Text2Sql: Pure Fine-Tuning and Pure Knowledge Distillation

Gaoyu Zhu[1,2] Wei Shao[1,2] Xichou Zhu[1,2]
Lei Yu[1] Jiafeng Guo[1] Xueqi Cheng[1]

[1]CAS Key Lab of Network Data Science and Technology, ICT, CAS
[2]University of Chinese Academy of Sciences
{zhugaoyu23s, shaowei23s, zhuxichou22s, yulei, guojiafeng, cxq}@ict.ac.cn

Abstract

Text2Sql is a task that converts natural language questions into SQL queries. In previous research on LLM fine-tuning, researchers typically input both the entire database schema and the natural language question into the model. This approach has two issues: 1) the model's context is limited when dealing with a large number of database tables; 2) the question is often related to only a few tables, leading to excessive irrelevant information that distracts the model. To address these issues, we employed pure fine-tuning strategy to reduce redundancy. The model fine-tuned with pure prompts, using prompts that are only 53% of the baseline length, outperforms the baseline (fine-tuned with all tables in the prompt) by 8.2% and 8.6% in Test-suite accuracy (TS) and exact-set-match accuracy (EM), respectively, on the Spider dev set. Using the most refined set of prompts for the Spider dev dataset, the model achieves TS and EM scores of 73.5% and 75.4%, respectively, approaching state-of-the-art (SOTA) levels. To leverage the capabilities of the model with pure prompts, we applied pure knowledge distillation strategy to transfer its abilities. The distilled student model achieved a 1.9% improvement in TS, while the teacher model's prompt length was only 23% of that of the student model.

1 Introduction

Text2Sql is a task that translates natural language questions and database schemas into SQL. It can effectively assist database administrators and even enable ordinary users to access databases using natural language, without requiring professional SQL knowledge (Sun et al., 2023).

Early Text2Sql datasets were relatively simple, with SQL statements often involving only a single table and no nested queries (Zhong et al., 2017). As a result, some research treats the Text2Sql task as multiple classification tasks, predicting aggregation functions and conditions separately (Lyu et al., 2020).

Recently, the emergence of Large Language Models (LLMs) (Achiam et al., 2023; Dubey et al., 2024; Bai et al., 2023) and their powerful semantic representation capabilities have led to a shift in the research paradigm of Text2Sql. Current research primarily focuses on two aspects: contextual learning and fine-tuning. In the area of contextual learning, Din-SQL (Pourreza and Rafiei, 2024) addresses the gap between natural language and SQL by decomposing the Text2Sql task into four sub-problems, with each sub-problem interacting with the LLM to generate the SQL statement corresponding to the natural language question. Dail-SQL (Gao et al., 2023) takes into account both the similarity of example questions and queries when selecting few-shot examples, prioritizing those with higher similarity for interaction with the LLM to retrieve SQL.

SQL-PaLM (Sun et al., 2023) uses retrieval or program-assisted methods to select table and column information from the database, taking into account the limited length of LLM prompts. In the field of fine-tuning. RASAT (Qi et al., 2022) modifies the self-attention mechanism in the T5 model (Raffel et al., 2020) to relation-aware version, with the model input including a prompt containing database information and the question, as well as an interaction graph of relationships between tokens. (Rai et al., 2023) add additional tokens to the natural language to represent semantic boundaries, as well as extra characters to the queries, tables, and columns in the schema to make tokenization more meaningful. SQL-PaLM (Sun et al., 2023) uses a large Palm model for fine-tuning, taking into account the impact of data diversity and synthetic data.

This paper focuses on the fine-tuning aspect of Text2Sql. In Text2Sql fine-tuning tasks, the prompt must include both the question and the database

Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Industry Track), pages 54–61

schema, with the model analyzing the relationship between them to generate the query. To our knowledge, existing Text2Sql fine-tuning methods typically use all tables and columns in the entire database as part of the prompt. This approach has two main issues:

1. When the database contains many tables, the model may struggle to handle such a long context.

2. Not all tables and columns in the database are relevant to the question. Including irrelevant information increases computational costs and distracts the model from focusing on the key tables and columns, thereby degrading performance.

