A comprehensive evaluation of ChatGPT's zero-shot Text-to-SQL capability

Aiwei Liu[1], Xuming Hu[1], Lijie Wen[1], Philip S. Yu[1,2]
[1]Tsinghua University
[2]University of Illinois at Chicago
[1]{liuaw20, hxm19}@mails.tsinghua.edu.cn
[1]wenlj@tsinghua.edu.cn [2]psyu@uic.edu

Abstract

This paper presents the first comprehensive analysis of ChatGPT's Text-to-SQL ability. Given the recent emergence of large-scale conversational language model ChatGPT and its impressive capabilities in both conversational abilities and code generation, we sought to evaluate its Text-to-SQL performance. We conducted experiments on 12 benchmark datasets with different languages, settings, or scenarios, and the results demonstrate that ChatGPT has strong text-to-SQL abilities. Although there is still a gap from the current state-of-the-art (SOTA) model performance, considering that the experiment was conducted in a zero-shot scenario, ChatGPT's performance is still impressive. Notably, in the ADVETA (RPL) scenario, the zero-shot ChatGPT even outperforms the SOTA model that requires fine-tuning on the Spider dataset by 4.1%, demonstrating its potential for use in practical applications. To support further research in related fields, we have made the data generated by ChatGPT publicly available at https://github.com/THU-BPM/chatgpt-sql.

1 Introduction

With the increasing attention given to large-scale language models, they have become an essential component in natural language processing. As the size of pre-trained models grows, their usage is also gradually changing. Different from models such as BERT (Devlin et al., 2018) and T5 (Raffel et al., 2020), which require fine-tuning with a small amount of data, models such as GPT-3 (Brown et al., 2020), require the prompt design to generate target outputs. The recent ChatGPT[1] model, which employs Reinforcement Learning for Human Feedback (RLHF) (Christiano et al., 2017), simplifies prompt design, enabling better utilization of the zero-shot ability of large-scale pre-trained models in a conversational way. Based on this, many works

have begun to analyze the zero-shot ability of ChatGPT in various natural language processing tasks, such as information extraction (Wei et al., 2023), text summarization (Wang et al., 2023), and mathematical abilities (Frieder et al., 2023). Due to ChatGPT's strong ability in code generation and the fact that code generation models usually require a large amount of annotated data to produce good results, a zero-shot code generation model is very important. This paper first conducts a comprehensive evaluation of ChatGPT's zero-shot performance on a challenging code generation task: Text-to-SQL.

The Text-to-SQL task involves converting user input text into SQL statements that can be executed on a database, allowing non-expert users to better access the contents of a database. The design of Text-to-SQL models is typically challenging because they need to work across different databases and consider various user text input text and database structures. Due to the complexity of the Text-to-SQL task, a comprehensive evaluation of its performance requires consideration of a variety of scenarios in addition to the classic Spider dataset (Yu et al., 2018). For example, Spider-SYN (Gan et al., 2021a) focuses on scenarios where the data schema mentioned in the user text input is synonymous with the database schema, Spider-DK (Gan et al., 2021b) considers scenarios where the input question contains additional knowledge, Spider-CG (Gan et al., 2022) emphasizes the combination generalization ability of models, and ADVETA (Pi et al., 2022) considers scenarios where column names in the database have been modified. Additionally, to better reflect real-world scenarios, SParC(Yu et al., 2019b) and CoSQL (Yu et al., 2019a) incorporate multi-turn interaction between the user and the system. Finally, to evaluate models' multilingual capabilities, CSpider (Min et al., 2019) and DuSQL (Wang et al., 2020) evaluate Text-to-SQL performance in Chinese.

During our experiments, we evaluate the ability

---

[1]https://chat.openai.com/

of ChatGPT on 12 different Text-to-SQL benchmark datasets. Based on the experimental results, we conclude the following observations:

1. Compared to the current state-of-the-art (SOTA) model that uses complete training data, ChatGPT without using task-specific training data only performs 14% worse. This already demonstrates that ChatGPT is a strong zero-shot Text-to-SQL converter.

