STAR: SQL Guided Pre-Training for Context-dependent Text-to-SQL Parsing

Zefeng Cai[1,2,✊], **Xiangyu Li**[1,2,✊], **Binyuan Hui**[3], **Min Yang**[2†], **Bowen Li**[3], **Binhua Li**[3], **Zheng Cao**[3], **Weijie Li**[3], **Fei Huang**[3], **Luo Si**[3], **Yongbin Li**[3†]

[1] University of Science and Technology of China

[2] Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences

[3] DAMO Academy, Alibaba Group

{zf.cai, xy.li3, min.yang}@siat.ac.cn

{binyuan.hby, binhua.lbh, shuide.lyb}@alibaba-inc.com

Abstract

In this paper, we propose a novel SQL guided pre-training framework STAR for context-dependent text-to-SQL parsing, which leverages contextual information to enrich natural language (NL) utterance and table schema representations for text-to-SQL conversations. Concretely, we propose two novel pre-training objectives which respectively explore the context-dependent interactions of NL utterances and SQL queries within each text-to-SQL conversation: (i) schema state tracking (SST) objective that tracks and explores the schema states of context-dependent SQL queries in the form of schema-states by predicting and updating the value of each schema slot during interaction; (ii) utterance dependency tracking (UDT) objective that employs weighted contrastive learning to pull together two semantically similar NL utterances and push away the representations of semantically dissimilar NL utterances within each conversation. In addition, we construct a high-quality large-scale context-dependent text-to-SQL conversation corpus to pre-train STAR. Extensive experiments show that STAR achieves new state-of-the-art performance on two downstream benchmarks (SPARC and COSQL), significantly outperforming previous pre-training methods and ranking first on the leaderboard. We believe the release of the constructed corpus, codebase and pre-trained STAR checkpoints would push forward the research in this area. For reproducibility, we release our code and data at https://github.com/AlibabaResearch/DAMO-ConvAI/tree/main/star.

1 Introduction

Text-to-SQL parsing (Zhong et al., 2017; Yu et al., 2018; Wang et al., 2022; Qin et al., 2022b) aims to translate natural language (NL) questions into executable SQL queries, which enables the users who are unfamiliar with SQL to query databases with natural language. Pre-trained language models (PLMs) have proved to be powerful in enhancing text-to-SQL parsing and yield impressive performances, which benefit from the rich linguistic knowledge in large-scale corpora. However, as revealed in previous works (Yin et al., 2020; Yu et al., 2021a; Qin et al., 2022a), there are intrinsic discrepancy between the distributions of tables and plain texts, leading to sub-optimal performances of general PLMs such as BERT (Devlin et al., 2019), ROBERTA (Liu et al., 2019), ELECTRA (Clark et al., 2020). Recently, some studies (Yu et al., 2021a,b; Shi et al., 2021; Deng et al., 2021; Liu et al., 2021a,b) alleviate the above limitation by designing tailored tabular language models (TaLMs) for text-to-SQL parsing, which simultaneously encode NL questions and tables.

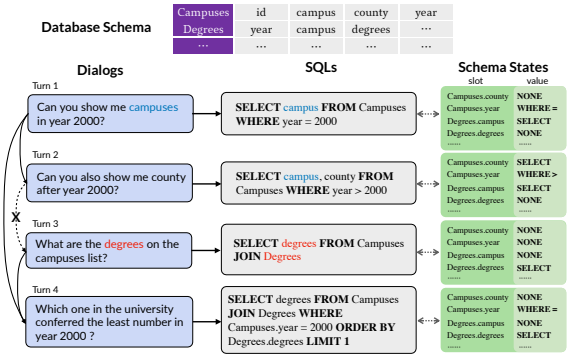


Figure 1: An example of cross-domain context-dependent Text-to-SQL conversation. Here, each database schema refers to the table/column names of databases and each schema state refers to a slot-value pair, whose slot is a column/table name (e.g., Degrees.campus) and its value is a SQL keyword (e.g., SELECT). "x" indicates that the semantic/intent is switched between Turn2 and Turn3 utterances.

Despite the remarkable progress of previous TaLMs, they still suffer from technical challenges in the context-dependent setting. **First**, existing TaLMs merely explore contextual information to enrich utterance representations without considering the interaction states determined by history

✊ Equal contribution.

†Corresponding authors.

SQL queries, which are relevant to the user intent of current utterance. Nevertheless, the trace and usage of historical SQL information can contribute greatly to model the current SQL query, as SQL conveys user intent in a compact and precise manner. As shown in Figure 1, the second SQL query is more likely to select the contents from the "Compuses" table since the first SQL query mentioned that table. Although tracking schema states is essential to keep track of user requests for context-dependent text-to-SQL parsing, how to model, track and utilize schema states throughout a conversation has not yet been explored in previous TaLMs. **Second**, context-dependent text-to-SQL parsing needs to effectively process context information so as to help the system better parse current NL utterance, since users may omit previously mentioned entities as well as constraints and introduce substitutions to what has already been stated. Taking Figure 1 as an example, the second utterance omit the implicit constraint of "campuses in year 2000" as mentioned in the first utterance. However, most prior TaLMs primarily model stand-alone NL utterances without considering the context-dependent interactions, which result in suboptimal performance. Although SCoRe (Yu et al., 2021b) model the turn contextual switch by predicting the context switch label between two consecutive user utterances, it ignores the complex interactions of context utterances and cannot track the dependence between distant utterances. For instance, in Figure 1, SCoRe fails to capture the long term dependency between the first and the fourth utterances since there is a switch between the second and the third utterances.