Therefore, it is important to identify which information in the database is useful during fine-tuning, in order to eliminate unnecessary data and shorten the prompts. We refer to the fine-tuning approach aimed at reducing prompt redundancy as pure fine-tuning.

We conducted experiments with the LLaMA 3.2 3B and 1B models on the Spider dataset (Yu et al., 2018b). The model fine-tuned with pure prompts, using prompts that are only 53% of the baseline length, outperforms the baseline (fine-tuned with all tables in the prompt) by 8.2% and 8.6% in Test-suite accuracy (TS) and exact-set-match accuracy (EM) (Zhong et al., 2020), respectively, on the Spider dev set. Using the most refined set of prompts for the Spider dev dataset, the model achieves TS and EM scores of 73.5% and 75.4%, respectively, approaching state-of-the-art (SOTA) levels. To leverage the capabilities of the model with pure prompts, we applied pure knowledge distillation strategy to transfer its abilities. The distilled student model achieved a 1.9% improvement in TS, with the teacher model's prompt length being only 23% of the student model.

In summary, our contributions are as follows:

1. We propose pure fine-tuning strategy that reduces redundant information in the database within the prompts. Our experiments show that overly pure prompts can impair the model's discriminative ability when faced with redundant information, leading to poorer performance. On the other hand, prompts with too much redundant information can distract the model from focusing on the key details, resulting in mediocre performance. We recommend including a small number of irrelevant tables alongside the relevant ones during fine-tuning. This approach improves model performance while significantly reducing the context length.

2. We have empirically verified that higher prompt purity leads to better model performance. To harness the model's capabilities under pure prompts, we propose a strategy called pure knowledge distillation.

2 Related Work

Text2Sql LLMs possess extensive world knowledge and, when given context for generating SQL from text, can respond based on the question and database information. Since LLMs generate different responses to different prompts, researchers have explored prompt engineering in both closed-source and open-source models to obtain high-quality responses (Pourreza and Rafiei, 2024; Gao et al., 2023; Sun et al., 2023; Dong et al., 2023). Prompt engineering involves providing examples to the LLM, and more examples result in higher computational costs. Some researchers have explored SFT for LLMs, allowing the model to generate SQL without requiring examples (Scholak et al., 2021; Qi et al., 2022; Li et al., 2023a). Others argue that a significant gap exists between natural language and SQL, and they bridge this gap using intermediate representations (Yu et al., 2018a; Guo et al., 2019; Herzig et al., 2021; Gan et al., 2021). Since the generated SQL must conform to SQL syntax and use tables and columns specified in the question, some researchers have applied constrained decoding to correct model outputs (Scholak et al., 2021; Sun et al., 2023; Lin et al., 2020).

Knowledge Distillation(KD) Knowledge distillation (Hinton, 2015) is a technique for transferring knowledge from a larger model to a smaller one (Rusu et al., 2015; Sanh, 2019). Standard distillation involves aligning the distributions of the teacher and student models (Song et al., 2020; Zhang et al., 2023; Liang et al., 2020; Gu et al., 2023). Some studies optimize the student model by fitting the intermediate states or attention scores of both the teacher and student (Sun et al., 2019; Jiao et al., 2019; Wang et al., 2020b,a). Others introduce a task module to the intermediate states and optimize the student by aligning the task distributions of the teacher and student (Liang et al.,

2023). Additionally, some researchers use symbolic knowledge distillation, where data generated by the teacher is used to directly fine-tune the student (Li et al., 2023b; Chen et al., 2024).

