# Jointly Learning Conversational Semantic Parsing and Answerability Detection

**Anonymous ACL submission**

## Abstract

Conversational semantic parsing is a challenging task that aims to automatically translate user utterances into logic forms (e.g., SQL queries) in multi-turn interactions. Most existing conversational semantic parsing models handle this task by assuming the user utterances are well-formed and answerable. Although these models have achieved prompting results on the Text2SQL task, few methods consider the answerability detection problem, causing the conversational semantic parser not able to deal with the practical scenario. To fill this gap, we propose to jointly learn the conversational semantic parsing and the answerability detection task on top of the pretrained sequence to sequence model. In this way, the model would be able to detect the answerability of the user utterance, respond with the translated SQL query for the answerable questions, and generate clarification answers for the unanswerable and ambiguous questions. Experimental results show that our joint learning framework performs satisfactorily for the answerability detection task, and results in performance improvements in terms of the generated SQL quality.

## 1 Introduction

Semantic parsing aims to translate natural language questions into machine-readable logical forms, such as SQL. Most previous text-to-SQL works (Rubin and Berant, 2021; Cao et al., 2021; Lin et al., 2020a,b; Wang et al., 2020a) focus on single-turn interaction between user and machine, where an individual utterance is translated into executable SQL query by the semantic parsing model. However, in practice users tend to explore the database in multi-turn interactions as shown in Table 1. To this end, Yu et al. propose the SparC (Yu et al., 2019b) and CoSQL (Yu et al., 2019a) dataset for conversational text-to-SQL towards cross-domain natrual language interfaces to databases. Compared to traditional semantic parsing, conversational semantic parsing is more challenging because it requires contextual understanding of user utterances.

| |
|---|
| **User:** How many dorms have a TV Lounge? |
| **Response:** Found 28 dorms. |
| **User:** Which one is closest to the University? |
| **Response:** *Sorry, do you mean among those dorms that have a TV lounge?* |
| **User:** Yes. |
| **Response:** The Lochrin Place dorm. |
| **User:** How many students are from the UK? |
| **Response:** *Sorry, unanswerable.* |

Table 1: An example dialog for our proposed conversational semantic parser with answerability detection. Grey boxes are the user inputs and the blues boxes are the model responses. The first sentence in italics is a clarification answer. The second sentence in italics means the current user question is not answerable.

Recent works (Yu et al., 2019a, 2020b; Hui et al., 2021; Wang et al., 2020b; Cai and Wan, 2020) in conversational semantic parsing focus on utilizing pretrained models to exploit the context information to improve the quality of translated SQL queries. Although encouraging progress has been achieved (Yu et al., 2019a, 2020b; Hui et al., 2021; Wang et al., 2020b; Cai and Wan, 2020), most current works assume the user questions are legal (Yu et al., 2020b; Hui et al., 2021; Wang et al., 2020b; Cai and Wan, 2020) and output a SQL query for any input, which is inconsistent with the real scenario. Practically, user questions can be ambiguous or unanswerable, which requires the system to be not only capable of translating natural language into SQL query, but also detecting the answerability of questions and generating clarification answers for the ambiguous questions. Recently, Yu et al.(Yu et al., 2019a) and Zhang et al.(Zhang et al., 2020) propose to regard the answerability detection as a separate classification task and ensemble the classification model with the semantic parser to make a complete dialogue system (Zhang et al., 2020).

However, this approach requires separately training an answerability detection model, a natural language decoder for clarification response generation, and another decoder for SQL query generation, which is rather complicated and cumbersome.