In this paper, we propose a novel pre-training framework STaR for context-dependent text-to-SQL parsing, which explores the multi-turn interactions of NL utterances and SQL queries within each conversation, respectively. **First**, we propose a schema state tracking (SST) objective to keep track of SQL queries in the form of schema-states, which predicts the value (a SQL keyword) of each schema slot of the current SQL query given the schema-state representation of previously predicted SQL query. By introducing the schema-states to represent SQL queries, we can better capture the alignment between the the historical and current SQL queries, especially for the long and complex SQL queries. **Second**, we propose an utterance dependency tracking (UDT) objective to capture complex semantic dependency of sequential NL questions, which employs weighted contrastive learning to pull together semantically similar NL utterances and push away dissimilar NL utterances within each conversation. A key insight is that the utterance corresponding to similar SQL will be more semantically relevant, as SQL is a highly structured indication of user intent. Concretely, we propose two novel similarity functions (SQL semantic similarity and SQL structure similarity) to comprehensively construct appropriate positive and negative NL question pairs.

We summarize our main contributions as follows. (1) To the best of our knowledge, we are the first to propose a schema state tracking (SST) objective for context-dependent TaLM, which tracks and updates the schema states of the context-dependent SQL queries in the form of schema states. (2) We propose an utterance dependency tracking (UDT) objective to capture complex semantic information of sequential NL questions, which employs weighted contrastive learning with two novel SQL-oriented similarity functions to pull together two semantically similar NL utterances and push away the representations of dissimilar NL utterances within each conversation. (3) We construct a high-quality large-scale context-dependent text-to-SQL conversation corpus to pre-train STaR. Experiments show that STaR achieves new state-of-the-art performance on two downstream benchmarks (SPARC and CoSQL) and ranking first on the leaderboard.

2 Task Definition

In this section, we first provide the formal task definition for context-dependent text-to-SQL parsing. Let $U = \{u_1, \ldots, u_T\}$ denote the utterances in a context-dependent text-to-SQL conversation with T turns, where u_i represents the i-th NL question. Each NL sentence u_i contains n_i tokens, denoted as $u_i = [w_1, \ldots, w_{n_i}]$. In addition, there is a corresponding database schema s, which consists of N tables $\{\mathcal{T}_i\}_{i=1}^{N}$. The number of columns of all tables in the schema is m. We use s^i to denote the name of the i-th item in schema s. At current turn t, the goal of text-to-SQL parsing is to generate the SQL query o_t given the current utterance u_t, historical utterances $\{u_1, \ldots, u_{t-1}\}$, schema s, and the last predicted SQL query o_{t-1}. STaR primarily consists of a stack of Transformer layer, which converts a sequence of L input tokens $x = [x_1, ..., x_L]$ into a sequence of contextualized vector represen-

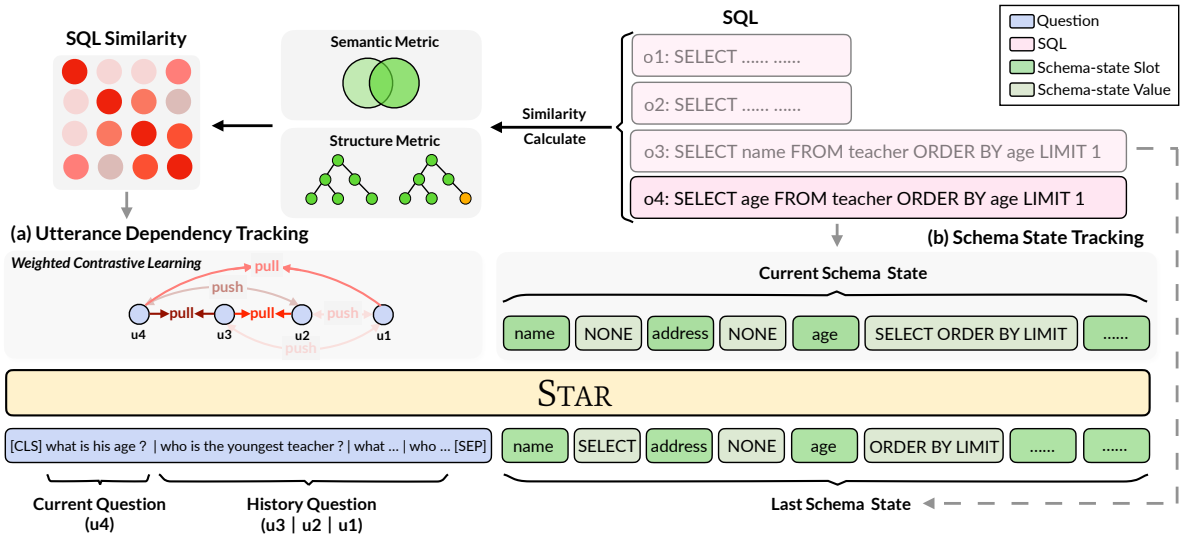


Figure 2: The overview of the proposed STAR framework consisting of two novel pre-training objectives: (a) the utterance dependency tracking and (b) the schema state tracking. For brevity, we do not show the masked language modeling objective here.

tations $\mathbf{h} = [\mathbf{h}_1, \ldots, \mathbf{h}_L]$.

3 Pre-training Objectives

As illustrated in Figure 2, we propose two novel pre-training objectives **SST** (**S**chema **S**tate **T**racking) and **UDT** (**U**tterance **D**ependency **T**racking) to explore the complex context interactions of NL utterances and SQL queries within each text-to-SQL conversation, respectively. In addition, we also employ the **MLM** (**M**asked **L**anguage **M**odeling) objective to help learn better contextual representations of the conversations. Next, we will introduce the pre-training objectives in detail.