3 Method

The fine-tuning task for Text2Sql involves a training dataset consisting of a serialized input set X and a corresponding SQL output set Y, with a total of n data points. The i-th element in X is denoted as x_i, and the i-th element in Y is denoted as y_i. As shown in Listing 1, x_i includes database information in black, a natural language question in green, and some auxiliary information in red. The goal of the fine-tuning task is to maximize the log-likelihood of generating X given Y.

$$\max_{\theta} \sum_{i=1}^{n} logP_{\theta}(y_i/x_i) \tag{1}$$

Listing 1: Example of Prompt

```
Given the following database schema:
CREATE TABLE ‘Products_Booked‘
(‘booking_id‘ INTEGER NOT NULL,
‘product_id‘ INTEGER NOT NULL,
‘returned_yn‘ VARCHAR(1),
‘returned_late_yn‘ VARCHAR(1),
‘booked_count‘ INTEGER,
‘booked_amount‘ FLOAT NULL,);

Answer the following:What are the maximum,
    minimum, and average booked count for the
    products booked?
answer:
```

In previous studies (Qi et al., 2022; Sun et al., 2023), database information (db) is typically presented in natural language format, such as:

$$db = T_1 : c_1^1, ..., c_{n_{col}^1}^1 | T_2 : c_1^2, ..., c_{n_{col}^2}^2 | ... \tag{2}$$

T_i represents the i-th table, c_i^j represents the i-th column of the j-th table, and n_{col}^j denotes the number of columns in the j-th table. Tables and columns are separated by colons, columns by commas, and each table by a vertical bar. Additional information about column types and database contents can also be included after the columns. Primary and foreign key relationships can be represented either through natural language descriptions or graphs. While this approach can capture all database information, it is relatively complex and requires an additional converter to translate table creation statements into this format. In contrast, following the Dail-SQL practice (Gao et al., 2023), we directly use SQL statements for table creation to represent the database. The prompt format is shown in Listing 1.

Pure Fine-tuning: In Text2Sql tasks, a database may contain many tables, but only a few are typically relevant to a specific query. Using all tables in the database as prompts for fine-tuning can result in high computational costs, excessively long contexts, and degraded model performance. To address this, we categorize prompts into four levels based on their information purity:

- Level 1: Includes only the tables and columns relevant to the query.
- Level 2: Includes only the tables relevant to the query.
- Level 3: Includes the tables relevant to the query as well as some irrelevant tables.
- Level 4: Includes all tables in the database.

We fine-tuned the model on the same dataset using these four types of prompts and evaluated it on Levels 1, 3, and 4. We refer to the fine-tuning approach that uses higher-purity prompts as pure fine-tuning. We extend Equation 1 as follows:

$$\max_{\theta} \sum_{i=1}^{n} logP_{\theta}(y_i/x_i^{l_j}) \tag{3}$$

l_j represents the prompt at the j-th level.

Pure Knowledge Distillation(Pure-KD): We found that models often exhibit stronger capabilities when using pure prompts. To leverage models under pure prompts, we employ distillation techniques. In traditional distillation, both the teacher model and student model use the same dataset during the distillation process. Unlike traditional methods, our strategy uses pure prompts for the teacher model and impure prompts for the student model, while keeping the same labels for both. In this setup, the teacher model exhibits the strongest capability, reducing the context and effectively lowering computational costs. We refer to this approach as pure knowledge distillation. The objective of distillation in this scenario is as follows:

$$\max_{\theta} E_{x \sim p_x, y \sim p(y|x^{l_i})} log \frac{p(y|x^{l_i})}{q_{\theta}(y|x)} \tag{4}$$

p_x represents the distribution of the prompt, $p(y|x^{l_i})$ represents the teacher's output distribution given prompt x at purity level i, and $q_{\theta}(y|x)$ represents the student's output distribution given prompt x.

