demonstrated a stronger performance in "Union" and "Optional Match"patterns. The ability of our model to handle intricate Cypher patterns such as time-sensitive constraints and optional graph traversals with high accuracy is a testament to the benefits of our training approach.

Delving into specific return templates, as illustrated in the right chart of Figure 2, Cypher-RI demonstrates exceptional proficiency in handling all clauses, achieving the highest execution accuracy in these categories. This suggests that the reinforcement learning process has effectively equipped the model to understand and generate complex query structures involving specific entity returns, ordering of results, and aggregation functions. In the "SORT" template, which typically requires more sophisticated semantic understanding to order values based on specific criteria, all other models achieve an execution accuracy of less than 25%. In contrast, Cypher-RI attains an accuracy of 84.21%.

Integrating schema selection into Cypher generation within a reinforcement learning framework and applying it to the training of LLMs can achieve state-of-the-art performance in the Text-to-Cypher task, providing a more efficient and targeted alternative to relying solely on larger, general-purpose models. The experimental results compellingly highlight the effectiveness of our method in training large language models for Cypher generation, enhancing their ability to capture semantic subtleties

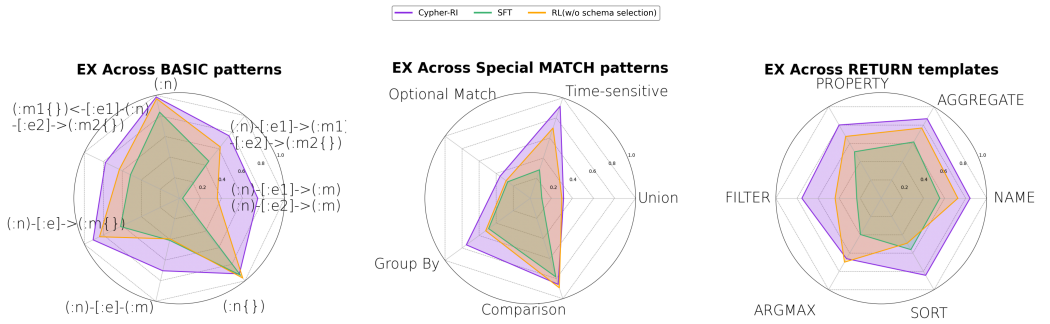

Figure 3: Performance evaluation of ablation models across distinct BASIC patterns, MATCH Patterns and RETURN templates.

and syntactic rules. Notably, the improvements are particularly pronounced in cases involving multiple relationships, time-related questions, and sorting requirements. Cypher-RI, our RL-trained model, significantly outperforms its base model and shows competitive, often superior, performance compared to GPT-4o across a variety of fundamental Cypher patterns. These findings demonstrate that trained open-source models can achieve highly accurate and reliable natural language-to-Cypher translation, validating the advantages of our specialized RL training method.

3.7 Ablation Study

In this section, we aim to assess the enhancement effects of reinforcement learning and the integration of schema selection into training through comparative experiments. To demonstrate the effectiveness of reinforcement learning, we performed supervised fine-tuning on the model. Specifically, we used the same model as in the reinforcement learning setup to collect rollouts on the training sets of CypherBench. For each instance in the training set, we prompted the LLM to first select a schema and then generate the corresponding Cypher query. The generated Cypher was executed against the graph database, and if the execution result matched the gold answer, we added the trajectory to the training data. We sampled trajectories until the model generated the correct answer, with a maximum of eight attempts per instance. To evaluate the effectiveness of integrating schema selection into the training process, we also trained the model using reinforcement learning without schema selection, similar to the approach in DeepSeek R1.

Model	EX (%)	PSJS (%)	Exec. (%)
SFT	44.34	49.80	94.34
RL(w/o schema selection)	58.35	66.55	96.12
Cypher-RI	**69.59**	**75.21**	**99.28**

Table 3: Ablation Study Experiment on the Cypher-Bench Test Set.

The results of these experiments are presented in Table 3. We can see that Cypher-RI outperforms SFT and RL (without schema selection) on the test sets, demonstrating superior Cypher generation capability and better generalization across graph databases. Figure 3 clearly illustrates the types of situations where our method provides improvements. Compared to other methods, Cypher-RI shows significant gains on complex basic patterns, “Group By”and “Time-sensitive” special patterns, as well as “FILTER” and “SORT” templates.

We also evaluated the model’s performance on the training data to better understand its learning behavior. As shown in Table 4, the SFT method achieved the highest EX score on the training data but the lowest EX score on the test data. This phenomenon demonstrates that RL excels at learning generalizable knowledge, whereas SFT tends to simply memorize the training data.