3.1 Schema State Tracking

The usage of context SQL information contributes greatly to model the current SQL query. Inspired by the dialogue state tracking (Ouyang et al., 2020; Wang et al., 2021a) which keeps track of user intentions in the form of a set of dialogue states (*i.e.* , slot-value pairs) in task-oriented dialogue systems, we propose a schema state tracking (SST) objective in a self-supervised manner to keep track of schema states (or user requests) of context-dependent SQL queries, which aims to predict the values of the schema slots. Concretely, we track the interaction states of the text-to-SQL conversation in the form of schema-states whose slots are column names of all tables in the schema and their values are from SQL keywords. Taking the SQL query in Figure 3 as example, the value of the schema slot `[cars_data]` is the SQL keyword `[SELECT]`.

Formally, we first convert the last predicted SQL query o_{t-1} into a set of schema states. Since the names of schema states are names of all schema, the values of those schema states that do not appear in the last SQL query o_{t-1} are set to `[NONE]`, as shown in Figure 3. We represent the SQL query o_{t-1} with m schema-states $\{(s^i_{t-1}, v^i_{t-1})\}^m_{i=1}$, where s^i_{t-1} denotes the schema-state slot, v^i_{t-1} denotes the schema-state value of the slot s^i_{t-1}, and m represents the number of schema. At the t-th turn, the goal of SST is to predict the value v^i_t of each schema-state slot s^i_t of the t-th SQL query given all the history utterances $\{u_1, \ldots, u_{t-1}\}$, the current utterance u_t and the schema-states $\{(s^i_{t-1}, v^i_{t-1})\}^m_{i=1}$ of the late query o_{t-1}. That is, at the t-th turn, the input I_t of the SST task is as:

$$I_t = \left[\{u_1, \ldots, u_t\}; \{(s^i_{t-1}, v^i_{t-1})\}^m_{i=1}\right] \quad (1)$$

Note that the SQL queries within a conversation share the same schema s, thus the schema-states of the t-th and $t-1$-th SQL queries have the same schema-state slots (i.e., $s^i_{t-1} = s^i_t = s^i$).

Since each schema state $c^i_{t-1} = (s^i_{t-1}, v^i_{t-1})$ contains multiple words, we apply an attentive layer to obtain the representation of $c^i_{t-1} = (s^i_{t-1}, v^i_{t-1})$. Concretely, given the output contextualized representation $\mathbf{h}^{c^i_{t-1}}_t = [\mathbf{h}^l_t, \ldots, \mathbf{h}^{l+|c^i_{t-1}|-1}_t]$ (l is the start index of c^i_{t-1}) of each schema state c^i_{t-1}, the attentive schema-state representation $\mathbf{c}^i_{t-1}$

of the schema state c^i_{t-1} can be calculated as:

$$\alpha^j_{t-1} = \text{softmax}\left(\tanh(\mathbf{h}^{l+j}_t \mathbf{W}_1)\mathbf{v}_1^\top\right) \quad (2)$$

$$\mathbf{c}^i_{t-1} = \sum_{j=1}^{|c^i_{t-1}|} \alpha^j_{t-1} \mathbf{h}^{l+j}_t \quad (3)$$

where $\mathbf{v}_1$ and $\mathbf{W}_1$ are trainable parameters. We use the attentive schema-state representation $\mathbf{c}^i_{t-1}$ in the last SQL query to predict the value v^i_t of the current schema state $\mathbf{c}^i_t$:

$$P(\mathbf{c}^i_t|\mathbf{c}^i_{t-1}) = \text{softmax}(\mathbf{W}_2\mathbf{c}^i_{t-1} + \mathbf{b}_2) \quad (4)$$

where $\mathbf{W}_2$ and $\mathbf{b}_2$ are trainable parameters.

Finally, the pre-training loss function of SST is defined as the cross-entropy between the predicted schema-state value $P(v^i_t|\mathbf{c}^i_{t-1})$ and the gold schema-state value v^i_t as follows:

$$\mathcal{L}_{\text{SST}} = -\frac{1}{m}\sum_{i=1}^{m} \mathbf{c}^i_t \log P(\mathbf{c}^i_t|\mathbf{c}^i_{t-1}) \quad (5)$$

where m is the number of slot (schema).

3.2 Utterance Dependency Tracking

We propose an utterance dependency tracking (UDT) objective to capture complex semantic dependency of sequential NL questions within each text-to-SQL conversation. A key challenge behind UDT is how to construct appropriate positive and negative labels by way of self-supervision.

Generally, it is intuitive that we can construct negative utterance pairs by selecting NL utterances from different conversations. However, it is non-trivial to construct positive utterance pairs, since the current utterance may be irrelevant to those of the historical utterances with prominent contextual shifts, as the second and third utterances shown in Figure 1. Hence, we treat the NL utterances within the same conversation as positive pairs, which are assigned with different similarity scores. SQL is a highly structured indication of user utterance, so by measuring the similarity of current SQL to historical SQL, pseudo-labels of utterance semantic dependencies can be obtained to guide the STAR in contextual modelling. Here we propose a method to measure SQL similarity from two perspectives.