4 Experimental Setup

Dataset: We consider Spider (Yu et al., 2018b), a publicly accessible and widely used benchmark for Text2Sql tasks. The Spider dataset is a challenging dataset across domains, where the validation set and the training set use different databases. Its training set contains 8,569 entries, involving 146 databases. The development set includes 1,034 entries across 20 databases, while the test set comprises 2,147 entries from 34 databases. In total, the dataset encompasses 10,181 questions paired with 5,693 unique and complex SQL queries. It was annotated by 11 college students over 1,000 person-hours, with the databases sourced from university courses, SQL tutorial websites, online CSV files, and WikiSQL (Zhong et al., 2017). The SQL queries in the dataset are categorized into four difficulty levels—easy, medium, hard, and extra hard—based on the number and complexity of SQL components and conditions. Since the test set is reserved, We trained the model on the Spider train dataset and evaluated the model on the Spider dev dataset.

Model: The Llama 3.2 series, released by Meta, is the latest in the Llama (Dubey et al., 2024) lineup. We used the pre-trained Llama 3.2 models with 1B and 3B parameters.

Metrics: We use two commonly employed evaluation metrics: exact-set match accuracy (EM) and test-suite accuracy (TS) (Zhong et al., 2020). EM treats SQL statements as sets of components, such as SELECT, WHERE, and GROUP BY clauses, and evaluates whether each component of the generated SQL matches the corresponding component in the gold standard SQL. TS compares the results of the predicted SQL with the gold standard SQL using a test suite. A test suite is a collection of databases that can effectively distinguish between the gold standard SQL and semantically similar but different SQL queries. Compared to evaluating correctness based on a single execution result, TS reduces false positives, as different questions may have SQL queries that produce the same result but differ in semantics.

Implementation: The experiments were conducted on an NVIDIA A800 80G GPU, using the AdamW optimizer with a learning rate of 1e-5. The distillation coefficient was set to 0.8, and greedy decoding was employed as the generation strategy.

3B\L4 dev	TS	EM	prompt_len
L4_train	0.602	0.612	1
L3_8 train	0.643	0.661	0.733
L3_4 train	**0.684**	**0.698**	**0.533**
L3_2 train	0.663	0.678	0.466
L3_1 train	0.489	0.439	0.386
L2_train	0.142	0.103	0.333
L1_train	0.057	0.031	0.266

Table 1: The results of the fine-tuned 3B model on the L4 dev test. The best results are **boldfaced**. prompt_len is a normalized length that represents the ratio of the length of the prompt used for fine-tuning to the length of the prompt that includes all database information.

3B	L2 dev		L1 dev	
	TS	EM	TS	EM
L4_train	0.625	0.646	0.670	0.686
L2_train	0.715	0.760	0.729	0.783
L1_train	0.567	0.600	0.733	0.789
L3_4 train	**0.719**	**0.742**	**0.735**	**0.754**

Table 2: The results of the fine-tuned 3B model on the L2 dev and L1 dev tests. The best results are **boldfaced**.

5 Result

We denote the training set with a level i prompt as Li train, and the Spider dev dataset with a level i prompt as Li dev. We used four types of L3 prompts: L3_1, L3_2, L3_4, and L3_8, where L3_i represents adding i irrelevant tables in addition to the relevant ones. L1 train has the highest purity, as its prompt includes only the tables and columns relevant to the question. L4 train has the lowest purity, with its prompt encompassing all tables in the database. We will use the model trained on L4 train as our **baseline** for comparison.

5.1 Pure Fine-tuning

For the 3B model: As shown in Table 1, our fine-tuned baseline achieves TS of 60.2% and EM of 61.2% on the L4 dev. As the purity of the training set increases, the performance of the fine-tuned model on the L4 dev initially improves, then declines. The model fine-tuned using L3_4 train reaches its peak performance, with an improvement of 8.2% in TS and 8.6% in EM compared to the baseline. When fine-tuning with L1 train, the model's performance on the L4 dev is the poorest, with TS of just 5.1% and EM of only 3.1%. Upon

1B\L4 dev	TS	EM
L4_train	**0.577**	**0.600**
L3_8 train	0.557	0.565
L3_4 train	0.569	0.596
L3_2 train	0.557	0.568
L3_1 train	0.319	0.288
L2_train	0.080	0.067
L1_train	0.049	0.043

Table 3: The results of the fine-tuned 1B model on the L4 dev test. The best results are **boldfaced**.