2. The robustness of ChatGPT in generating SQL statements is very strong, and the performance gap between ChatGPT and the SOTA models is only 7.8% on some robustness settings of the Spider dataset, which is lower than the 14% gap on the standard Spider dataset.

3. In the ADVETA (Pi et al., 2022) scenario where the column names in the database are adversarially modified, ChatGPT's performance even surpasses that of the current SOTA models by 4.1%.

4. The Exact Match metric of the data generated by ChatGPT is very low because there are many different ways to express SQLs with the same purpose. Therefore, we mainly use execution accuracy as the evaluation metric.

Overall, our experiments demonstrate that ChatGPT has strong Text-to-SQL capabilities and robustness, and it outperforms SOTA models in certain scenarios.

2 Method

In order to enable ChatGPT to generate accurate SQL outputs, we utilized the prompt as shown in Figure 1. To ensure a fair demonstration of ChatGPT's Text-to-SQL capabilities, we directly adopted the Text-to-SQL prompt used in the OpenAI demo webwite[2] without conducting further prompt exploration.

The upper half of Figure 1 represents the prompt in a single-turn Text-to-SQL scenario, where only the database and question information is required in the prompt. Meanwhile, in order to facilitate further evaluations, we emphasize in the prompt that the generated SQL statements can be executed in an SQLite database. The lower half of Figure 1 represents the prompt in a multi-turn Text-to-SQL
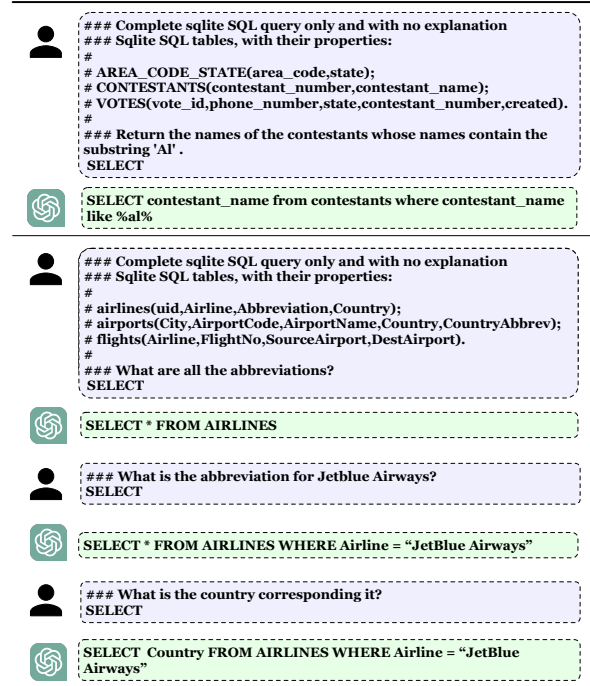

Figure 1: Example prompts for Text-to-SQL using ChatGPT. The prompt at the top is for a single-turn scenario, while the one below is for multi-turn scenarios where only new questions are added in each interaction.

scenario, where the prompt for the first interaction is the same as that in the single-turn scenario, and for subsequent interactions, only the new questions are required.

3 Experiment

3.1 Experiment Setup

Datasets. We conduct extensive experiments on twelve public benchmark datasets as follows: (1) **Spider** (Yu et al., 2018) is a large-scale cross-domain Text-to-SQL benchmark. It contains 8659 training samples across 146 databases and 1034 evaluation samples across 20 databases. (2) **Spider-SYN** (Gan et al., 2021a) is a challenging variant of the Spider evaluation dataset. Spider-SYN is constructed by manually modifying natural language questions with synonym substitutions. (3) **Spider-DK** (Gan et al., 2021b) is a human-curated dataset based on Spider, which samples 535 question-SQL pairs across 10 databases from the Spider development set and modifies them to incorporate the domain knowledge. (4) **Spider-Realistic** (Deng et al., 2020) is a new evaluation set based on the Spider dev set with explicit mentions of column names removed, which contains 508 samples. (5) **Spider-CG(SUB)**