SQL Semantic Similarity To compute the similarity of two SQL queries, we first convert each SQL query into m schema-states as described in Section 3.1, where the schema slots are names of all schema and their values are from SQL keywords. As illustrated in Figure 3, given two SQL queries (denotes as o_x and o_y), we obtain the schema states $\{(s^i_x, v^i_x)\}^m_{i=1}$ and $\{(s^i_y, v^i_y)\}^m_{i=1}$ of the SQL queries o_x and o_y respectively. Since all the schema-states share the same schema slots, we have $s^i_x = s^i_y$. Then, we adopt the Jaccard similarity (Niwattanakul et al., 2013) to compute the semantic similarity of the SQL queries o_x and o_y by comparing v^i_x and v^i_y. Mathematically, we compute the SQL semantic similarity of o_x and o_y as:

$$f_{\text{semantic}}(o_x, o_y) = \frac{\sum_{i=1}^{m} \text{Jaccard}(v^i_x, v^i_y)}{|\hat{s}_{x,y}|} \quad (6)$$

$$\text{Jaccard}(v^i_x, v^i_y) = \frac{|v^i_x \cap v^i_y|}{|v^i_x \cup v^i_y|} \quad (7)$$

where $|\hat{s}_{x,y}|$ represents the number of non-duplicate schema states whose values are not `[NONE]` in o_x and o_y. Jaccard function computes the ratio of intersection over the union of v^i_x and v^i_y.

SQL Structure Similarity To take advantage of the tree-structure of SQL queries, we first parse each SQL query o_x into a SQL tree G_x as illustrated in Figure 3. Given two SQL trees G_x and G_y for SQL queries o_x and o_y, we leverage the Weisfeiler-Lehman sub-tree kernel (Shervashidze et al., 2011) to compute the SQL tree-structure similarity score $f_{\text{tree}}(o_x, o_y)$ as follows:

$$f_{\text{tree}}(o_x, o_y) = \text{Norm}(\mathcal{K}_{\text{WL}}(\mathbf{G}_x, \mathbf{G}_y)) \quad (8)$$

$$\mathcal{K}_{\text{WL}}(\mathbf{G}_x, \mathbf{G}_y) = \sum_{i=0}^{h} \mathcal{K}(\mathbf{G}^i_x, \mathbf{G}^i_y) \quad (9)$$

where Norm() is a normalization function, $\mathcal{K}_{\text{WL}}()$ is the Weisfeiler-Lehman subtree kernel function and $\mathcal{K}()$ is the base kernel on graphs. $\mathbf{G}^i_x$ denotes the Weisfeiler-Lehman graph at height i of the tree $\mathbf{G}_x$ and h is the number of Weisfeiler-Lehman iterations. We refer the readers to Shervashidze et al. (2011) for the implementation details of Weisfeiler-Lehman sub-tree kernel.

Overall, we define the final similarity score of two SQL queries o_x and o_y as follows:

$$\begin{aligned} f_{\text{SQL}}(o_x, o_y) =& \lambda \cdot f_{\text{semantic}}(o_x, o_y) \\ &+ (1-\lambda) \cdot f_{\text{tree}}(o_x, o_y) \end{aligned} \quad (10)$$

where λ is a hyper-parameter controlling the impact of the two kinds of similarity.

Weighted Contrastive Loss After obtaining the SQL similarity, we employ weighted contrastive learning (Oord et al., 2018; He et al., 2022) to pull together two semantically similar NL utterances and push away the representations of semantically dissimilar NL utterances within each conversation. We first convert the input sequence $I_t = [x^1_t, \ldots, x^L_t]$ into a sequence of contextualized vectors $\mathbf{h}_t = [\mathbf{h}^1_t, \ldots, \mathbf{h}^L_t]$, where L repre-

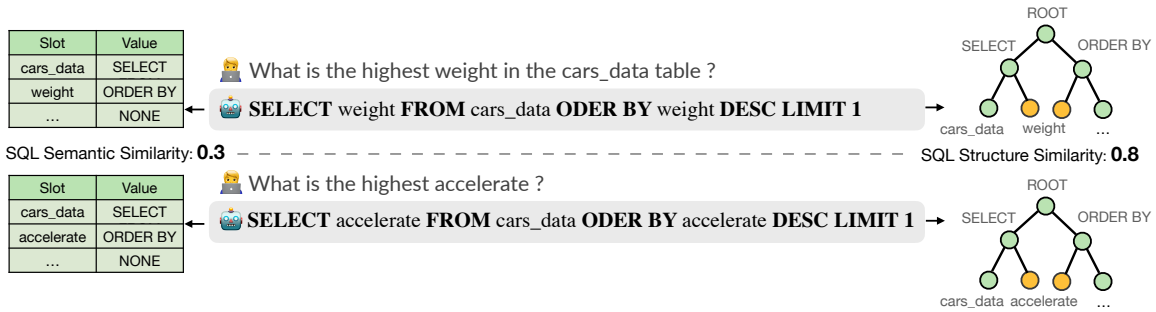


Figure 3: Two metrics for calculating SQL similarity, including semantic similarity and structure similarity.

sents the length of the input sequence. We leverage an attention mechanism to learn the input representation $\tilde{\mathbf{h}}_t$ as:

$$\beta_t^i = \text{Softmax}\left(\tanh(\mathbf{h}_t^i \mathbf{W}_3)\mathbf{v}_3^\top\right) \quad (11)$$

$$\widetilde{\mathbf{h}}_t = \sum_{i=1}^{L} \beta_t^i \mathbf{h}_t^i \quad (12)$$

where $\mathbf{v}_3$ and $\mathbf{W}_3$ are trainable parameters.

Specifically, we minimize a weighted contrastive loss function $\mathcal{L}_{\text{UDT}}$ to optimize the network as:

$$\mathcal{L}_{\text{UDT}} = -\sum_{x\in\mathcal{D}}\sum_{p\in\mathcal{D}_x^+} \frac{f_{\text{SQL}}(o_x, o_p)}{\sum_{k\in\mathcal{D}} f_{\text{SQL}}(o_x, o_k)} \cdot \log \frac{e^{\text{sim}(\tilde{\mathbf{h}}_x, \tilde{\mathbf{h}}_p)/\tau}}{e^{\text{sim}(\tilde{\mathbf{h}}_x, \tilde{\mathbf{h}}_p)/\tau} + \sum_{m\in\mathcal{D}_x^-} e^{\text{sim}(\tilde{\mathbf{h}}_x, \tilde{\mathbf{h}}_m)/\tau}} \quad (13)$$

where τ is a temperature hyper-parameter. $D = \{1, \ldots, N\}$ denotes the index set of the training utterances. $\mathcal{D}_x^+$ denotes the index set of positive utterances that co-occurs in the same conversation with utterance x. $\mathcal{D}_x^-$ denotes the index set of positive utterances other than x and p, and negative utterances chosen from other conversations.