1B	L2 dev		L1 dev	
	TS	EM	TS	EM
L4_train	0.598	0.630	0.603	0.636
L3_4 train	**0.625**	**0.658**	**0.640**	**0.671**

Table 4: The results of the fine-tuned 1B model on the L2 dev and L1 dev tests. The best results are **boldfaced**.

analyzing the generated results, we find that the improvement is due to the model's increased ability to focus on the relevant tables and columns in the prompts as their purity increases, which reduces the generation of incorrect tables and columns and database values. The performance decline occurs because, when only relevant tables are included in the training set, the model assumes all tables in the prompt are relevant. When the input includes prompts with irrelevant tables, the model's performance drops significantly. As shown in Table 2, the model fine-tuned with L3_4 train demonstrates notable improvements on the L1 dev, L2 dev, and L4 dev compared to the baseline, with at least a 6% increase in both TS and EM metrics due to its enhanced focusing ability. However, due to its inability to distinguish irrelevant tables, the model fine-tuned with L1 train experiences a decrease in performance when tested on the L2 dev and L4 dev. On the L1 dev, the fine-tuned model's TS decreases by 16.6% and EM by 18.9%. These experiments support the aforementioned observations.

For the 1B model: Due to having fewer parameters, the 1B model performs worse than the 3B model across various settings. As shown in Table 3, our fine-tuned baseline achieves TS of 57.7% and EM of 60.0% on the L4 dev. The performance pattern observed with the 1B model on the L4 dev mirrors that of the 3B model but does not surpass the baseline performance. The model fine-tuned

1B\L4 dev	TS	EM
L4 train	0.577	0.600
Pure-KD	**0.596**	**0.602**

Table 5: The test results of the 1B model on L4 dev before and after distillation. The best results are **boldfaced**.

with L3_4 train achieves the highest performance, with TS 0.8% lower than the baseline and EM 0.4% lower, making it comparable to the baseline. However, as shown in Table 4, when tested on the L1 dev and L2 dev, the model fine-tuned with L3_4 shows improvements of at least 2.7% in both TS and EM. We believe that although the 1B model fine-tuned with L3_4 is better at focusing on important information, it lacks the foundational capabilities of the 3B model to manage the excessive irrelevant information in the L4 dev, and therefore fails to improve upon the baseline.

As shown in Table 1, 2, 3, and 4, the model performs better when the informational purity of the prompts used during testing is higher. For the same model tested with prompts of varying purities, the difference between TS and EM reached 5%. The 3B model fine-tuned with L3_4 achieved comparable results on the L2 dev and L1 dev tests to those of models fine-tuned with L2 train and L1 train, respectively. This suggests that our fine-tuned model has reached its performance limit.

5.2 Pure Knowledge Distillation

To leverage the excellent performance of the fine-tuned model on L1 level prompts, we use the 3B model trained with L1 train as the teacher model and the 1B model trained with L4 train as the student model. During distillation, the teacher model uses L1 train for inference, while the student model is trained on L4 train. The distillation results are shown in Table 5. After distillation, the student model's TS improves by 1.9%.

We compare our results with state-of-the-art Text2Sql methods. Following the approach of SQL-PaLM, we select the top-performing methods from the Spider leaderboard for comparison. For fine-tuning methods, we only choose those with similar model sizes. As shown in Table 6, our method outperforms GPT-4 with few-shot learning. Although our experimental results are lower than those of advanced contextual learning combined with GPT-4, our method has the advantages of lower inference

1B	Model	TS
fine-tune	RASAT+PICARD	0.703
	RESDSQL-3B+ NatSQL	**0.735**
few-shot	GPT-4 (Few-shot)	0.674
	SQL-PaLM	0.724
	DIN-SQL (w/ GPT-4)	**0.742**
ours	L3_4 train	0.684
ours(L1)	L3_4 train	**0.735**