Methods / Datasets	SPIDER			SPIDER-SYN			SPIDER-REALISTIC		
	VA	EX	TS	VA	EX	TS	VA	EX	TS
T5-3B + PICARD	98.4	79.3	69.4	98.2	69.8	61.8	97.1	71.4	61.7
RASAT + PICARD	98.8	80.5	70.3	98.3	70.7	62.4	97.4	71.9	62.6
RESDSQL-3B + NatSQL	99.1	84.1	73.5	98.8	76.9	66.8	98.4	81.9	70.1
ChatGPT	97.7	70.1(14↓)	60.1	96.2	58.6(18.3↓)	48.5	96.8	63.4(18.5 ↓)	49.2

Table 1: Comparison of the performance of ChatGPT and other models on Spider, Spider-SYN, and Spider-Realistic datasets.

Methods / Datasets	SPIDER-DK			ADVETA(RPL)			ADVETA(ADD)		
	VA	EX	TS	VA	EX	TS	VA	EX	TS
T5-3B + PICARD	97.8	62.5	-	92.7	50.6	-	97.2	69.4	-
RASAT + PICARD	98.5	63.9	-	92.9	51.5	-	97.4	70.7	-
RESDSQL-3B + NatSQL	98.8	66.0	-	93.9	54.4	-	97.9	71.9	-
ChatGPT	96.4	62.6(3.4 ↓)	-	91.4	58.5(**4.1** ↑)	-	93.1	68.1(3.8 ↓)	-

Table 2: Performance of different methods on the Spider-DK, ADVETA(RPL) and ADVETA(ADD) benchmark datasets.

and **Spider-CG(APP)** (Gan et al., 2022) are two evaluation datasets to measure the compositional generalization of models, which is constructed by sub-sentence substitution between different examples and appending a sub-sentence into another sentence separately. (6) **ADVETA(rpl)** and **ADVETA(add)** (Pi et al., 2022) are two challenging test datasets for the Spider dataset which are composed of adversarial replacements of column names and the addition of new column names, respectively. (7) **CSpider** (Min et al., 2019) dataset is constructed by translating Spider into Chinese, which is the same size as the origin Spider dataset (8) **DuSQL** (Wang et al., 2020) is a larger scale Chinese Text-to-SQL dataset with 23,797 question/SQL pairs. (9) **SParC** (Yu et al., 2019b) and **CoSQL** (Yu et al., 2019a) are two multi-turn Text-to-SQL dataset with 1625 and 1007 questions in the dev set separately.

Evaluation Metrics. We mainly adopt three evaluation metrics which are valid SQL (VA), execution accuracy(EX), and test-suite accuracy (TS). Valid SQL (VA) is the proportion of SQL statements that can be executed successfully. Execution accuracy (EX) is the proportion of data where the execution results match the standard SQL statements. Test-suite accuracy (TS) (Zhong et al., 2020) could achieve high code coverage from a distilled test suite of the database, which is also based on execution. Note that we do not

use the main-stream exact match accuracy, as SQL queries that achieve the same goal can often be expressed in different ways, making it difficult for zero-shot ChatGPT models to achieve high exact match accuracy.

Baselines. Due to our exclusive reliance on execution-based evaluation, we did not employ baselines such as RatSQL (Wang et al., 2019) and LGESQL (Cao et al., 2021), which generate only SQL skeletons without generating values. Instead, we primarily utilized three baselines: (1) PICARD (Scholak et al., 2021) is a method for constraining auto-regressive decoders of language models through incremental parsing. (2) RASAT (Qi et al., 2022) introduces relation-aware self-attention into transformer models and also utilizes constrained auto-regressive decoders. (3) RESD-SQL (Li et al., 2023) proposes a ranking-enhanced encoding and skeleton-aware decoding framework to decouple the schema linking and the skeleton parsing. Among those, PICARD and RASAT are based on T5-3B (Raffel et al., 2020) model.