3.3 Masked Language Modeling

In order to jointly learn the contextual representation of utterances and schema, we retain the masking mechanism in the pre-training stage. Concretely, given the input I_t (defined in Eq.1) of the t-th turn, masked language modeling (MLM) selects a random set of positions and replaces these positions with `[MASK]`, and then learns to predict the original tokens of the masked-out tokens. We follow the hyperparameters of prior work (Devlin et al., 2019), which randomly masks utterances and schema tokens with a 15% probability. We denote the MLM loss as $\mathcal{L}_{\text{MLM}}$, which is computed by minimizing the cross-entropy function on the masked tokens.

3.4 Joint Pre-training Objective

In this paper, we combine three pre-training objectives to learn a pre-training framework for context-dependent text-to-SQL parsing. Instead of combining the objectives by simply performing a weighted linear sum of individual losses, we jointly learn three objectives by considering the homoscedastic uncertainty of each objective (Kendall et al., 2018). In this way, we can avoid the huge expense to tune weight hyper-parameters. We define the joint loss function based on homoscedastic uncertainty as:

$$\mathcal{L}_{\text{joint}} = \frac{1}{2\sigma_1^2}\mathcal{L}_{\text{SST}} + \frac{1}{2\sigma_2^2}\mathcal{L}_{\text{UDT}} + \frac{1}{2\sigma_3^2}\mathcal{L}_{\text{MLM}} + \log(1+\sigma_1) + \log(1+\sigma_2) + \log(1+\sigma_3) \quad (14)$$

where $\sigma_1, \sigma_2, \sigma_3$ represent the model's observation noise parameters, capturing how much noise we have in the outputs.

4 Data Construction for Pre-training

The cost of expensive SQL annotation poses a challenge to the construction of large scale pre-training data. Previous work (Yu et al., 2021a,b) resort to data augmentation to address this issue. Typically in a conversational setting, context-dependent data augmentation techniques require two steps: (1) single-turn context-free grammar for utterance-SQL pair generation, and (2) a follow-up context-free grammar to expand single-turn data into context-dependent conversations. SCoRE synthesized a total of 435k text-to-SQL conversations following this setup, and we noticed two limitations with it. Firstly, it relies on the template-filling construction to convert SQL to utterances, resulting in rather rigid generated utterances in step (1). Secondly, SPARC is the only data resource employed to induce the follow-up context-free grammar in step (2). Nevertheless, the contextual diversity in SPARC is insufficient to simulate complex contextual dependencies.

To this end, we propose a new pre-training data construction method. Inspired by the SNOWBALL framework (Shu et al., 2021), we harness a generative model, i.e., BART, to bring more diversity to the generated utterances. For the follow-up conversational context-free grammar induction, we consider both COSQL and SPARC datasets and man-

Model	SPARC				COSQL			
	QM		IM		QM		IM	
	Dev	Test	Dev	Test	Dev	Test	Dev	Test
Previous Parsing Systems.								
GAZP + BERT	48.9	45.9	29.7	23.5	42.0	39.7	12.3	12.8
EditSQL + BERT	47.2	47.9	29.5	25.3	39.9	40.8	12.3	13.7
IGSQL + BERT	50.7	51.2	32.5	29.5	44.1	42.5	15.8	15.0
IST-SQL + BERT	47.6	-	29.9	-	44.4	41.8	14.7	15.2
R^2SQL + BERT	54.1	55.8	35.2	30.8	45.7	46.8	19.5	17.0
DELTA + BART	58.6	59.9	35.6	31.8	51.7	50.8	21.5	19.7
RAT-SQL + SCORE	62.2	62.4	42.5	38.1	52.1	51.6	22.0	21.2
T5-3B + PICARD	-	-	-	-	56.9	54.6	24.2	23.7
HIE-SQL + GRAPPA	64.7	64.6	45.0	42.9	56.4	53.9	28.7	24.6
Pre-trained Models.								
LGESQL	52.4	-	31.3	-	41.2	-	15.0	-
w. BERT	59.8	-	40.5	-	50.7	-	20.8	-
w. ROBERTA	61.6	-	41.2	-	51.9	-	20.8	-
w. GRAPPA	62.5	-	42.4	-	52.6	-	21.5	-
w. SCORE	62.3	-	43.6	-	52.3	-	22.5	-
w. **STAR**	**66.9**	**67.4** (↑ 2.8)	**46.9**	**46.6** (↑ 3.7)	**59.7**	**57.8** (↑ 3.9)	**30.0**	**28.2** (↑ 3.6)

Table 1: Experimental results of various methods in terms of question match (QM) accuracy and interaction match (IM) accuracy on both SPARC and COSQL datasets. "-" means that the test results are not accessible since the test accuracy needs to be officially evaluated and only two models can be submitted every two months.