To address the aforementioned issues, we investigate to jointly learn the conversational semantic parsing task with answerability detection and resolution using the sequence to sequence (Sutskever et al., 2014) architecture with a single encoder and decoder. Figure 1 shows the illustration of our proposed framework. Using the joint learning setup, we hypothesize that the encoder can learn the intermediate features that encode information effectively for all three downstream tasks. Meanwhile, we hypothesize that the decoder can learn to identify whether a user question is answerable based on both the context and the knowledge base, generate clarification answers for ambiguous questions, and generate SQL queries for answerable questions simultaneously. To be more specific, as pretrained sequence to sequence models nowadays are widely used in most sequence to sequence tasks, we choose to use T5 (Raffel et al., 2019), the most popular large-scale pretrained model for sequence transduction, as the backbone model in our experiments. Due to the lack of dataset-specific to the answerability detection task, we propose a novel dynamically negative samples generation method during the training process to augment the unanswerable questions. Additionally, since the T5 model is pretrained on human language corpus and the target domain includes SQL query, we apply the two-stage finetuning (Gururangan et al., 2020) to transfer the model to the target domain gradually and smoothly. We conduct experiments based on the CoSQL (Yu et al., 2019a) dataset. The experimental result shows that our joint learning framework performs satisfactorily for the answerability detection task, and results in performance improvements in terms of the generated SQL quality. The contributions can be summarized as follows:

- We propose a jointly learning framework that can learn conversational semantic parsing and answerability detection and clarification.
- We propose a novel dynamically negative sample generation method for unanswerable data augmentation, and apply the two-stage finetuning strategy for domain transfer.
- The experimental result shows our joint learning framework performs satisfactorily for answerability detection, and results in performance improvements of SQL generation.

## 2 Proposed Method

Our proposed model is built on T5, a pretrained sequence to sequence model, which requires both the input and output are sequences. Therefore, we elaborately design the input format to linearize the different input components (interaction history $I$, current user utterance $U$, and the database schema $S$). For the output format, we use the first token of the output sequence to indicate whether the question is answerable or not. If answerable, the following part of the sequence will be the machine-executable SQL query. If the question is ambiguous, the following part will be the clarification response. Lastly, the output would be a constant response if the question is not unanswerable (e.g., unrelated to the knowledge base). During the training stage, we propose a novel dynamic negative sample generation method for the unanswerable question augmentation. Additionally, we exploit two-stage finetuning strategy for gradual domain transfer, which recently shows great performance on the graph-to-text task (Ribeiro et al., 2021).

### 2.1 Task Formulation

In conversational semantic parsing (i.e., text-to-SQL) task, we have three input components: interaction history $I$, database schema $U$, and the current user question $S$. Each interaction includes human utterance $Q_i (1 \leq i \leq K)$ and corrresponding response. The response could be either a machine-readable SQL query or a clarification answer (e.g., disambiguation, greetings, etc), which depends on whether the question is legal, answerable, and unambiguous. Overall speaking, the task requires the model to detect whether the current user utterance is answerable based on the interaction history and database schema. For the answerable questions, the model needs to generate executable SQL queries. For the unanswerable questions, the model needs to generate proper clarification to guide the user clarify the ambiguous question or describe the situation (e.g., not answerable for the current database). In this project, we define three types of output, namely answerable, not answerable based on the given schema, and ambiguous questions.

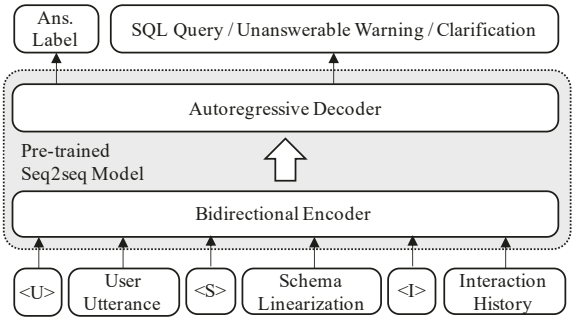


Figure 1: The illustration of our proposed framework. We append different special tokens before different components in the input linearization. The Ans. Label stands for answerability label. For the answerable questions, our model outputs the corresponding executable SQL query. If the question is irrelevant to the database, the model will respond with a warning. If the question is ambiguous, our model will generate a clarification response to guide the users clarify their questions.

### 2.2 Linearization

The input comprises the current user utterance $U$, the database schema $S$, and the interaction history $I$. We concatenate different components and add different special tokens at the beginning of different components. As shown in the euqation 1, we prepend *<U>*, *<S>*, and *<I>* at the beginning of the current user utterance $U$, the database schema $S$, and the interaction history $I$ respectively. We do this inspired by the Google Multilingual Translation (Johnson et al., 2017), which argues that such usage of special tokens is able to make the model learn to be aware of the role of the following parts.

