Synthesizing Text-to-SQL Data from Weak and Strong LLMs

Jiaxi Yang[1,2,*‡], **Binyuan Hui**[3,*], **Min Yang**[1†], **Jian Yang** [3], **Junyang Lin**[3], **Chang Zhou**[3†]
[1] Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences
[2] University of Chinese Academy of Sciences
[3] Alibaba Group
{jx.yang, min.yang}@siat.ac.cn
binyuan.hby@alibaba-inc.com
https://github.com/Yangjiaxi/Sense

Abstract

The capability gap between open-source and closed-source large language models (LLMs) remains challenging in text-to-SQL tasks. In this paper, we introduce a synthetic data approach that amalgamates strong data generated by larger, more potent models (strong models) with weak data produced by smaller, less well-aligned models (weak models). Our approach contributes to the improvement of domain generalization in text-to-SQL models and investigates the potential of weak data supervision through preference learning. Moreover, we utilize the synthetic data approach for instruction tuning on open-source LLMs, yielding SENSE, a specialized text-to-SQL model. The effectiveness of SENSE is substantiated by achieving state-of-the-art results on the SPIDER and BIRD benchmarks, thereby mitigating the performance disparity between open-source models and the methods derived from closed-source models.

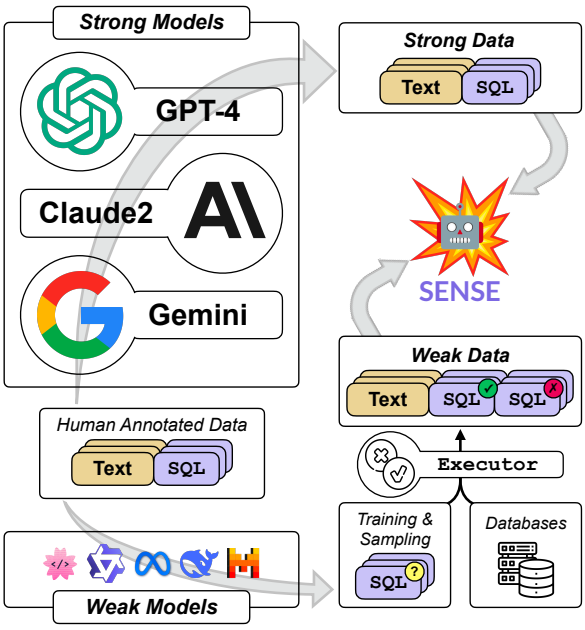

Figure 1: Overview of SENSE: Integrating human-annotated data with synthetic data from strong models for domain diversity, and weak models for preference learning, aligning with executors for enhanced text-to-SQL performance.

1 Introduction

The ability to convert a natural language question into a structured query language (SQL), i.e., text-to-SQL (Zhong et al., 2017; Yu et al., 2018; Li et al., 2023c; Qin et al., 2022), can assist non-experts in interacting with databases using natural language, democratizing data access and analysis. In recent studies (Gao et al., 2023; Zhang et al., 2023), notable accomplishments have been observed in powerful closed-source LLMs, exemplified by GPT-4, employing a range of prompting methods (Wei et al., 2022). However, the adoption of closed-source LLMs introduces concerns pertaining to issues of openness, privacy, and substantial costs.

Recently, the proliferation of numerous open-source LLMs (Touvron et al., 2023; Roziere et al., 2023; Bai et al., 2023) has attracted considerable attention as these models demonstrate comparable capabilities to their closed-source counterparts across a broad spectrum of natural language processing (NLP) tasks. This motivates us to undertake a thorough evaluation of prominent open-source LLMs in the text-to-SQL task, aiming to gauge their viability as alternatives. However, following an assessment utilizing a standardized prompt, we observed that open-source models still exhibit a substantial performance gap compared to closed-source models. In particular, the popular open-source model CodeLLaMA-13B-Instruct demonstrates a 30% lower execution accuracy than GPT-4 on the BIRD (Li et al., 2023c) benchmark.

Developing specialized text-to-SQL models built upon open-source LLMs that attains performance

* Equal contribution.
‡Work done during an intern at Alibaba Group.
†Corresponding authors.

Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 7864–7875

levels comparable to close-source models holds critical importance in contexts marked by sensitivity to policy, privacy, and resource limitations. To this end, we focus on supervised fine-tuning (SFT) to enhance the text-to-SQL capabilities of open-source base models. However, enhancing the text-to-SQL ability of open-source models through SFT remains an open challenge. A significant barrier to this progress is the high cost of achieving text-to-SQL data, which relies on manual expert annotation. The generation of high-quality text-to-SQL fine-tuning data should consider two primary perspectives. First, the inclusion of diverse data aims to facilitate cross-domain generalization, allowing the model to be successfully applied to new domains or databases. Second, alignment with executors becomes crucial to better enable the model to learn SQL from execution feedback, particularly from errors, mirroring how humans often learn from their mistakes.

In response to the data scarcity challenge, numerous endeavors (Wang et al., 2023; Xu et al., 2024; Wei et al., 2023) have sought to generate synthetic data utilizing larger and more powerful LLMs (strong models), such as GPT-4, creating what is denoted as **strong data**. Although strong data inherently contributes to the enhancement of data diversity (Xu et al., 2024), a critical factor for the domain generalization of models, its application in the text-to-SQL task remains unexplored. Additionally, the generation of valuable erroneous text-to-SQL data poses a separate challenge. Strong models often exhibit significant efforts toward correct alignment and safety, making it difficult to elicit erroneous samples. Consequently, we redirect our focus towards smaller, less well-aligned open-source models (weak models). Weak models produce valuable weak SQL samples, which can subsequently be validated and subjected to error induction with the assistance of executors. Preference learning (Rafailov et al., 2023) is employed to instruct language models to learn from both correct and incorrect samples, constituting what we refer to as **weak data**.