Model	EX (%)	PSJS (%)	Exec. (%)
SFT	**85.66**	85.63	97.15
RL(w/o schema selection)	80.89	81.20	99.39
Cypher-RI	84.59	**86.39**	**99.78**

Table 4: Ablation Study Experiment on the Training Data.

4 Related Work

Text-to-Cypher Methods. Recent progress in large language models has markedly advanced methods for translating natural language into Cypher queries. Hornsteiner et al. [15] proposed a framework that notably improves user–graph database interaction and establishes a scaffold for integrating multiple database systems with large language models. The R^3-NL2GQL [41] approach combines big and small language models across different stages, exploiting smaller models' strengths for preliminary ranking and textual normalization, while assigning final query synthesis and broad generalization to larger models to produce high-quality GQL outputs. Liang et al. [22] synthesize paired NL–GQL examples grounded in the target graph schema firstly. Next, these synthetic pairs serve to fine-tune LLMs so that their outputs better conform to the database's schema and querying conventions. Text2Cypher [30] demonstrate the value of aggregating, sanitizing, and structuring several public resources into a consolidated training corpus; models trained on this cleaned collection exhibited notable gains in downstream query-generation performance. NAT-NL2GQL [23] proposed three agents to generate Cypher: a Preprocessor for context processing, a Generator for GQL creation, and a Refiner that optimizes outputs using execution feedback.

Reinforcement Learning for Large Language Models. Reinforcement learning (RL) has proven to be a valuable approach for improving the reasoning abilities of large language models. Ouyang et al. [29] were among the first to apply RL to LLM fine-tuning by using reinforcement learning from human preferences. Their pipeline trains a reward model from human preference annotations and then leverages that learned reward signal to perform policy optimization on the base language model — a process commonly implemented with PPO. DeepSeek proposed GRPO, which eliminates the need for a critic model by estimating baselines from group scores, exhibiting strong performance in math field. Beyond these examples, reinforcement learning has been increasingly applied to a variety of generation-oriented tasks, each leveraging RL to enhance model reasoning. For instance, in SQL generation, systems such as SQL-R1 [27] and Reasoning-SQL [31] employ reward functions derived from query execution accuracy to guide policy updates, enabling the models to generate correct SQL. In the logical reasoning domain, Logic-RL [39] trains language models by using the exact match of the final answer as the reward signal, enabling the model to iteratively refine its logical reasoning when solving puzzles. For code generation, Code-R1 [25] integrates RL objectives based on functional correctness, optimizing the model toward producing executable and efficient programs. Search-R1 [20] combines reinforcement learning with search-enhanced training, where the model is rewarded for selecting informative retrieval paths and reasoning chains, effectively aligning its search strategy with the goal of more accurate multi-hop reasoning.

5 Limitations

This study focuses specifically on the generation of Cypher queries. While Cypher is the most widely used graph query language, we acknowledge that other graph query languages such as Gremlin, PGQL and G-CORE are also utilized in the graph database landscape. Future research could extend our reinforcement learning methodology to investigate its applicability and performance in generating queries for these alternative languages, thereby broadening the scope of LLM-driven text-to-query translation for graph databases.

6 Conclusion

This study presents Cypher-RI, an innovative framework that integrates schema selection into Cypher query generation through reinforcement learning, without relying on any annotated reasoning step supervision. In our methodology, schema selection is embedded as a critical element within the reasoning process: the model first performs text-driven schema selection, and the selected schema is then concatenated into the rollout sequence to guide the subsequent Cypher generation. Extensive empirical evaluations across multiple Text-to-Cypher benchmarks demonstrate that Cypher-RI achieves substantial gains compared to existing baseline models. Moreover, the findings suggest strong applicability of our framework to practical, real-world environments. Overall, this research underscores the promise of reinforcement learning in jointly modeling schema selection and Cypher query generation, offering a compelling direction toward building more capable, trustworthy LLM-driven Text-to-Cypher systems.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Tom B Brown. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

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

[4] Mingyang Chen, Tianpeng Li, Haoze Sun, Yijie Zhou, Chenzheng Zhu, Fan Yang, Zenan Zhou, Weipeng Chen, Haofen Wang, Jeff Z Pan, et al. Learning to reason with search for llms via reinforcement learning. *arXiv preprint arXiv:2503.19470*, 2025.

[5] Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V Le, Sergey Levine, and Yi Ma. Sft memorizes, rl generalizes: A comparative study of foundation model post-training. *arXiv preprint arXiv:2501.17161*, 2025.

[6] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[7] Guandong Feng, Guoliang Zhu, Shengze Shi, Yue Sun, Zhongyi Fan, Sulin Gao, and Jun Hu. Robust nl-to-cypher translation for kbqa: Harnessing large language model with chain of prompts. In *China Conference on Knowledge Graph and Semantic Computing*, pages 317–326. Springer, 2023.