Model	SPARC		COSQL	
	QM	IM	QM	IM
STAR	**66.9**	**46.9**	**59.7**	**30.0**
w/o MLM	66.1	45.7	59.0	28.7
w/o SST	66.8	45.5	57.9	28.3
w/o UDT	66.4	46.1	58.0	28.7
w/o SST+UDT	65.3	45.6	57.0	27.3

Table 2: Ablation study of STAR in terms of question match accuracy (QM) and interaction match accuracy (IM) on the dev sets of both SPARC and COSQL.

ually craft 100 templates. Overall, we synthesize a new large-scale pre-training dataset that consists of about 480K high-quality context-dependent text-to-SQL conversations. We provide examples of the induced grammar rules and synthesized procedure in detail in Appendix D.

5 Experiment

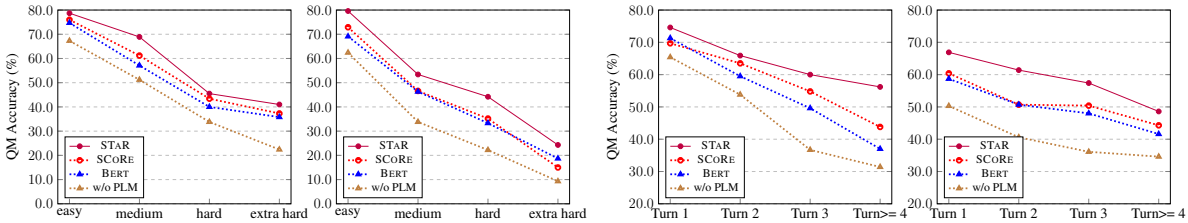
5.1 Experimental Setup

Downstream Datasets We evaluate STAR on two context-dependent semantic parsing benchmarks: SPARC (Yu et al., 2019b) and COSQL (Yu et al., 2019a). SPARC is a collection of cross-domain context-dependent dataset, which consists of about 4.3k question sequences and 12k+ individual questions annotated with SQL queries. COSQL is a conversational text-to-SQL corpus, which contains about 3k dialogues and 10k+ annotated SQL queries. Both SPARC and COSQL query 200 complex databases spanning across 138 domains. We provide more detailed statistics of these two datasets in Appendix B.

Evaluation Metrics We employ two official evaluation metrics (Yu et al., 2019b,a) to verify the effectiveness of STAR: question match accuracy (QM) and interaction match accuracy (IM). Concretely, QM denotes the exact set match accuracy over SQL templates and IM denotes the ratio of interactions over all correctly predicted questions.

Implementation Details In pre-training, STAR is initialized with ELECTRA (Clark et al., 2020). Similar to ELECTRA, we also employ the replaced token detection objective to further improve the text-to-SQL pre-training. The maximum length of each input sequence is set to 256. The batch size is set to 80 and an Adam optimizer is employed for optimization with an initial learning rate of 1e-6. Gradient clipping is applied to STAR with a maximum gradient value of 1. For computing the SQL similarity, the impact factor λ is set to 0.5. We provide more details of implementation in Appendix A.

Baselines First, we compare STAR with several state-of-the-art context-dependent parsing methods, including **GAZP** (Zhong et al., 2020), **EditSQL** (Zhang et al., 2019), **IGSQL** (Cai and Wan, 2020), **IST-SQL** (Wang et al., 2021a), **R^2SQL** (Hui et al., 2021), **PICARD** (Scholak et al., 2021), **DELTA** (Chen et al., 2021) and **HIE-SQL** (Zheng et al., 2022). Second, we compare STAR with four strong pre-training models, including **BERT** (Devlin et al., 2019), **ROBERTA** (Liu et al., 2019), **GRAPPA** (Yu



(a) SPARC (b) COSQL (c) SPARC (d) COSQL

Figure 4: The results of STAR and baselines on SPARC and COSQL dev sets (a-b) by varying the difficulty levels of the data and (c-d) by varying the conversation turns.

Model	COSQL		SPARC	
	QM	IM	QM	IM
STAR	59.7	30.0	66.9	46.9
STAR w/o structural	59.1	29.0	66.5	46.7
STAR w/o semantic	59.5	29.6	66.8	46.5
STAR w/o UDT	58.0	28.6	66.4	46.1

Table 3: Results of STAR on the dev sets of SPARC and COSQL by using different metrics for calculating SQL similarity.

Model	COSQL	
	QM	IM
STAR (w/ MLM) + SCORE data	55.4	25.6
STAR (w/ MLM) + Our data	57.0	27.3
STAR (w/ MLM + SST) + SCORE data	57.3	27.3
STAR (w/ MLM + SST) + Our data	58.0	28.7

Table 4: Results of STAR on the dev set of COSQL with MLM and SST objectives by using different pre-training data.

et al., 2021a) and **SCORE** (Yu et al., 2021b). In particular, GRAPPA and SCORE are the representative TaLMs for context-independent and context-dependent text-to-SQL parsing, respectively.

5.2 Model Comparison on Downstream Tasks

In the experiments, we choose LGESQL (Cao et al., 2021) as our base model given its superior performance. Since LGESQL is originally developed for single-turn setting, we extend LGESQL to context-dependent setting by taking as input the concatenation of historical and current utterances. For a fair comparison, the four compared PLMs also leverage LGESQL as the base model.

The experimental results on SPARC and COSQL are summarized in Table 1. STAR outperforms all the compared methods on the two datasets by a noticeable margin. First, STAR achieves substantially better results than the four strong PLMs. In particular, STAR surpasses the well-known SCORE by 7.4% QM score and 7.5% IM score on the COSQL dev set. Second, LGESQL+STAR achieves better results than the compared downstream methods which use BERT, ROBERTA, SCORE, GRAPPA as the PLMs, such as the best performing baseline HIE-SQL+GRAPPA.