Table 6: Evaluation on SPIDER dev set with top-ranked methods. The results for SQL-PaLM's few-shot were obtained without using consistent decoding. Both the best fine-tuning result and the best few-shot result are **boldfaced**.

costs and zero-shot capabilities. Our experimental results are not as good as those of advanced fine-tuning methods, which may be due to the fact that we did not incorporate the constrained decoding module Picard or utilize the content within the database. As mentioned earlier, our fine-tuned model seems to have reached its upper limit, performing comparably to models fine-tuned with L1 train and L2 train on the L1 dev and L2 dev, respectively. When provided with L1 level prompts, our model can achieve performance comparable to the state-of-the-art (SOTA) among models of the same level.

The lengths of the prompts we used are shown in Table 1. Reducing redundant database information in the prompts significantly decreased their length. The prompt length for L3_4 train is 47% shorter than that for L4 train. In Pure-KD, the length of the teacher's prompt is only 26.6% of the student's.

6 Discussion

We have observed that prompts in previous Text2Sql fine-tuning tasks typically include all tables related to the question in the database. We recommend the pure fine-tuning strategy to reduce redundant information. When the prompt length is only 53% of the baseline, the 3B model trained on L3_4 train data outperforms the baseline by over 5.5% in both TS and EM metrics across Spider dev sets with three different prompt purity levels (L1, L2, and L4), achieving up to a 9% improvement. To leverage the model with pure prompts, we propose the pure distillation strategy, which further enhances the model's performance. Our fine-tuned model outperforms GPT-4 with few-shot learning. When tested with the most accurate prompt, the fine-tuned model's TS and EM metrics approach state-of-the-art (SOTA) levels.

Our experiments have some limitations. We did not take into account the content of the database or intermediate representations in our current setup, and incorporating these elements could enhance the fine-tuned model's capabilities. Additionally, we only conducted experiments on the Spider dataset, so incorporating more data should further improve the model's performance. Now that the Spider2 dataset has been released, more research opportunities should be explored.

7 Acknowledgments

This work was supported by the National Key R&D Program of China (2022YFB2404200), the Beijing Natural Science Foundation (No. 4252022).

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*.

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*.

Justin Chih-Yao Chen, Swarnadeep Saha, Elias Stengel-Eskin, and Mohit Bansal. 2024. Magdi: Structured distillation of multi-agent interaction graphs improves reasoning in smaller language models. *arXiv preprint arXiv:2402.01620*.

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

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

Yujian Gan, Xinyun Chen, Jinxia Xie, Matthew Purver, John R Woodward, John Drake, and Qiaofu Zhang. 2021. Natural sql: Making sql easier to infer from natural language specifications. *arXiv preprint arXiv:2109.05153*.

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*.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. 2023. Knowledge distillation of large language models. *arXiv preprint arXiv:2306.08543*.

Jiaqi Guo, Zecheng Zhan, Yan Gao, Yan Xiao, Jian-Guang Lou, Ting Liu, and Dongmei Zhang. 2019. Towards complex text-to-sql in cross-domain database with intermediate representation. *arXiv preprint arXiv:1905.08205*.

Jonathan Herzig, Peter Shaw, Ming-Wei Chang, Kelvin Guu, Panupong Pasupat, and Yuan Zhang. 2021. Unlocking compositional generalization in pre-trained models using intermediate representations. *arXiv preprint arXiv:2104.07478*.

Geoffrey Hinton. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2019. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*.

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*, 11, pages 13067–13075.

Liunian Harold Li, Jack Hessel, Youngjae Yu, Xiang Ren, Kai-Wei Chang, and Yejin Choi. 2023b. Symbolic chain-of-thought distillation: Small models can also" think" step-by-step. *arXiv preprint arXiv:2306.14050*.

Chen Liang, Simiao Zuo, Qingru Zhang, Pengcheng He, Weizhu Chen, and Tuo Zhao. 2023. Less is more: Task-aware layer-wise distillation for language model compression. In *International Conference on Machine Learning*, pages 20852–20867. PMLR.