[8] Yanlin Feng, Simone Papicchio, and Sajjadur Rahman. Cypherbench: Towards precise retrieval over full-scale modern knowledge graphs in the llm era. *arXiv preprint arXiv:2412.18702*, 2024.

[9] Nadime Francis, Alastair Green, Paolo Guagliardo, Leonid Libkin, Tobias Lindaaker, Victor Marsault, Stefan Plantikow, Mats Rydberg, Petra Selmer, and Andrés Taylor. Cypher: An evolving query language for property graphs. In *Proceedings of the 2018 international conference on management of data*, pages 1433–1445, 2018.

[10] Balaji Ganesan, Sambit Ghosh, Nitin Gupta, Manish Kesarwani, Sameep Mehta, and Renuka Sindhgatta. Llm-powered graphql generator for data retrieval. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pages 8657–8660, 2024.

[11] Alastair Green, Paolo Guagliardo, Leonid Libkin, Tobias Lindaaker, Victor Marsault, Stefan Plantikow, Martin Schuster, Petra Selmer, and Hannes Voigt. Updating graph databases with cypher. *Proceedings of the VLDB Endowment*, 12(12):2242–2254, 2019.

[12] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

[13] Florian Holzschuher and René Peinl. Performance of graph query languages: comparison of cypher, gremlin and native access in neo4j. In *Proceedings of the Joint EDBT/ICDT 2013 Workshops*, pages 195–204, 2013.

[14] Markus Hornsteiner, Michael Kreussel, Christoph Steindl, Fabian Ebner, Philip Empl, and Stefan Schönig. Real-time text-to-cypher query generation with large language models for graph databases. *Future Internet*, 16(12):438, 2024.

[15] Markus Hornsteiner, Michael Kreussel, Christoph Steindl, Fabian Ebner, Philip Empl, and Stefan Schönig. Real-time text-to-cypher query generation with large language models for graph databases. *Future Internet*, 16(12):438, 2024.

[16] Jian Hu. Reinforce++: A simple and efficient approach for aligning large language models. *arXiv preprint arXiv:2501.03262*, 2025.

[17] Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. *arXiv preprint arXiv:2503.24290*, 2025.

[18] Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, et al. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*, 2024.

[19] Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

[20] Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*, 2025.

[21] Jean Kaddour, Joshua Harris, Maximilian Mozes, Herbie Bradley, Roberta Raileanu, and Robert McHardy. Challenges and applications of large language models. *arXiv preprint arXiv:2307.10169*, 2023.

[22] Yuanyuan Liang, Keren Tan, Tingyu Xie, Wenbiao Tao, Siyuan Wang, Yunshi Lan, and Weining Qian. Aligning large language models to a domain-specific graph database. *arXiv preprint arXiv:2402.16567*, 2024.

[23] Yuanyuan Liang, Tingyu Xie, Gan Peng, Zihao Huang, Yunshi Lan, and Weining Qian. Natnl2gql: A novel multi-agent framework for translating natural language to graph query language. *arXiv preprint arXiv:2412.10434*, 2024.

[24] Geling Liu, Yunzhi Tan, Ruichao Zhong, Yuanzhen Xie, Lingchen Zhao, Qian Wang, Bo Hu, and Zang Li. Solid-sql: Enhanced schema-linking based in-context learning for robust text-to-sql. *arXiv preprint arXiv:2412.12522*, 2024.

[25] Jiawei Liu and Lingming Zhang. Code-r1: Reproducing r1 for code with reliable rewards. 2025.

[26] Yang Liu, Xin Wang, Jiake Ge, Hui Wang, Dawei Xu, and Yongzhe Jia. Text to graph query using filter condition attributes. *Proceedings of the VLDB Endowment. ISSN*, 2150:8097.

[27] Peixian Ma, Xialie Zhuang, Chengjin Xu, Xuhui Jiang, Ran Chen, and Jian Guo. Sql-r1: Training natural language to sql reasoning model by reinforcement learning. *arXiv preprint arXiv:2504.08600*, 2025.

[28] Justin J Miller. Graph database applications and concepts with neo4j. In *Proceedings of the southern association for information systems conference, Atlanta, GA, USA*, volume 2324, pages 141–147, 2013.

[29] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

[30] Makbule Gulcin Ozsoy, Leila Messallem, Jon Besga, and Gianandrea Minneci. Text2cypher: Bridging natural language and graph databases. *arXiv preprint arXiv:2412.10064*, 2024.

[31] Mohammadreza Pourreza, Shayan Talaei, Ruoxi Sun, Xingchen Wan, Hailong Li, Azalia Mirhoseini, Amin Saberi, Sercan Arik, et al. Reasoning-sql: Reinforcement learning with sql tailored partial rewards for reasoning-enhanced text-to-sql. *arXiv preprint arXiv:2503.23157*, 2025.