To verify the effectiveness of our SENSE, we conduct SFT on a popular open-source base model, i.e., CodeLLaMA (Roziere et al., 2023), and obtain a new specialized model named SENSE. We comprehensively evaluate SENSE's performance on the text-to-SQL tasks, achieving state-of-the-art (SOTA) results on both the standard benchmark Spider (Yu et al., 2018) and the challenging benchmark BIRD (Li et al., 2023c), narrowing the gap between open-source and closed-source models. Additionally, we evaluate SENSE on three robustness datasets: SYN (Gan et al., 2021a), REALISTIC (Deng et al., 2021), and DK (Gan et al., 2021b), demonstrating its advantages in robustness. Moreover, we conduct an in-depth experimental analysis offers insights into the influence of synthetic data on model performance. In summary, our contributions are threefold:

- We first evaluate both open-source and closed-source LLMs on text-to-SQL benchmarks using a standardized prompt. We observe that the text-to-SQL capabilities of open-source models were significantly inferior. It motivated us to train our specialized model SENSE through SFT on an open-source LLM.
- We propose a synthetic data approach that uses strong models to generate strong data to enhance data diversity and employs weak models to generate weak data combined with an executor to learn from feedback.
- Extensive experiments shows the effectiveness of SENSE, achieving SOTA performance, even competing with methods based on GPT-4. We believe that making these data and models publicly available can contribute to the advancement of the text-to-SQL community.

2 Preliminaries

Notation Definition The objective of the text-to-SQL task is to convert a natural language (NL) question Q into the corresponding SQL query Y, grounded in the database schema $\mathcal{S} = \langle \mathcal{T}, \mathcal{C}, \mathcal{V} \rangle$, which includes table names $\mathcal{T}$, column names $\mathcal{C}$, database values $\mathcal{V}$. In more challenging tasks such as BIRD (Li et al., 2023c), understanding NL questions or database values may require external knowledge, represented as $\mathcal{K}$. The current popular text-to-SQL task employs a cross-domain setting to evaluate a model's ability to generalize to new domains, ensuring there is no overlap between the domains of the training, development, and test sets.

Prompt Construction Achieving consistent results in text-to-SQL tasks with LLMs requires a standardized prompt structure to allow fair model comparisons. Following the guidelines from (Chang and Fosler-Lussier, 2023) and illustrated in Fig 2, our prompt comprises four elements:

Dataset	#Examples	#Databases	#Examples/#Databases	Avg(#Tokens)	Avg(#JOIN)
Spider	7000	140	50.0	37.3	0.54
Bird	9428	69	136.6	64.0	1.02
Synthetic	8216	503	16.3	60.3	1.13

Table 1: Statistical information of the strong data in supervised fine-tuning stage. For synthetic data, similar databases has been merged based on semantic similarity.

Prompt template for text-to-SQL tasks.

```
CREATE TABLE "list" (
  "LastName" TEXT,
  "FirstName" TEXT,
  "Grade" INTEGER,
  "Classroom" INTEGER,
  PRIMARY KEY(LastName, FirstName)
);
/* 3 example rows:
SELECT * FROM list LIMIT 3;
LastName     FirstName  Grade   Classroom
CAR          MAUDE      2       101
KRISTENSEN   STORMY     6       112
VANDERWOUDE  SHERWOOD   3       107
*/
(...other tables omitted...)
-- External Knowledge: ...
-- Using valid SQLite and understanding
    External Knowledge, answer the following
    questions for the tables provided above.
Question: How many students are there?

SELECT count(*) FROM list;
```

Figure 2: Unified prompt (Chang and Fosler-Lussier, 2023) template for text-to-SQL tasks.

database schema, task instructions, optional external knowledge, and the natural language question. We employ the `CreateTable` method for detailing database schemas and `SelectRow` to showcase table contents, ensuring SQL keywords and schema are presented uniformly. Task instructions are clear: "– *Using valid SQLite, answer the following questions for the tables provided above.*" External knowledge, if necessary, precedes the question. This standardized prompt, denoted as X, is designed to serve as the input to the LLM.

3 Methodology

Overall, our approach is divided into two distinct phases. Initially, we enhance the base model's text-to-SQL capabilities through *Supervised Fine-tuning (SFT)*, with a primary focus on the diversity and quality of data. We refer to this portion of data as **strong data**. Subsequently, we employ *Preference Learning* to inspire the model to learn from incorrect SQLs, which we denote them as **weak data**, necessitating the use of weaker language models for error generation. The details of these two processes are described below.

3.1 Strong Data: Supervised Fine-tuning

Supervised Fine-tuning (SFT) will significantly enhance the model's performance in generating appropriate responses, including text-to-SQL. The currently popular cross-domain datasets, primarily Spider and BIRD, incur high annotation costs due to the need for human experts. To mitigate this and further expand the scale, we turn to the powerful language model GPT-4 for assistance, utilizing prompts to synthesize target data. Given that cross-domain generalization is a central challenge for text-to-SQL, we designed hints to encourage diversity by prompting GPT-4 to generate sufficiently diverse datasets, as shown in Figure 3. As illustrated in Table 1, the ratio of examples per domain in our synthetic dataset is markedly lower than that observed in Spider and Bird, signifying a higher domain diversity. Additionally, the synthetic data features a higher average number of JOIN operations in SQL queries, indicating a greater complexity and depth in the constructed SQLs. These include mechanisms for controlling question difficulty, promoting domain diversity, and explicitly excluding over-represented domains, thereby guiding GPT-4 to generate data points that are not only diverse but also tailored to various levels of complexity.

Given a stong data set $\mathcal{D}_s$ of input prompt $\boldsymbol{x}$ and target response y generated, the supervised fine-tuning could be formulated as the log likelihood loss:

$$\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}_s}\left[\sum_{t=1}^{T}\log p_\theta\left(y_t \mid \boldsymbol{y}_{1:t-1},\boldsymbol{x}\right)\right], \quad (1)$$

where θ is the parameters of language model, and $p_\theta(\boldsymbol{y} \mid \boldsymbol{x}) = \prod_{t=1}^{T} p_\theta\left(y_t \mid \boldsymbol{y}_{<t},\boldsymbol{x}\right)$ is the conditional probability distribution of target SQL sequence y given prompt x. T is the sequence length of y, and t is the auto-aggressive decoding step.