3.2 Main Experiment

Evaluation on Spider Dataset. In Table 1, we present a comparison between ChatGPT and the current state-of-the-art (SOTA) models. Overall, ChatGPT exhibits a strong Text-to-SQL ability.Despite the 14% gap in execution accuracy compared to the current SOTA models and a 13.4%

Methods / Datasets	SPIDER-CG(SUB)			SPIDER-CG(APP)		
	VA	EX	TS	VA	EX	TS
T5-3B + PICARD	98.4	82.1	74.3	95.8	68.0	60.5
RASAT + PICARD	99.0	82.6	76.1	96.2	68.6	61.0
RESDSQL-3B + NatSQL	99.4	83.3	77.5	96.4	69.4	62.4
ChatGPT	98.3	76.6(6.7 ↓)	67.2	91.2	61.3(8.1 ↓)	47.9

Table 3: Performance of different methods on the Spider-CG(SUB) and Spider-CG(APP) benchmark datasets.

gap in test suite accuracy, it is remarkable that ChatGPT achieved such results in a zero-shot scenario considering that it was not fine-tuned on the Spider training set.

Evaluation on Spider-SYN and Spider-Realistic Datasets. Table 1 also includes a comparison of ChatGPT's performance on the Spider-SYN and Spider-Realistic datasets. The main difference between these datasets and the Spider dev set is that they eliminate the explicit appearance of the database schema in the questions. Overall, although ChatGPT still performs well on these two settings, the performance gap between ChatGPT and the original SOTA models becomes slightly larger than that on the Spider dataset. This suggests that the current models have already achieved sufficient robustness in these two scenarios.

Evaluation on Spider-DK and ADVETA Datasets. In Table 2, we further compare and analyze ChatGPT's performance on Spider-DK, ADVETA (RPL), and ADVETA (ADD). We find that ChatGPT performs exceptionally well on these datasets, with very small performance gaps compared to the current SOTA models. In fact, ChatGPT outperforms all current SOTA models on ADVETA (RPL). For the Spider-DK dataset, we speculate that ChatGPT's excellent performance is due to its additional knowledge provided by the large-scale pretraining. As for scenarios such as ADVETA, where the dataset's column names undergo adversarial modifications, the poor generalization performance of current models may be due to the significant distribution difference from the original dataset. Overall, ChatGPT exhibits strong robustness in scenarios that require additional knowledge or adversarial modifications are applied to the database column names.

Evaluation on Spider-CG Dataset. In Table

Methods / Datasets	SPARC		COSQL	
	VA	EX	VA	EX
T5-3B + PICARD	-	-	97.5	64.7
RASAT + PICARD	98.4	74.0	97.8	66.3
ChatGPT	97.3	63.1	95.8	60.7

Table 4: The performance of ChatGPT on two multi-turn Text-to-SQL datasets: SParC and CoSQL.

Methods / Datasets	CSPIDER		DUSQL	
	VA	EX	VA	EX
ChatGPT	96.0	65.1	82.7	53.7

Table 5: The performance of ChatGPT on two Chinese Text-to-SQL datasets: CSpider and DuSQL.

3, we further analyze ChatGPT's ability in the compositional generalization scenario. We found that in Spider-CG (SUB), SQL substructures are replaced to form combinations that do not exist in the training set. In this scenario, ChatGPT even outperforms the original Spider dev set. Even on the more challenging Spider-CG (APP) dataset, ChatGPT achieves strong performance, and the performance gap with SOTA models is relatively smaller than that on the original Spider dataset. Overall, since ChatGPT is a zero-shot model, it is not as affected by compositional generalization as the SOTA models. Overall, zero-shot models have greater advantages in the compositional generalization setting.

Evaluation on multi-turn Text-to-SQL scenarios. Given ChatGPT's strong contextual modeling ability, we further evaluate its performance on multi-turn Text-to-SQL scenarios: SPARC and COSQL. As shown in Table 4, ChatGPT exhibits strong multi-turn Text-to-SQL ability. Although there is still a gap compared to the current SOTA models, the gap is relatively smaller compared to the single-turn Spider dataset. Meanwhile, Chat-

GPT also performs better on CoSQL datasets with more average interactions, which also indicates that ChatGPT's strong contextual modeling ability is very helpful for multi-turn Text-to-SQL.