*<U> utterance <S> schema <I> interactions* (1)

Note that each database schema may contain more than one tables. As shown in equation 2, we append *<TAB>* at the beginning of each table name, and append *<COL>* at the the beginning of each column name followed by the data type.

*<TAB> table <COL> column_name data_type* (2)

Equation 3 shows how to linearize the interactions:

$$\textit{<Q> ques}_k \textit{ <Q> ques}_{k-1} \textit{ ... <Q> ques}_0 \quad (3)$$

The order of the components are elaborately designed due to the input length limitation. We put the current user utterance $U$ at the beginning of the linearization because this is the most important component of the input and we don't want it to be truncated. The interaction history is put at the end of the input because it's less important compared to the other two components. In many cases we don't need to know the previous conversation to answer the current questions. Within the linearization of the interaction history, we put the most recent question $k$ at the beginning because the most recent history is more relevant to the current utterance. We use the first token of the output sequence to indicate the answerability of the question. More precise, we use *<0>* to indicate the question is answerable and followed by the translated SQL query. *<1>* means the question is unanswerable based on the given schema, and *<2>* means the question is ambiguous and will be followed by a clarification response. For the SQL queries, we capitalize all the SQL keywords to distinct them from the corresponding English words (e.g., SELECT vs. select) because we want the model to learn different embeddings for the SQL keywords instead of sharing the embeddings across SQL and natural language.

### 2.3 Two-stage Finetuning

Inspired by recent works (Gururangan et al., 2020; Ribeiro et al., 2021) that have shown the benefit of task-specific adaptation, we investigate whether leveraging additional task-specific data can improve the performance of pretrained language models on the conversational semantic parsing task. Task-specific data refers to a corpus that is from relevant (not exactly the same) domains of the downstream task. In order to leverage the task-specific data, we add an intermediate adaptive fine-tuning step between the original pretraining and the fine-tuning stage for conversational semantic parsing. More specifically, we first continue fine-tuning the pretrained sequence to sequence model (i.e., T5) on Spider (Yu et al., 2018), a single-turn text-to-SQL dataset. The goal is to adapt the pretrained model to the target domain gradually and smoothly. We use the same linearization methods for both Spider and CoSQL. The only difference is that we don't have the interaction history in the Spider dataset and the linearization wouldn't contain this component.

### 2.4 Dynamic Negative Sample Generation

In real-world scenarios, users may ask questions irrelevant to the database or ambiguous questions that need clarifications to be answerable. Our proposed model is able to detect the answerability. More precisely, we use the first token in the output sequence to classify the input utterance into three

categories: answerable, unanswerable based on the given schema, and ambiguous questions. CoSQL provides ambiguous questions and corresponding human-written answers. But there doesn't exist any unanswerable questions in CoSQL. Therefore, we propose to generate unanswerable questions by replacing the original schema of an answerable sample with randomly selected knowledge schema.

However, it would be prone to overfit the dataset if we statically generate the negative (i.e., unanswerable) samples before training. Inspired by how RoBERTa (Liu et al., 2019) improves BERT(Devlin et al., 2019), we propose to generate the unanswerable samples dynamically during the training stage. Before each epoch in training process, each answerable sample is corrupted into an unanswerable question with a probability of 0.2 by replacing the current schema with randomly selected schema of another knowledge base. This can give our model the ability to detect if the use question is irrelevant to the knowledge base and unanswerable, and generate appropriate responses to the users.