Prompt for Synthesizing Strong Data

Your task is to generate one additional data point at the `{the_level}` difficulty level, in alignment with the format of the two provided data points.
1. **Domain**: Avoid domains that have been over-represented in our repository. Do not opt for themes like Education/Universities, Healthcare/Medical, Travel/Airlines, or Entertainment/Media.
2. **Schema**: Post your domain selection, craft an associated set of tables. These should feature logical columns, appropriate data types, and clear relationships.
3. **Question Difficulty** - `{the_level}`:
   - **Easy**: Simple queries focusing on a single table.
   - **Medium**: More comprehensive queries involving joins or aggregate functions across multiple tables.
   - **Hard**: Complex queries demanding deep comprehension, with answers that use multiple advanced features.
4. **Answer**: Formulate the SQL query that accurately addresses your question and is syntactically correct.
Additional Guidelines:
   - Venture into diverse topics or areas for your questions.
   - Ensure the SQL engages multiple tables and utilizes advanced constructs, especially for higher difficulty levels.
Ensure your submission only contains the Domain, Schema, Question, and Answer. Refrain from adding unrelated content or remarks.
(...examples and generations goes here...)

Figure 3: Prompt for synthesizing strong data. The placeholder `the_level` is filled on-the-fly by program, controlling the desired hardness level of the generated data point. For limited token consideration, we randomly draw two examples from Spider training set as few-shot demonstrations.

3.2 Weak Data: Preference Learning

The second phase involves a more nuanced approach for weak data. Here, we introduce the model to incorrect SQL queries intentionally generated by weaker LLMs. Through Preference Learning (Rafailov et al., 2023), the model is encouraged to discern between correct and incorrect SQL, effectively learning from its mistakes. This process not only refines the model's understanding of SQL syntax but also enhances its resilience to common errors that might occur in real-world scenarios.

Given a natural language description x, we generate an output y' using weaker models (smaller in size and less well-aligned). We then execute y' using an SQL executor E, and if the execution result matches the ground truth y, we consider it a positive sample y_w. Conversely, if the result is inconsistent, we label it as a negative sample y_l.

$$\begin{cases} y_w = y', & \text{if } \mathsf{E}(y') = \mathsf{E}(y) \\ y_l = y', & \text{if } \mathsf{E}(y') \neq \mathsf{E}(y). \end{cases} \tag{2}$$

We construct a dataset $\mathcal{D}_w$ with both positive and negative examples and optimize the model using the recently popular preference learning method, direct preference optimization (DPO, Rafailov et al., 2023). DPO fine-tunes the model directly based on preference data, bypassing the reward modeling stage and aiming to maximize the following objective function:

$$\mathop{\mathbb{E}}_{(x,y_w,y_l)\sim\mathcal{D}_w} \log \sigma \left(\beta \log \frac{p_\theta(y_w|x)}{p_{\text{ref}}(y_w|x)} - \beta \log \frac{p_\theta(y_l|x)}{p_{\text{ref}}(y_l|x)} \right) \tag{3}$$

where p_θ represents the probability distribution of the target model's predictions, p_{ref} denotes the probability distribution from a reference model, and β is a parameter that regulates the extent of the target model's divergence from the reference model. By training on the preference dataset, we can align LLM with executor preferences.

Leveraging the described methodology, we conducted two phases of training using CodeLLaMA-7B and CodeLLaMA-13B, successfully yielding SENSE-7B and SENSE-13B as final models.

4 Experiments

4.1 Evaluation Benchmarks

We evaluated the effectiveness of SENSE using popular text-to-SQL benchmarks across five datasets.

General Benchmark Spider (Yu et al., 2018) comprises 7,000 Text-SQL pairs in its training set and 1,034 pairs in its development set, across 200 different databases and 138 domains.

Challenge Benchmark BIRD (Li et al., 2023c) is a new benchmark of large real-world databases, containing 95 large databases with high-quality Text-SQL pairs, totaling 33.4GB of data across 37 fields. Unlike Spider, BIRD focuses on massive and real database contents, the external knowledge reasoning between natural language questions and database content.

Robust Benchmarks SYN (Gan et al., 2021a) replaces simple string-matched problem tags or pat-