Kevin J Liang, Weituo Hao, Dinghan Shen, Yufan Zhou, Weizhu Chen, Changyou Chen, and Lawrence Carin. 2020. Mixkd: Towards efficient distillation of large-scale language models. *arXiv preprint arXiv:2011.00593*.

Xi Victoria Lin, Richard Socher, and Caiming Xiong. 2020. Bridging textual and tabular data for cross-domain text-to-sql semantic parsing. *arXiv preprint arXiv:2012.12627*.

Qin Lyu, Kaushik Chakrabarti, Shobhit Hathi, Souvik Kundu, Jianwen Zhang, and Zheng Chen. 2020. Hybrid ranking network for text-to-sql. *arXiv preprint arXiv:2008.04759*.

Mohammadreza Pourreza and Davood Rafiei. 2024. Din-sql: Decomposed in-context learning of text-to-sql with self-correction. *Advances in Neural Information Processing Systems*, 36.

Jiexing Qi, Jingyao Tang, Ziwei He, Xiangpeng 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. *arXiv preprint arXiv:2205.06983*.

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*, 21(140):1–67.

Daking Rai, Bailin Wang, Yilun Zhou, and Ziyu Yao. 2023. Improving generalization in language model-based text-to-sql semantic parsing: Two simple semantic boundary-based techniques. *arXiv preprint arXiv:2305.17378*.

Andrei A Rusu, Sergio Gomez Colmenarejo, Caglar Gulcehre, Guillaume Desjardins, James Kirkpatrick, Razvan Pascanu, Volodymyr Mnih, Koray Kavukcuoglu, and Raia Hadsell. 2015. Policy distillation. *arXiv preprint arXiv:1511.06295*.

V Sanh. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Torsten Scholak, Nathan Schucher, and Dzmitry Bahdanau. 2021. Picard: Parsing incrementally for constrained auto-regressive decoding from language models. *arXiv preprint arXiv:2109.05093*.

Kaitao Song, Hao Sun, Xu Tan, Tao Qin, Jianfeng Lu, Hongzhi Liu, and Tie-Yan Liu. 2020. Lightpaff: A two-stage distillation framework for pre-training and fine-tuning. *arXiv preprint arXiv:2004.12817*.

Ruoxi Sun, Sercan Ö Arik, Alex Muzio, Lesly Miculicich, Satya Gundabathula, Pengcheng Yin, Hanjun Dai, Hootan Nakhost, Rajarishi Sinha, Zifeng Wang, et al. 2023. Sql-palm: Improved large language model adaptation for text-to-sql (extended). *arXiv preprint arXiv:2306.00739*.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for bert model compression. *arXiv preprint arXiv:1908.09355*.

Wenhui Wang, Hangbo Bao, Shaohan Huang, Li Dong, and Furu Wei. 2020a. Minilmv2: Multi-head self-attention relation distillation for compressing pretrained transformers. *arXiv preprint arXiv:2012.15828*.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020b. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in Neural Information Processing Systems*, 33:5776–5788.

Tao Yu, Michihiro Yasunaga, Kai Yang, Rui Zhang, Dongxu Wang, Zifan Li, and Dragomir Radev. 2018a. Syntaxsqlnet: Syntax tree networks for complex and cross-domaintext-to-sql task. *arXiv preprint arXiv:1810.05237*.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, et al. 2018b. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. *arXiv preprint arXiv:1809.08887*.

Rongzhi Zhang, Jiaming Shen, Tianqi Liu, Jialu Liu, Michael Bendersky, Marc Najork, and Chao Zhang. 2023. Do not blindly imitate the teacher: Using perturbed loss for knowledge distillation. *arXiv preprint arXiv:2305.05010*.

Ruiqi Zhong, Tao Yu, and Dan Klein. 2020. Semantic evaluation for text-to-sql with distilled test suites. *arXiv preprint arXiv:2010.02840*.

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