Evaluation on Chinese Text-to-SQL scenarios.
We further evaluate ChatGPT's Text-to-SQL ability on other languages in Table 5. The experiments are mainly conducted on two datasets, CSpider and DuSQL, where only the questions are in Chinese for CSpider and both the schema names and questions are in Chinese for DuSQL. The results show that while ChatGPT performs well in the Chinese Text-to-SQL scenario, there is still a performance gap compared to the English Text-to-SQL scenario. Moreover, the performance is even worse when the table names and column names are also in Chinese, with a large number of generated SQL queries being non-executable and a lower execution accuracy. This suggests the cross-lingual generalization ability of ChatGPT requires further improvement.

3.3 Case Study

In Table 6, we present four typical prediction errors made by ChatGPT on the Spider dev dataset. The first error case shows that ChatGPT tends to design JOIN statements more finely by using LEFT JOIN, but this level of granularity is not present in the original Spider dev dataset. The second error case arises from ChatGPT's confusion regarding the database structure, and it is not clear which column the term "full name" specifically refers to. The third example's error was due to the generated SQL statement lacking correct semantic interpretation, resulting in incorrect output for the "where" clauses with nested SQL statements. The fourth case of error is due to errors in copying specific values, where the case sensitivity of the original value was not preserved when regenerating the value.

In summary, ChatGPT's errors mostly occur in small details, and some of these issues can be addressed and improved in later stages of development, such as in the first, third, and fourth cases. However, for errors like the second case, which indicate a lack of understanding of the database schema, further improvements to the model's ability may be necessary to resolve them.

4 Related Work

Text-to-SQL is an important semantic parsing task that converts natural language questions posed by

Question: Show the stadium name and the number of concerts in each stadium.
ChatGPT: SELECT stadium.Name, COUNT(concert.concert_ID) FROM stadium **LEFT JOIN** concert ON stadium.Stadium_ID = concert.Stadium_ID GROUP BY stadium.Name;
Gold: SELECT T2.name , count(*) FROM concert AS T1 **JOIN** stadium AS T2 ON T1.stadium_id = T2.stadium_id GROUP BY T1.stadium_id
Question: How many car models were produced by the maker with full name American Motor Company?
ChatGPT: SELECT COUNT(*) FROM model_list WHERE **Maker** = 'American Motor Company'
Gold: SELECT count(*) FROM CAR_MAKERS AS T1 JOIN model_list AS T2 ON T1.Id = T2.Maker WHERE **T1.FullName** = 'American Motor Company';
Question: How many cars have a larger accelerate than the car with the largest horsepower?
ChatGPT: SELECT COUNT(*) FROM cars_data WHERE Accelerate > (SELECT **MAX(Horsepower)** FROM cars_data)
Gold: SELECT COUNT(*) FROM CARS_DATA WHERE Accelerate > (SELECT **Accelerate** FROM CARS_DATA ORDER BY Horsepower DESC LIMIT 1);
Question: What is the abbreviation of Airline "JetBlue Airways"?
ChatGPT: SELECT Abbreviation FROM airlines WHERE Airline = **'Jetblue Airways'** ;
Gold: SELECT Abbreviation FROM AIRLINES WHERE Airline = **"JetBlue Airways"**;

Table 6: Case study: We selected four cases of incorrect predictions generated by ChatGPT on the Spider development set for analysis.

users into SQL statements that can be executed on a database. On the classic Spider dataset (Yu et al., 2018), many classic works such as RatSQL (Wang et al., 2019) and LGESQL (Cao et al., 2021) have achieved excellent results. Since Text-to-SQL is a very complex task involving both user input questions and database structure, the robustness of the model is crucial. To further explore this issue, Gan et al. (2021a) proposed the Spider-SYN dataset to evaluate the robustness of models under synonym substitution scenarios. Some works, such as Proton (Wang et al., 2022) and ISESL-SQL (Liu et al., 2022), are also devoted to improving the robustness of models in this scenario. Meanwhile, many works explore the robustness of the Text-to-SQL task in other scenarios. The Spider-DK dataset (Gan et al., 2021b) evaluates the robustness of models in scenarios requiring additional knowl-