Model / Method	*Spider*			*Bird*	
	Dev-EX	*Dev-TS*	*Test*	*Dev*	*Test*
Prompting Methods w/ Closed-Source LLMs					
PaLM-2 (Anil et al., 2023)	-	-	-	27.4	33.1
Claude-2 (Anthropic, 2023)	-	-	-	42.7	49.0
ChatGPT (OpenAI, 2022)	72.3	-	-	36.6	40.1
GPT-4 (Achiam et al., 2023)	72.9	64.9	-	49.2	54.9
Few-shot SQL-PaLM (Sun et al., 2023)	82.7	77.3	-	-	-
DIN-SQL + GPT-4 (Pourreza and Rafiei, 2023)	82.8	74.2	85.3	50.7	55.9
ACT-SQL + GPT-4 (Zhang et al., 2023)	82.9	74.5	-	-	-
DAIL-SQL + GPT-4 (Gao et al., 2023)	83.5	76.2	**86.6**	54.8	57.4
Fine-tuning Models					
T5-3B + PICARD (Scholak et al., 2021)	79.3	69.4	75.1	-	-
RASAT + PICARD (Qi et al., 2022)	80.5	70.3	75.5	-	-
RESDSQL-3B + NatSQL (Li et al., 2023a)	**84.1**	73.5	79.9	-	-
Fine-tuned SQL-PaLM (Sun et al., 2023)	82.8	78.2	-	-	-
Open-Source LLMs					
LLaMA2-7B (Touvron et al., 2023)	28.0	23.8	-	7.1	-
LLaMA2-7B-Chat (Touvron et al., 2023)	36.9	34.9	-	11.3	-
Qwen-1.8B (Bai et al., 2023)	54.8	48.6	-	13.2	-
LLaMA2-13B-Chat (Touvron et al., 2023)	49.6	45.5	-	14.2	-
LLaMA2-13B (Touvron et al., 2023)	47.4	39.4	-	15.3	-
StarCoder-3B (Li et al., 2023d)	52.7	47.0	-	15.3	-
StarCoder-7B (Li et al., 2023d)	60.7	55.1	-	17.2	-
DeepSeek-Coder-1.3B (Guo et al., 2024)	59.3	53.2	-	22.0	-
♣CodeLLaMA-7B (Roziere et al., 2023)	61.1	52.3	-	22.5	-
♡CodeLLaMA-13B (Roziere et al., 2023)	61.7	53.5	-	22.9	-
CodeLLaMA-7B-Instruct (Roziere et al., 2023)	63.4	54.2	-	23.0	-
DeepSeek-Coder-1.3B-Instruct (Guo et al., 2024)	53.2	48.7	-	24.1	-
StarCoder-15B (Li et al., 2023d)	63.9	57.9	-	24.4	-
CodeLLaMA-13B-Instruct (Roziere et al., 2023)	62.3	52.5	-	24.7	-
Qwen-7B (Bai et al., 2023)	63.6	54.5	-	26.1	-
Ours					
♣SENSE-7B	83.2	81.7	83.5	51.8	59.3
♡SENSE-13B	**84.1**	**83.5**	**86.6**	**55.5**	**63.4**

Table 2: Performance comparison on Spider and Bird benchmarks. To show the relationship of our final presented models and base models, we denote them by ♣ and ♡. Specifically, SENSE-7B is based on CodeLLaMA-7B and SENSE-13B is based on CodeLLaMA-13B.

tern names with their synonyms. DK (Gan et al., 2021b) requires the text-to-SQL parser to have domain knowledge reasoning capabilities. REALISTIC (Deng et al., 2021) replaces mentioned schema items in questions to make them closer to real-world scenarios.

4.2 Evaluation Metrics

For Spider and its robustness benchmarks, we follow Spider's official evaluation protocol [*], using EX (Yu et al., 2018) and TS (Zhong et al., 2020) metrics[†]. EX measures if the SQL output exactly matches the execution result of provided golden SQL. TS is a more reliable metric that confirms if a SQL query passes all EX checks on various tests created through database augmentation. For BIRD, we adopt its official evaluation scripts[‡], focusing on EX accuracy evaluation.

4.3 Compared Methods

We compared a variety of baseline methods, which can be categorized into three categories.

Prompting Methods ACT-SQL (Zhang et al., 2023) introduced a method for automatically generating Chain of Thought (CoT) (Wei et al., 2022) examples. DIN-SQL (Pourreza and Rafiei, 2023) uses prompts to break down the complex text-to-SQL task into smaller subtasks for enhanced performance. DAIL-SQL (Gao et al., 2023) made

[*] https://yale-lily.github.io/spider

[†] TS is not reported for DK due to incompatibility.

[‡] https://bird-bench.github.io

Model / Method	*SYN*		*REALISTIC*		*DK*	*Average*
	EX	*TS*	*EX*	*TS*	*EX*	
RESDSQL-3B+NatSQL	76.9	66.8	81.9	70.1	66.0	72.3
Few-shot SQL-PaLM	74.6	67.4	77.6	72.4	66.5	71.7
Fine-tuned SQL-PaLM	70.9	66.4	77.4	73.2	67.5	71.1
SENSE-7B	72.6	64.9	82.7	75.6	77.9	74.7
SENSE-13B	**77.6**	**70.2**	**84.1**	**76.6**	**80.2**	**77.7**

Table 3: Evaluation of SENSE and various previously proposed methods on Spider-based robustness benchmarks: Spider-SYN, REALISTIC and Spider-DK. TS is not reported for DK due to incompatibility.

Model / Method	*Easy*	*Medium*	*Hard*	*Extra Hard*	*All*
DIN-SQL + GPT-4	91.1	79.8	64.9	43.4	74.2
ACT-SQL + GPT-4	91.1	79.4	67.2	44.0	74.5
DAIL-SQL + GPT-4	90.3	81.8	66.1	50.6	76.2
Few-shot SQL-PaLM	93.5	84.8	62.6	48.2	77.3
Fine-tuned SQL-PaLM	93.5	85.2	68.4	47.0	78.2
SENSE-13B	**95.2**	**88.6**	**75.9**	**60.3**	**83.5**

Table 4: Test Suite accuracy on Spider Dev, categorized by SQL hardness levels.

improvements in question representation, example selection, and sample sequence organization.

Fine-tuning Models PICARD (Scholak et al., 2021) is a constrained decoding approach fine-tuned on T5-3B. RASAT (Qi et al., 2022) and Graphix (Li et al., 2023b) focus on how to incorporate structure information into the fine-tuning process of the T5 model (Raffel et al., 2020), while RESDSQL-3B (Li et al., 2023a) decouples schema linking and skeleton parsing.

Open-Source LLMs Recently, there has been a surge in open-source LLMs. We selected some of the most recently popular LLMs, including various sizes and versions of DeepSeek-Coder (Guo et al., 2024), Qwen (Bai et al., 2023), StarCoder (Li et al., 2023d), LLaMA2 (Touvron et al., 2023), and CodeLLaMA (Roziere et al., 2023). We adopted a unified prompt as shown in Figure 2 to ensure a fair comparison with SENSE.