# 3 Experiments

## 3.1 Dataste

**CoSQL**[1] We train and evaluate our model on the CoSQL (Yu et al., 2019a) dataset, which consists of 30,000 turns and 10,000 annotated SQL queries. It is obtained from a Wizard-of-Oz collection of 3k dialogues querying 200 complex databases spanning 138 domains. Each dialogue simulates a real-world DB query scenario with a crowd worker as a user exploring the database and a SQL expert retrieving answers with SQL, clarifying ambiguous questions, or otherwise informing of unanswerable questions. The original CoSQL doesn't explicitly annotate the ambiguous questions and they are concatenated with clarifications as new inputs in the dataset. We extract these samples manually and add the pairs of ambiguous utterances and corresponding clarification answers into the dataset. Note that the original CoSQL doesn't contain any unanswerable questions and we generate such data by the dynamic algorithm as described in Section 2.4.

**Spider**[2] Spider is a large-scale complex and cross-domain semantic parsing and text-to-SQL dataset annotated by 11 Yale students. The goal of the Spider challenge is to develop natural language interfaces to cross-domain databases. It consists of 10,181 questions and 5,693 unique complex SQL queries on 200 databases with multiple tables covering 138 different domains. In Spider 1.0, different complex SQL queries and databases appear in train and test sets. To do well on it, systems must generalize well to not only new SQL queries but also new database schemas. As described in Section 2.3, we use the Spider dataset as the external task-specific dataset for the two-stage finetuning strategy.

## 3.2 Implementation Details

The model was implemented using PyTorch Lightning[3] and T5 models provided by HuggingFace[4]. We used T5-base in our experiments. We remove all punctuations and special tokens from the dataset (both CoSQL and Spider). For the column type , we remove the detailed information and only keep the type word. For example, $varchar(15)$ will be modified to $varchar$. Due to the resource constraint, we set the max lengths to 512 and 128 for the input and output respectively. We finetuned the model on one NVIDIA T4 GPU. We use the AdamW (Loshchilov and Hutter, 2018) optimizer The batch size is 8 and learning rate is set to 0.0001. The early stopping is used to monitor the corss-entropy loss on the validation set with patience of 10.

## 3.3 Metrics

We need to evaluate both the quality of generated SQL queries and the detection accuracy in our proposed jointly learning framework.

**For SQL queries** We use exact set matching to evaluate the quality of generated SQL queries. Note that the task definition of both Spider and CoSQL does not predict the value strings. Predicting correct SQL query structures and columns is more realistic and critical according to the original paper of Spider. We follow the evaluation setting of the Spider dataset, which does not take the value strings into account. More precisely, exact matching is a component-based evaluation method that decomposes each component in both prediction and ground truth as bags of sub-components, and check if the two sets of components match exactly. Besides, interaction matching is used for evaluating the generated SQL queries in conversational level. The exact set matching score is 1 for each question only if all predicted SQL clauses are correct, and

[1] https://yale-lily.github.io/cosql
[2] https://yale-lily.github.io/spider
[3] https://www.pytorchlightning.ai/
[4] https://huggingface.co/

| Model Name | Exact Set Match | Interaction Match |
|---|---|---|
| CD-Seq2seq | 13.9 | 2.6 |
| T5 baseline | 20.4 | 7.0 |
| + Special Tokens | 29.3 | 8.2 |
| + Capitalization | 31.4 | 8.6 |
| + Two-stage finetuning | **34.3** | **10.4** |

Table 2: The results of SQL query prediction. CD-Seq2seq refers to Context-Dependent Seq2Seq model proposed in the original CoSQL paper. T5 baselines model means directly fine-tuning the T5-base model on the CoSQL with our proposed linearization methods.

1 for each interaction only if there is an exact set match for every question in the interaction.

**For answerability detection accuracy** We regard the answerability detection as a classification task and use the recall to evaluate the performance.

### 3.4 Results

**SQL prediction** Table 2 shows the results of SQL prediction task. We can observe that T5 baseline outperforms the CD-Seq2seq model on both exact set match score and interaction match score. This indicates that knowledge T5 learning during the pretraining is helpful for the conversational semantic parsing task. Adding different special tokens before different components in the input can improve the exact set match score by 8.9 points and 1.2 points in terms of the interaction match score. We could gain another 2.1 point and 0.4 point improvement on exact set match and interaction match score by capitalizing the SQL keywords in the output sequence. This shows that the hypothesis of treating SQL and English as different languages helps with our task. In the two-stage finetuning setting, we finetune the T5 model on Spider dataset before finetuning on CoSQL. This could bring us 2.9 points improvement on exact set match score and 1.8 on the interaction match.