edge. The Spider-Realistic dataset (Deng et al., 2020) removes the explicit appearance of dataset schema information in user questions, thereby increasing the difficulty of the original task. The Spider-CG dataset (Gan et al., 2022) evaluates the robustness of models in compositional generalization scenarios. The ADVETA dataset (Pi et al., 2022) evaluates the robustness of models in scenarios involving adversarial modifications of database table information. In addition, to verify the robustness of models in cross-lingual scenarios, CSpider (Min et al., 2019) and DuSQL (Wang et al., 2020) have been proposed to evaluate the robustness of models in the Chinese language. To evaluate the performance of Text-to-SQL in more realistic scenarios, SParC (Yu et al., 2019b) and CoSQL (Yu et al., 2019a) have been proposed to evaluate the performance of multi-turn Text-to-SQL. Models such as STAR (Cai et al., 2022) and CQR-SQL (Xiao et al., 2022) have also achieved good results in this scenario.

Currently, several methods have been attempted to explore the improvement of large-scale language models for Text-to-SQL models. The PICARD (Scholak et al., 2021) and RASAT (Qi et al., 2022) utilize the T5-3B model, but still require the training data for fine-tuning. Rajkumar et al. (2022) investigated the Text-to-SQL capabilities of the GPT3 model in a zero-shot setting. Cheng et al. (2022) proposed the BINDER model based on the GPT3 codex, which has similar Text-to-SQL generation capabilities with the need for in-context exemplar annotations. However, these works do not provide a comprehensive evaluation of Text-to-SQL and are limited to a few datasets without other robustness settings. In this work, we are the first to evaluate the comprehensive Text-to-SQL capabilities of ChatGPT.

5 Conclusion

In this work, we conducted a comprehensive analysis of ChatGPT's zero-shot ability in Text-to-SQL. We found that even without using any training data, ChatGPT still has strong Text-to-SQL ability, although there is still some gap compared to the current SOTA models. Additionally, ChatGPT demonstrated strong robustness, performing relatively better on most robustness benchmarks and even surpassing the current SOTA models on the ADVETA benchmark. Although this paper has made some findings, we only utilize a common prompt to evaluate ChatGPT's ability. And in future work, better prompts could be designed to explore ChatGPT's Text-to-SQL ability.

6 Future work

In future work, we will primarily consider the following two directions to further explore ChatGPT's capabilities in the Text-to-SQL task. Firstly, we will conduct more interactions with ChatGPT to address the issue of generating non-executable SQL statements. We can design ChatGPT to engage in multi-turn dialogues with the provided database error messages to further ensure the validity of generated SQL statements. Secondly, we will add more highly correlated in-context examples to the prompt to enhance ChatGPT's ability to generate Text-to-SQL.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Zefeng Cai, Xiangyu Li, Binyuan Hui, Min Yang, Bowen Li, Binhua Li, Zheng Cao, Weijie Li, Fei Huang, Luo Si, et al. 2022. Star: Sql guided pre-training for context-dependent text-to-sql parsing. *arXiv preprint arXiv:2210.11888*.

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. *arXiv preprint arXiv:2106.01093*.

Zhoujun Cheng, Tianbao Xie, Peng Shi, Chengzu Li, Rahul Nadkarni, Yushi Hu, Caiming Xiong, Dragomir Radev, Mari Ostendorf, Luke Zettlemoyer, et al. 2022. Binding language models in symbolic languages. *arXiv preprint arXiv:2210.02875*.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30.

Xiang Deng, Ahmed Hassan Awadallah, Christopher Meek, Oleksandr Polozov, Huan Sun, and Matthew Richardson. 2020. Structure-grounded pretraining for text-to-sql. *arXiv preprint arXiv:2010.12773*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Simon Frieder, Luca Pinchetti, Ryan-Rhys Griffiths, Tommaso Salvatori, Thomas Lukasiewicz, Philipp Christian Petersen, Alexis Chevalier, and Julius Berner. 2023. Mathematical capabilities of chatgpt. *arXiv preprint arXiv:2301.13867*.