4.4 Implementation Details

We choose CodeLLaMA-7B and CodeLLaMA-13B as our primary models, and DeepSeek-Coder-1.3B as a weak model to generate preference data. Our experiments were run on 8×A100 GPUs, combining datasets from Spider and Bird with GPT-4-generated data for supervised fine-tuning using the AdamW optimizer with a learning rate of 2e-5 and a cosine warmup scheduler over three epochs. The preference learning phase starts with the generation of the weak data through the fine-tuned weak model and a SQL evaluator. The evaluator recognize each generated SQL as positive or negative, thus further enabled us to craft a preference dataset. This dataset served as the foundation for Direct Preference Optimization (DPO) (Rafailov et al., 2023) training. For DPO training, the Adam optimizer was selected, with the learning rate adjusted to 2e-6 and the β parameter maintained at 0.2, consistent with the original DPO settings.

4.5 Overall Performance

Results on General Settings Table 2 shows prompting methods surpass fine-tuning in text-to-SQL, thanks to closed-source LLMs and tailored prompts. Open-source LLMs lag in generalization. Larger models correlate with better outcomes, and instruction tuning boosts performance, showcasing synthetic data's tuning utility. Remarkably, SENSE achieves state-of-the-art (SOTA) results on the Spider dataset, surpassing the GPT-4-based DAIL-SQL. Specifically, SENSE-13B shows a 21.8% improvement over CodeLLaMA-13B-Instruct in the development set and marginally exceeds DAIL-SQL, indicating SENSE's potential in narrowing the performance gap between open and closed-source models in text-to-SQL challenges.

Results on Challenge Settings Experiments on BIRD reveal its complexity, none of the open-source LLMs perform well on BIRD, yet SENSE-13B sets a new standard, outperforming DAIL-SQL by 5.98% on the test set, as Table 2 shows. This

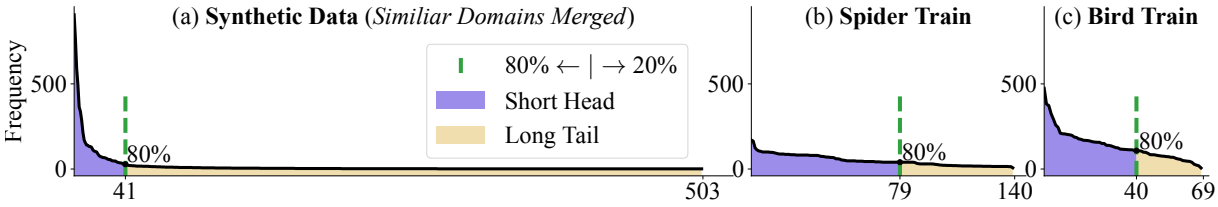

Figure 4: Domain density comparison. This visualization sorts domains by example count, showcasing a long-tail distribution to highlight the broad diversity within our synthetic dataset.

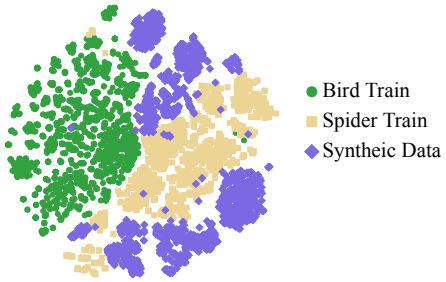

Figure 5: 2-D t-SNE visualization comparing original and synthetic data's last-layer hidden representations post-supervised fine-tuning on last token.

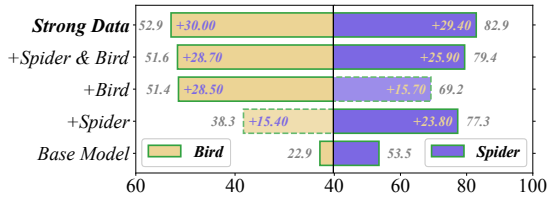

Figure 6: Bird dev and Spider dev scores for different Supervised fine-tuning data on CodeLLaMA-13B. We report EX and TS for Bird and Spider, respectively.

highlights the benefits of specialized open-source LLMs for challenging environments.

Results on Robust Settings Table 3 reveals SENSE excels in robustness (SYN, DK, REALISTIC) even without extra training. SENSE-7B and SENSE-13B lead, surpassing RESDSQL-3B by 1.4% and 5.4% on average. Notably, SENSE's strength in DK suggests synthetic data effectively leverages the base model's domain knowledge.

4.6 Fine-grained Analysis on Hardness

Spider's difficulty labels reveal SENSE-13B's superiority across all levels, as shown in Table 4. Performance gains over the best alternatives are notable: Easy (1.6%), Medium (3.4%), Hard (7.5%), and Extra Hard (9.7%). This suggests that it has an advantage in processing hard samples, benefiting from the difficulty control in synthetic data prompt.

4.7 Ablation Study

Table 5 presents an ablation study on SENSE, dissecting its components to assess their individual impact. The study focuses on three key topics.

Model	*Spider-Dev* *EX*	*TS*	*Bird Dev*
Ablation for Main Results			
DeepSeek-Coder-1.3B♣	79.1	80.2	40.7
w/o **Strong**	59.3	53.2	22.0
SENSE-7B	83.2	81.7	51.8
w/o **Weak**	82.3	81.6	49.9
w/o **Strong**	61.1	52.3	22.5
SENSE-13B	84.0	83.5	55.5
w/o **Weak**	83.9	82.9	52.9
w/o **Strong**	61.7	53.5	22.9
Transferability on Homogeneous Models			
Qwen-1.8B♣	76.2	75.2	38.1
w/o **Strong**	54.8	48.6	13.2
SENSE^{Q}-7B	84.2	84.6	52.5
w/o **Weak**	83.9	83.4	50.5
w/o **Strong**	63.1	54.4	26.1

Table 5: Ablation and transferability results. Upper section details the effects of excluding **weak** and **strong data** in fine-tuning. Lower section assesses model transferability, using Qwen-1.8B and Qwen-7B. ♣ marks a model fine-tuned with strong data.