[32] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[33] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

[34] Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.

[35] Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

[36] Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

[37] Bing Tong, Yan Zhou, Chen Zhang, Jianheng Tang, Jing Tang, Leihong Yang, Qiye Li, Manwu Lin, Zhongxin Bao, Jia Li, et al. Galaxybase: A high performance native distributed graph database for htap.

[38] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.

[39] Tian Xie, Zitian Gao, Qingnan Ren, Haoming Luo, Yuqian Hong, Bryan Dai, Joey Zhou, Kai Qiu, Zhirong Wu, and Chong Luo. Logic-rl: Unleashing llm reasoning with rule-based reinforcement learning. *arXiv preprint arXiv:2502.14768*, 2025.

[40] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 2023.

[41] Yuhang Zhou, Yu He, Siyu Tian, Yuchen Ni, Zhangyue Yin, Xiang Liu, Chuanjun Ji, Sen Liu, Xipeng Qiu, Guangnan Ye, et al. r^3-nl2gql: A model coordination and knowledge graph alignment approach for nl2gql. 2024.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Abstract and Section 1 Introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 5 Limitations.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: The paper does not include theoretical results.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We report the setup throughout the paper as well as in the Section 3.4 and Appendix B.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We use publicly available datasets and the code of our work is fully provided.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We report the training and test details in the Section 3.4, Appendix A and Appendix B

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The LLMs only have one checkpoint, so we only edit once for each setting. But we test our method and baselines under various models, settings, and datasets, therefore, the statistical significance of the experiments can be verified and supported.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: In Appendix B

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We use publicly standard datasets that do not contain information about individual people or offensive context to our knowledge.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: In Appendix C

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

Justification: [NA]

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: Section 3.2. We use publicly available artifacts.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: Our code is provided.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: [NA]

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

Table 5: Dataset description

Dataset	Database	Samples
Training	biology	2289
	soccer	1820
	art	2750
	terrorist attack	1436
CypherBench	geography	366
	flight accident	189
	politics	390
	company	347
	fictional character	385
	movie	401
	nba	270
Neo4j-Text2Cypher	bluesky	39
	buzzoverflow	120
	companies	275
	fincen	146
	gameofthrones	148
	grandstack	182
	movies	203
	neoflix	261
	network	109
	northwind	197
	offshoreleaks	115
	recommendations	253
	stackoverflow2	91
	twitch	185
	twitter	146

A Dataset Description

We conduct our experiments on two representative and diverse datasets: CypherBench and Neo4j-Text2Cypher. These datasets are selected to comprehensively evaluate the generalization ability and query generation performance of our proposed method across different domains and database schemas. For training purposes, we construct our dataset by utilizing the training portion of CypherBench, ensuring that the model is exposed to a wide variety of graph structures and query patterns during learning.

Table 5 provides a detailed summary of the databases involved and the number of samples available for each. The training dataset consists of four domains: biology, soccer, art, and terrorist attack, comprising a total of 8,295 examples. This diverse collection enables the model to learn from a broad spectrum of semantic structures and query intents.

The CypherBench test set, designed to evaluate zero-shot generalization, includes seven distinct domains such as geography, flight accident, politics, company, fictional character, movie, and NBA. Each domain presents unique challenges in terms of entity types, relationships, and query complexity, making it a robust benchmark for assessing model performance on unseen data.

The Neo4j-Text2Cypher dataset, which covers 15 diverse domains, including real-world and synthetic graphs like Bluesky, BuzzOverflow, Fincen, Game of Thrones, Northwind, and Twitter. With a total of 2,380 examples, Neo4j-Text2Cypher tests the model's ability to handle heterogeneous graph schemas and varied user query styles .

B Implementation Details

Our training is conduct on $4\times$ Nvidia A800 GPUs, with full parameter optimization and gradient checkpointing. We show some important parameter settings in Table 6.

Table 6: Implementation details of *Cypher-RI*.

Parameter	**Value**
Base Model	Qwen2.5-Coder-7B
Train Batch Size	1024
Micro Train Batch Size	8
Rollout Batch Size	128
Micro Rollout Batch Size	16
Learning Rate	1e-6
Prompt Max Length	1,024
Generation Max Length	2,000
Initial KL Coefficient	0
Mixed Precision	BF16
Rollout Temperature	1.0
Optimizer	AdamW
Clip Ratio	0.2
Number of Rollout	8

C Social Impacts

This paper introduces Cypher-RI as a foundation model tailored for Text-to-Cypher tasks. Cypher-RI focuses on improving the accuracy, reliability, and generalization of natural language to Cypher query translation, enabling enhanced interaction with structured graph data. Our work aims to facilitate user-friendly access to graph databases through natural language, without posing any potential ethical concerns or negative social impacts.