5.3 Ablation Study

Effectiveness of Pre-training Objectives We conduct ablation test to investigate the effectiveness of each pre-training objective in STAR. We report the results of removing the MLM loss (called w/o MLM), the SST loss (called w/o SST), the UDT loss (called w/o UDT), and both SST and UDT (called w/o SST+UDT) respectively. Table 2 shows the ablation test results on both SPARC and COSQL. We can observe that removing the SST or UDT objective bring the most significant performance drop. Not surprisingly, combining all the three objectives achieves the best results on both datasets.

Effectiveness of SQL Similarity Metrics To analyze the impact of metrics for calculating the SQL similarity in STAR, we also conduct an ablation test by removing the structural similarity metric (called w/o structural), the semantic similarity metric (called w/o semantic), and both (called w/o UDT), respectively. Table 3 shows the ablation test results on the dev sets of SPARC and COSQL. As expected, both similarity metrics contribute great improvements to STAR.

Effectiveness of Synthesized Pre-training Data We also analyze the quality of our constructed pre-training data. We compare our pre-training data with the data created by SCORE (Yu et al., 2021b) which to our knowledge is the only existing work on pre-training for context-dependent text-to-SQL parsing. Since the pre-training data created by SCORE is inapplicable to the $\mathcal{L}_{\text{UDT}}$ objective, we merely employ $\mathcal{L}_{\text{MLM}}$ (denoted as STAR w/ MLM) and $\mathcal{L}_{\text{MLM}} + \mathcal{L}_{\text{SST}}$ (denoted as STAR w/ MLM + SST) as the pre-training objectives in the experiments. As shown in Table 4, our pre-training data is more effective than the pre-training data created by SCORE.

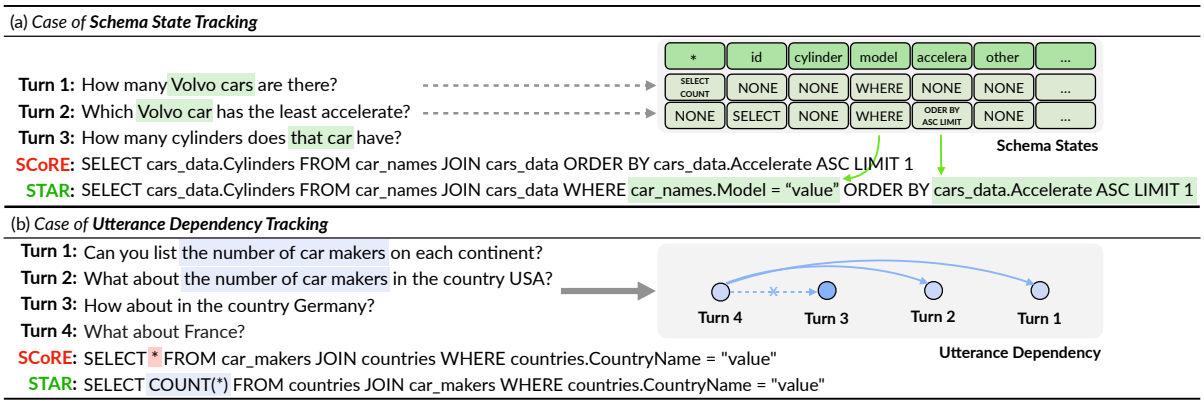


Figure 5: Two cases on the CoSQL dev dataset.

5.4 Discussion

Model Comparison on Samples with Different Levels of Difficulty The SQL queries in both SPARC and COSQL can be further divided into four levels based on the difficulty of the SQL queries: easy, medium, hard, extra hard, which can be used to better evaluate the model performance on different queries. As shown in Figure 4a-b, STAR achieves better results than the compared methods on the four kinds of data, even on the extra hard samples.

Model Comparison on Samples at Different Turns Figure 4c-d illustrate the QM results of STAR and compared methods along with the increase of conversation turns on SPARC and COSQL dev sets. The QM results of baselines decrease sharply as the conversation turns increase, while STAR achieves much more stable performance even for the third and fourth turns. This suggests that STAR can better track and explore the interaction states in history utterances to assist the models to better parse current utterance.

5.5 Case Study

To evaluate STAR qualitatively, we choose two exemplary conversations from the CoSQL dev set and illustrate the generated SQL queries by SCORE and STAR in Figure 5. In the first case, we observe that STAR can exploit the usage of table information in history queries (e.g., `[car_names.Model]` to correctly generate the third SQL query, while SCORE fails to track this kind of schema state. In the second case, STAR successfully tracks the long-term utterance dependency between the first and fourth utterances, and generates the correct SQL keyword `[SELECT COUNT(*)]` in the fourth SQL query by tracking and referring to the query "the number of" in the second utterance. However, SCORE fails to track such long-term dependency with being disturbed by the third utterance.

5.6 Limitation Analysis

To better analyze the limitations of STAR, we carry out an analysis of the errors made by STAR on the CoSQL dev dataset. We reveal several reasons of the errors, which can be divided into following categories. **First**, STAR fails to select the correct names from table schemas in some hard or extra hard samples, where NL questions use synonyms to refer to tables or columns in SQL queries without the explicit correspondence between NL questions and table schemas. One possible solution is to exploit the rich semantic information contained in PLMs to capture the implicit schema linking information via knowledge probing techniques. **Second**, for some samples, STAR incorrectly inherits part of the previous turn SQL query. One possible solution is to design an additional classifier to predict the changes (e.g. RETAIN, MODIFY, DELETE) between the schema state of the current turn and that of the previous turn. **Third**, there are some SQL grammar errors such as the redundancy of `[WHERE]` clause, repetition of table names, structure error of `[SELECT NEST]`. The reason may be that the schema state tracking objective only tracks the state of the database schema in conversation, which do not consider the overall grammatical structure of SQL queries. One possible idea is to add an extra objective to predict the general structure of SQL (e.g., abstract syntax tree) so as to capture the overall grammatical structure information of SQL.