Why Strong Data is Helpful? Analysis from Figure 6 and Table 5 shows strong data significantly boosts Spider accuracy due to its simpler SQL queries and the emphasis on domain generalization, and it can be seen that the data of bird and spider can mutually enhance each other when the data in the corresponding field is absent. Figure 4 illustrates strong data's broader long-tail distribution, enabled by LLMs' stored knowledge, enhancing SENSE's ability to adapt to new domains. Additionally, t-SNE visualizations in Figure 5 highlight synthetic samples' role in bridging the gaps left by human-annotated data, further validating the value of synthetic data.

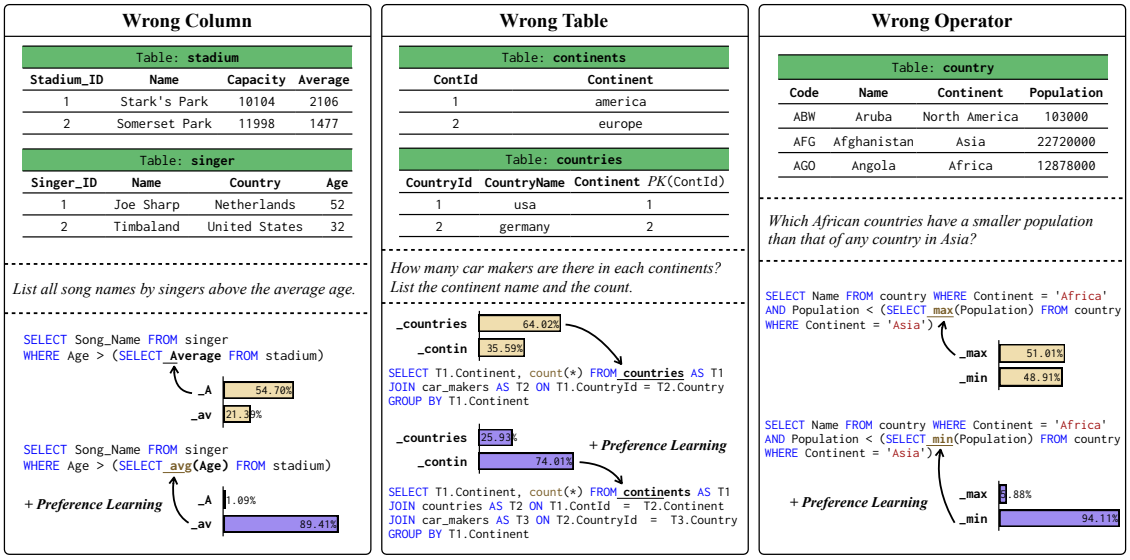

Figure 7: Preference learning enhances text-to-SQL performance in critical tokens.

Model	*MMLU* 5-shot	*ARC-Challenge* 25-shot	*GSM8K* 8-shot, CoT	*HumanEval* Greedy	*Average*
CodeLLaMA-7B	39.2	42.1	11.4	29.3	30.5
SENSE-7B	39.1	42.5	11.0	31.7	31.1
CodeLLaMA-13B	43.7	46.4	21.8	34.2	36.5
SENSE-13B	42.7	47.9	18.1	40.2	37.2

Table 6: Performance of SENSE and corresponding base models on MMLU, ARC-Challenge, GSM8K and HumanEval.

Why Weak Data is Helpful? Weak data, when used with preference learning, helps SENSE to refine its output by learning from errors, ensuring closer alignment with the SQL executor. As indicated in Table 5, weak data significantly enhances overall performance, particularly by improving the complexity of SQL queries generated, with a notable 4.9% uplift on BIRD when compared to strong data-trained models. Further, Figure 7 and case studies reveal weak data's role in reducing hallucinations in SQL generation, minimizing errors in selecting columns, tables, and operators, vital for crafting intricate SQL commands.

Transferablity across Different LLMs In Table 2, SENSE is initialized with CodeLLaMA and utilizes the smaller DeepSeek-Coder as the generator for weak data. From the perspective of model differences, CodeLLaMA and DeepSeek-Coder can be considered heterogeneous models due to their distinct structural details and pre-training data. This make our curiosity about whether SENSE could effectively transfer to homogeneous models with identical pre-training data. We selected the Qwen series, a family of open-source models with a rich variety of sizes. We used Qwen-7B as the base model and Qwen-1.8B as the weak model for synthesizing, creating a new variant, SENSE^{Q}. As shown in Table 5, we found that the approach of using synthetic data works equally well under homogeneous models, demonstrating the same level of improvement as SENSE. This confirms the transferability of the proposed method.

Performance on General and Board Tasks In addition to evaluating the performance of SENSE on several text-to-SQL tasks, we assessed its generalization capabilities through experiments on several benchmarks: MMLU (Hendrycks et al., 2021) for language understanding, ARC-Challenge (Clark et al., 2018) for common sense reasoning, GSM8K (Cobbe et al., 2021) for math reasoning, and HumanEval (Chen et al., 2021) for code generation. The results, presented in Table 6, highlight SENSE's competitive performance across these diverse tasks, demonstrating its broad applicability beyond the SQL domain. Notably, while SENSE maintaining competitive performance on math reasoning, the SENSE-13B exhibits significant improvement in code generation performance

compared to its base models. These findings underscore SENSE's robust versatility and generalization capabilities across various tasks. Furthermore, our experiments show that SENSE models maintain performance on MMLU tasks, even when specifically fine-tuned for text-to-SQL tasks, confirming our proposed method won't affect the knowledge stored in the LLMs. Moreover, if there is a need to enhance NLP task performance, additional general data could be incorporated through instruction fine-tuning, though this is beyond the scope of this paper.