Yujian Gan, Xinyun Chen, Qiuping Huang, and Matthew Purver. 2022. Measuring and improving compositional generalization in text-to-sql via component alignment.

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 *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2505–2515.

Yujian Gan, Xinyun Chen, and Matthew Purver. 2021b. Exploring underexplored limitations of cross-domain text-to-sql generalization. *arXiv preprint arXiv:2109.05157*.

Haoyang Li, Jing Zhang, Cuiping Li, and Hong Chen. 2023. Decoupling the skeleton parsing and schema linking for text-to-sql. *arXiv preprint arXiv:2302.05965*.

Aiwei Liu, Xuming Hu, Li Lin, and Lijie Wen. 2022. Semantic enhanced text-to-sql parsing via iteratively learning schema linking graph. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1021–1030.

Qingkai Min, Yuefeng Shi, and Yue Zhang. 2019. A pilot study for chinese sql semantic parsing. *arXiv preprint arXiv:1909.13293*.

Xinyu Pi, Bing Wang, Yan Gao, Jiaqi Guo, Zhoujun Li, and Jian-Guang Lou. 2022. Towards robustness of text-to-SQL models against natural and realistic adversarial table perturbation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2007–2022, Dublin, Ireland. Association for Computational Linguistics.

Jiexing Qi, Jingyao Tang, Ziwei He, Xiangpeng Wan, 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. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Nitarshan Rajkumar, Raymond Li, and Dzmitry Bahdanau. 2022. Evaluating the text-to-sql capabilities of large language models. *arXiv preprint arXiv:2204.00498*.

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

Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. 2019. Rat-sql: Relation-aware schema encoding and linking for text-to-sql parsers. *arXiv preprint arXiv:1911.04942*.

Jiaan Wang, Yunlong Liang, Fandong Meng, Zhixu Li, Jianfeng Qu, and Jie Zhou. 2023. Crosslingual summarization via chatgpt. *arXiv preprint arXiv:2302.14229*.

Lihan Wang, Bowen Qin, Binyuan Hui, Bowen Li, Min Yang, Bailin Wang, Binhua Li, Jian Sun, Fei Huang, Luo Si, et al. 2022. Proton: Probing schema linking information from pre-trained language models for text-to-sql parsing. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1889–1898.

Lijie Wang, Ao Zhang, Kun Wu, Ke Sun, Zhenghua Li, Hua Wu, Min Zhang, and Haifeng Wang. 2020. Dusql: A large-scale and pragmatic chinese text-to-sql dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6923–6935.

Xiang Wei, Xingyu Cui, Ning Cheng, Xiaobin Wang, Xin Zhang, Shen Huang, Pengjun Xie, Jinan Xu, Yufeng Chen, Meishan Zhang, et al. 2023. Zeroshot information extraction via chatting with chatgpt. *arXiv preprint arXiv:2302.10205*.

Dongling Xiao, Linzheng Chai, Qian-Wen Zhang, Zhao Yan, Zhoujun Li, and Yunbo Cao. 2022. Cqr-sql: Conversational question reformulation enhanced context-dependent text-to-sql parsers. *arXiv preprint arXiv:2205.07686*.

Tao Yu, Rui Zhang, He Yang Er, Suyi Li, Eric Xue, Bo Pang, Xi Victoria Lin, Yi Chern Tan, Tianze Shi, Zihan Li, et al. 2019a. Cosql: A conversational text-to-sql challenge towards cross-domain natural language interfaces to databases. *arXiv preprint arXiv:1909.05378*.

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

Tao Yu, Rui Zhang, Michihiro Yasunaga, Yi Chern Tan, Xi Victoria Lin, Suyi Li, Heyang Er, Irene Li, Bo Pang, Tao Chen, et al. 2019b. Sparc: Crossdomain semantic parsing in context. *arXiv preprint arXiv:1906.02285*.

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