**Answerability Detection** Table 3 shows the results of the answerability detection experiments. We can observe that the jointly learning setting could slightly improve the exact set match score. Without the dynamic negative sample generation method, the recalls for the answerable and unanswerable questions are 98.7% and 96%. The reason for this is there are significant features in answerable and unanswerable questions, i.e., overlapped column or table names. Our proposed dynamic negative sample generation algorithm could improve the recall of unanswerable questions by 2.2%, which shows the effectiveness of this method. Meanwhile, it can bring minor improvement on exact set match score and the recall scores for answerable. However, we can observe that the recall for ambiguous question is only around 28%. We found that in many cases the model generates SQL queries despite the user questions are ambiguous. Another problem is different tasks don't converge simultaneously. We can observe from Figure 2 that the exact set match score, the recall for answerable and unanswerable questions reach optima in the first few epochs and then slowly decrease. However, the recall for ambiguous question stay zero until the $15^{th}$ epoch and then gradually increase.

### 3.5 Case Study

In this section we analyse some cases for the SQL prediction and disambiguation task.

**SQL prediction** Table 4 shows a bad case of SQL prediction. In the first example, from the schema we can observe that $LocalName$ is a column in table $country$. However, our model predicts $LocalName$ belongs to another table $city$. This means that our model doesn't manage to parse the connection between tables and columns in some cases. We argue the reason is that T5 is pretrained on English corpus and is not able to link the table/column name and the query word. We could incorporate schema linking related pretraining objectives to address the issue (Yu et al., 2020a).

**Disambiguation** Table 5 is a case for disambiguation. We can observe from the schema that we have two columns related to name – $first\_name$ and $last\_name$. This would cause ambiguation if the user doesn't specify it is the first or the last name they want. The model of $5^{th}$ epoch is the model with best SQL prediction performance, and it classifies this case as answerable question and generates a SQL to retrieval the $first\_name$. While the model of the last epoch performs best on the ambiguation question detection, and it classifier this case as ambiguous question and generate reasonable clarification answer.

## 4 Conclusion

In this paper we propose a joint learning framework for both conversational semantic parsing and the answerability detection task. Experimental results

| Model | Exact Set Match | Ans. Recall | Unans. Recall | Ambig. Recall |
|---|---|---|---|---|
| T5 + SpeTok + Cap | 31.4 | - | - | - |
| + AnswDet | 31.7 | 98.7% | 96.0% (epoch 10) | **28.4%** (epoch 30) |
| + DynGeneration | **32.0** | **98.9%** | **98.2%** (epoch 19) | 28.3% (epoch 30) |

Table 3: The results of answerability detection experiments. SpeTokn, Cap, AnswDet, and DynGeneartion mean special tokens, capitalization, answerability detection, and dynamic negative sample generation respectively.

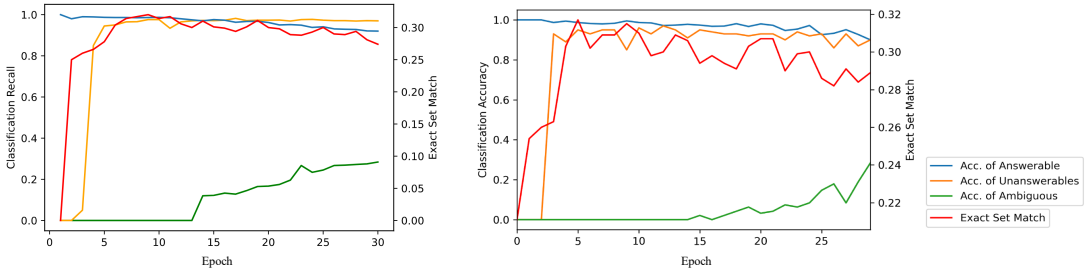


Figure 2: The results of the answerability detection experiments. Left is the model with dynamic negative sample generation algorithm while the right figure is the model without dynamic negative sample generation algorithm.