5 Related Work

Text-to-SQL Parsing In the realm of text-to-SQL parsing, early methods like IRNET (Guo et al., 2019) focused on learning representations using attention-based models, while subsequent works introduced fine-tuning based models (Bogin et al., 2019; Wang et al., 2020; Cao et al., 2021; Hui et al., 2022; Li et al., 2023a; Liu et al., 2021; Shi et al., 2021). Other advancements (Shaw et al., 2021; Scholak et al., 2021; Xie et al., 2022) have leveraged T5, demonstrating their effectiveness. Recently, the emergence of LLMs(OpenAI, 2022; Achiam et al., 2023; Anil et al., 2023; Anthropic, 2023) has garnered significant attention. Building on top of these models, various works have explored innovative prompting techniques. Such as the automatic generation of Chain-of-Thought (Wei et al., 2022) examples by ACT-SQL (Zhang et al., 2023), the decomposition of complex tasks into sub-tasks by DIN-SQL (Pourreza and Rafiei, 2023), and sample organization by DAIL-SQL (Gao et al., 2023) have significantly improved performance in the text-to-SQL domain. TAP4LLM (Sui et al., 2023) proposed a table provider to better perform semi-structured data reasoning. (Tai et al., 2023) study how to enhance reasoning ability CoT style prompting. Our work sets apart by leveraging open-source LLMs, matching the prowess of proprietary models in text-to-SQL tasks.

Synthetic Data There are many methods have investigated the usage of LLMs to synthesize data. Self-Instruct (Wang et al., 2023) introduced a framework for improving the instruction-following capabilities. Yue et al. (2024) managed to use hybrid rationales in developing advanced math models. Yu et al. (2024) generated plenty of mathematical questions by rewriting from multiple aspects. Yuan et al. (2024) uses supervised models to bootstrap more augmented samples for math. Our approach, distinct from others, utilizes both large and smaller models for data generation, showcasing efficacy in text-to-SQL tasks.

6 Conclusion

In this paper, we proposed a novel model SENSE to explore synthetic data in text-to-SQL parsing. By combining synthetic strong data from larger models with weak data from smaller models, SENSE enhances domain generalization and learns from executor feedback through preference learning. Extensive experiments demonstrate that SENSE achieves state-of-the-art performance on well-known benchmarks, significantly narrowing the gap between open-source models and closed-source models. The release of SENSE data and models aims to further the progress in the text-to-SQL domain, highlighting the potential of open-source LLM to be fine-tuned with synthetic data.

Limitations

Although our method has shown promising results and significant progress in various aspects, it's crucial to explore the potential limitations. Firstly, due to limited computational resources and time constraints, we were unable to fine-tune our method on larger language models, such as LLaMA2-70B. The effectiveness of synthesizing data on larger models remains unclear. Secondly, our evaluation mainly focused on the text-to-SQL task. However, the potential of our data synthesis technique across diverse tasks, including code generation and math problems, which also benefit from execution-based validation, remains to be fully examined.

Acknowledgments

Min Yang was supported by National Key Research and Development Program of China (2022YFF0902100), National Natural Science Foundation of China (62376262), the Natural Science Foundation of Guangdong Province of China (2024A1515030166), Shenzhen Science and Technology Innovation Program (KQTD20190929172835662), Shenzhen Basic Research Foundation (JCYJ20210324115614039). This work was supported by Alibaba Group through Alibaba Research Intern Program. This work was supported by Alibaba Group through Alibaba Research Intern Program.

References

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

Rohan Anil, Andrew M Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, et al. 2023. Palm 2 technical report. *arXiv preprint arXiv:2305.10403*.

Anthropic. 2023. Claude 2. Technical report, Anthropic.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Ben Bogin, Jonathan Berant, and Matt Gardner. 2019. Representing schema structure with graph neural networks for text-to-SQL parsing. In *Proc. of ACL*.

Ruisheng Cao, Lu Chen, Zhi Chen, Yanbin Zhao, Su Zhu, and Kai Yu. 2021. LGESQL: Line graph enhanced text-to-SQL model with mixed local and non-local relations. In *Proc. of ACL*.

Shuaichen Chang and Eric Fosler-Lussier. 2023. How to prompt LLMs for text-to-SQL: A study in zero-shot, single-domain, and cross-domain settings. In *NeurIPS 2023 Second Table Representation Learning Workshop*.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Xiang Deng, Ahmed Hassan Awadallah, Christopher Meek, Oleksandr Polozov, Huan Sun, and Matthew Richardson. 2021. Structure-grounded pretraining for text-to-SQL. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1337–1350, Online. Association for Computational Linguistics.

Yujian Gan, Xinyun Chen, Qiuping Huang, Matthew Purver, John R. Woodward, Jinxia Xie, and Pengsheng Huang. 2021a. Towards robustness of text-to-SQL models against synonym substitution. In *Proc. of ACL*.

Yujian Gan, Xinyun Chen, and Matthew Purver. 2021b. Exploring underexplored limitations of cross-domain text-to-SQL generalization. In *Proc. of EMNLP*.

Dawei Gao, Haibin Wang, Yaliang Li, Xiuyu Sun, Yichen Qian, Bolin Ding, and Jingren Zhou. 2023. Text-to-sql empowered by large language models: A benchmark evaluation. *arXiv preprint arXiv:2308.15363*.

Daya Guo, Qihao Zhu, Dejian Yang, Zhenda Xie, Kai Dong, Wentao Zhang, Guanting Chen, Xiao Bi, Y Wu, YK Li, et al. 2024. Deepseek-coder: When the large language model meets programming–the rise of code intelligence. *arXiv preprint arXiv:2401.14196*.

Jiaqi Guo, Zecheng Zhan, Yan Gao, Yan Xiao, Jian-Guang Lou, Ting Liu, and Dongmei Zhang. 2019. Towards complex text-to-SQL in cross-domain database with intermediate representation. In *Proc. of ACL*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Binyuan Hui, Ruiying Geng, Lihan Wang, Bowen Qin, Yanyang Li, Bowen Li, Jian Sun, and Yongbin Li. 2022. S^2sql: Injecting syntax to question-schema interaction graph encoder for text-to-sql parsers. In *Proc. of ACL Findings*.