| |
|---|
| **Current User Question:** What is the local name?<br>**Schema:** <TAB> city <COL> Name char <COL> District char <COL> Population integer ... <TAB> country <COL> LocalName char <COL> GovernmentForm char <COL> Capital integer ...<br>**Ground Truth:**<br><0> SELECT LocalName FROM country |
| **Model Prediction:**<br><0> SELECT LocalName FROM city |

Table 4: The case study of SQL prediction.

| |
|---|
| **Current User Question:** Which player got first rank in the competition and please tell me their name.<br>**Schema:** <TAB> players <COL> player_id INT <COL> first_name TEXT <COL> last_name ...<br>**Interactions:** Find the total ranking point for each player in the competition and their first name. |
| **Ground Truth:**<br><2> Do you want the first name or last name? |
| **Output of Epoch 5 (Best SQL Score):**<br><0> SELECT T1.first_rank, T1.first_name FROM players AS T1 JOIN rankings AS T2 ON T1.player_id = T2.player_id GROUP BY T1.player_id ORDER BY COUNT (*) DESC LIMIT |
| **Output of epoch 39 (Best Recall of Ambig.):**<br><2> Do you mean the first name and last name? |

Table 5: The case study of disambiguation.

indicate that the parser benefits from answerability detection task. There are several future research directions: (1) Apply the joint learning framework on top of other sequence to sequence model to test the generalization ability. (2) Assign different weights to balance different tasks and make them converge simultaneously. (3) Generalize the dynamic negative sample generation to the ambiguous questions.

## References


Yitao Cai and Xiaojun Wan. 2020. IGSQL: Database schema interaction graph based neural model for context-dependent text-to-SQL generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6903–6912, Online. Association for Computational Linguistics.

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 *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 2541–2555.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360.

Binyuan Hui, Ruiying Geng, Qiyu Ren, Binhua Li, Yongbin Li, Jian Sun, Fei Huang, Luo Si, Pengfei Zhu, and Xiaodan Zhu. 2021. Dynamic hybrid relation network for cross-domain context-dependent semantic parsing. *arXiv preprint arXiv:2101.01686*.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Xi Victoria Lin, Richard Socher, and Caiming Xiong. 2020a. Bridging textual and tabular data for cross-domain text-to-sql semantic parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 4870–4888.

Xi Victoria Lin, Richard Socher, and Caiming Xiong. 2020b. Bridging textual and tabular data for cross-domain text-to-SQL semantic parsing. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4870–4888, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2018. Fixing weight decay regularization in adam.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Leonardo FR Ribeiro, Martin Schmitt, Hinrich Schütze, and Iryna Gurevych. 2021. Investigating pretrained language models for graph-to-text generation. In *Proceedings of the 3rd Workshop on Natural Language Processing for Conversational AI*, pages 211–227.

Ohad Rubin and Jonathan Berant. 2021. Smbop: Semi-autoregressive bottom-up semantic parsing. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 311–324.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. 2020a. 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.

Run-Ze Wang, Zhen-Hua Ling, Jing-Bo Zhou, and Yu Hu. 2020b. Tracking interaction states for multi-turn text-to-sql semantic parsing. *arXiv preprint arXiv:2012.04995*.

Tao Yu, Chien-Sheng Wu, Xi Victoria Lin, Yi Chern Tan, Xinyi Yang, Dragomir Radev, Caiming Xiong, et al. 2020a. Grappa: Grammar-augmented pre-training for table semantic parsing. In *International Conference on Learning Representations*.

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, Alex Polozov, Christopher Meek, and Ahmed Hassan Awadallah. 2020b. Score: Pre-training for context representation in conversational semantic parsing. In *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, et al. 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.

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: Cross-domain semantic parsing in context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4511–4523.

Yusen Zhang, Xiangyu Dong, Shuaichen Chang, Tao Yu, Peng Shi, and Rui Zhang. 2020. Did you ask a good question? a cross-domain question intention classification benchmark for text-to-sql. *arXiv preprint arXiv:2010.12634*.