Haoyang Li, Jing Zhang, Cuiping Li, and Hong Chen. 2023a. Resdsql: Decoupling schema linking and skeleton parsing for text-to-sql. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Jinyang Li, Binyuan Hui, Reynold Cheng, Bowen Qin, Chenhao Ma, Nan Huo, Fei Huang, Wenyu Du, Luo Si, and Yongbin Li. 2023b. Graphix-t5: mixing pre-trained transformers with graph-aware layers for text-to-sql parsing. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'23/IAAI'23/EAAI'23. AAAI Press.

Jinyang Li, Binyuan Hui, GE QU, Jiaxi Yang, Binhua Li, Bowen Li, Bailin Wang, Bowen Qin, Ruiying Geng, Nan Huo, Xuanhe Zhou, Chenhao Ma, Guoliang Li, Kevin Chang, Fei Huang, Reynold Cheng, and Yongbin Li. 2023c. Can LLM already serve as a database interface? a BIg bench for large-scale database grounded text-to-SQLs. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Raymond Li, Loubna Ben Allal, Yangtian Zi, Niklas Muennighoff, Denis Kocetkov, Chenghao Mou, Marc Marone, Christopher Akiki, Jia Li, Jenny Chim, et al. 2023d. Starcoder: may the source be with you! *Transactions on Machine Learning Research*.

Qian Liu, Dejian Yang, Jiahui Zhang, Jiaqi Guo, Bin Zhou, and Jian-Guang Lou. 2021. Awakening latent grounding from pretrained language models for semantic parsing. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1174–1189, Online. Association for Computational Linguistics.

OpenAI. 2022. Introducing ChatGPT.

Mohammadreza Pourreza and Davood Rafiei. 2023. DIN-SQL: Decomposed in-context learning of text-to-SQL with self-correction. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Jiexing Qi, Jingyao Tang, Ziwei He, Xianpeng Wan, Yu Cheng, Chenghu Zhou, Xinbing Wang, Quanshi Zhang, and Zhouhan Lin. 2022. Rasat: Integrating relational structures into pretrained seq2seq model for text-to-sql. In *Proc. of EMNLP*.

Bowen Qin, Binyuan Hui, Lihan Wang, Min Yang, Jinyang Li, Binhua Li, Ruiying Geng, Rongyu Cao, Jian Sun, Luo Si, et al. 2022. A survey on text-to-sql parsing: Concepts, methods, and future directions. *arXiv preprint arXiv:2208.13629*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*.

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, et al. 2023. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.

Torsten Scholak, Nathan Schucher, and Dzmitry Bahdanau. 2021. PICARD: Parsing incrementally for constrained auto-regressive decoding from language models. In *Proc. of EMNLP*.

Peter Shaw, Ming-Wei Chang, Panupong Pasupat, and Kristina Toutanova. 2021. Compositional generalization and natural language variation: Can a semantic parsing approach handle both? 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 922–938, Online. Association for Computational Linguistics.

Peng Shi, Patrick Ng, Zhiguo Wang, Henghui Zhu, Alexander Hanbo Li, Jun Wang, Cicero Nogueira dos Santos, and Bing Xiang. 2021. Learning contextual representations for semantic parsing with generation-augmented pre-training. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13806–13814.

Yuan Sui, Jiaru Zou, Mengyu Zhou, Xinyi He, Lun Du, Shi Han, and Dongmei Zhang. 2023. Tap4llm: Table provider on sampling, augmenting, and packing semi-structured data for large language model reasoning. *arXiv preprint arXiv:2312.09039*.

Ruoxi Sun, Sercan O Arik, Hootan Nakhost, Hanjun Dai, Rajarishi Sinha, Pengcheng Yin, and Tomas Pfister. 2023. Sql-palm: Improved large language modeladaptation for text-to-sql. *arXiv preprint arXiv:2306.00739*.

Chang-Yu Tai, Ziru Chen, Tianshu Zhang, Xiang Deng, and Huan Sun. 2023. Exploring chain of thought style prompting for text-to-SQL. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5376–5393, Singapore. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

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

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508, Toronto, Canada. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.

Yuxiang Wei, Zhe Wang, Jiawei Liu, Yifeng Ding, and Lingming Zhang. 2023. Magicoder: Source code is all you need. *arXiv preprint arXiv:2312.02120*.

Tianbao Xie, Chen Henry Wu, Peng Shi, Ruiqi Zhong, Torsten Scholak, Michihiro Yasunaga, Chien-Sheng Wu, Ming Zhong, Pengcheng Yin, Sida I. Wang, Victor Zhong, Bailin Wang, Chengzu Li, Connor Boyle, Ansong Ni, Ziyu Yao, Dragomir Radev, Caiming

Xiong, Lingpeng Kong, Rui Zhang, Noah A. Smith, Luke Zettlemoyer, and Tao Yu. 2022. UnifiedSKG: Unifying and multi-tasking structured knowledge grounding with text-to-text language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 602–631, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, Qingwei Lin, and Daxin Jiang. 2024. WizardLM: Empowering large pre-trained language models to follow complex instructions. In *The Twelfth International Conference on Learning Representations*.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng YU, Zhengying Liu, Yu Zhang, James Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2024. Metamath: Bootstrap your own mathematical questions for large language models. In *The Twelfth International Conference on Learning Representations*.

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

Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting Dong, Keming Lu, Chuanqi Tan, Chang Zhou, and Jingren Zhou. 2024. Scaling relationship on learning mathematical reasoning with large language models.

Xiang Yue, Xingwei Qu, Ge Zhang, Yao Fu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. 2024. MAmmoTH: Building math generalist models through hybrid instruction tuning. In *The Twelfth International Conference on Learning Representations*.

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

Ruiqi Zhong, Tao Yu, and Dan Klein. 2020. Semantic evaluation for text-to-sql with distilled test suite. In *The 2020 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Victor Zhong, Caiming Xiong, and Richard Socher. 2017. Seq2SQL: Generating structured queries from natural language using reinforcement learning. In *CoRR abs/1709.00103*.