6 Related Work

Context-dependent Text-to-SQL Parsing Most of previous text-to-SQL works focused on the context-independent setting. Notably, the graph-

based parser, *e.g.*, RAT-SQL (Wang et al., 2020), LGESQL (Cao et al., 2021) , S^2SQL (Hui et al., 2022), and the T5-based parser, *e.g.*, PICARD (Scholak et al., 2021), achieving the impressive performance on SPIDER (Yu et al., 2018). In recent years, context-dependent (multi-turn) text-to-SQL parsing has attracted increasing attention due to its broad applications and realistic setting. SPARC (Yu et al., 2019b) and COSQL (Yu et al., 2019a) are two benchmark datasets for context-dependent text-to-SQL parsing. Subsequently, several works (Zhang et al., 2019; Cai and Wan, 2020; Wang et al., 2021a,b; Hui et al., 2021; Chen et al., 2021; Zheng et al., 2022) were proposed, which consider contextual information or conversation history so as to synthesise the correct SQL query. In particular, Zhang et al. (2019) exploited the conversation history by editing the previous predicted SQL to improve the generation quality. The schema interaction graph in IGSQL (Cai and Wan, 2020) and two kinds of interaction states in IST-SQL (Wang et al., 2021a) are designed to capture the historical schema evolution in context. Furthermore, Zheng et al. (2022) improve contextual accuracy by incorporating additional SQL encoders to integrate historical SQL into the input. In contrast to above works, STAR focus on the pre-training stage, expecting to extract general knowledge from large-scale unsupervised or self-supervised data that will be useful for downstream parsing tasks.

Pre-training Models for Text-to-SQL Parsing In parallel, tabular language models (TaLMs) have been proposed to simultaneously encode tables and texts, which further improved the results of downstream text-to-SQL parsing tasks. For example, TABERT (Yin et al., 2020) and TAPAS (Herzig et al., 2020) jointly encoded texts and tables with self-supervised or weakly-supervised objectives, which was trained on a large corpus of tables. STRUG (Deng et al., 2021) proposed a structured-grounded pre-training technique and GAP (Shi et al., 2021) introduced a generation-augmented pre-training framework to capture the alignment relationship of utterance and table. Similarly, GRAPPA (Yu et al., 2021a) introduced a grammar-augmented pre-training framework for text-to-SQL parsing, which explored the schema linking by encouraging the model to identify table schema components that could be grounded to logical form constituents. SCORE (Yu et al., 2021b) was the state-of-the-art pre-training approach for context-dependent text-to-SQL parsing designed to induce representations that captured the switch between the adjacency turns. Unlike these TaLMs, STAR is the first to leverage both historical SQL and complex utterance dependency in the pre-training stage.

7 Conclusion

In this paper, we proposed STAR, a pre-trained TaLM, which could jointly learn user utterance and table schema representations for context-dependent text-to-SQL conversations. STAR contained two novel pre-training objectives (schema state tracking and utterance dependency tracking) to explore the complex context interactions of NL utterances and SQL queries within each text-to-SQL conversation, respectively. We constructed a diverse large-scale context-dependent text-to-SQL conversation corpus to pre-train STAR. Experiments demonstrated that STAR achieves new state-of-the-art performance on SPARC and COSQL.

Acknowledgements

My acknowledges: This work was partially supported by National Natural Science Foundation of China (No. 61906185), Youth Innovation Promotion Association of CAS China (No. 2020357), Shenzhen Science and Technology Innovation Program (Grant No. KQTD20190929172835662), Shenzhen Basic Research Foundation (No. JCYJ20210324115614039 and No. JCYJ20200109113441941). This work was supported by Alibaba Group through Alibaba Innovative Research Program.

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, Online. Association for Computational Linguistics.

Zhi Chen, Lu Chen, Hanqi Li, Ruisheng Cao, Da Ma, Mengyue Wu, and Kai Yu. 2021. Decoupled dialogue modeling and semantic parsing for multi-turn text-to-SQL. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3063–3074, Online. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pre-training text encoders as discriminators rather than generators. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

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.

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.

Wanwei He, Yinpei Dai, Binyuan Hui, Min Yang, Zhen Cao, Jianbo Dong, Fei Huang, Luo Si, and Yongbin Li. 2022. Space-2: Tree-structured semi-supervised contrastive pre-training for task-oriented dialog understanding. In *COLING*.

Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Eisenschlos. 2020. TaPas: Weakly supervised table parsing via pre-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4320–4333, Online. Association for Computational Linguistics.

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 exploration network for cross-domain context-dependent semantic parsing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13116–13124.

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

Alex Kendall, Yarin Gal, and Roberto Cipolla. 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 7482–7491. IEEE Computer Society.

Qian Liu, Bei Chen, Jiaqi Guo, Zeqi Lin, and Jian-Guang Lou. 2021a. TAPEX: table pre-training via learning a neural SQL executor. *CoRR*, abs/2107.07653.

Qian Liu, Dejian Yang, Jiahui Zhang, Jiaqi Guo, Bin Zhou, and Jian-Guang Lou. 2021b. Awakening latent grounding from pretrained language models for semantic parsing. In *ACL*.

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*, abs/1907.11692.

Suphakit Niwattanakul, Jatsada Singthongchai, Ekkachai Naenudorn, and Supachanun Wanapu. 2013. Using of jaccard coefficient for keywords similarity. In *Proceedings of the international multiconference of engineers and computer scientists*, volume 1, pages 380–384.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *ArXiv preprint*, abs/1807.03748.

Yawen Ouyang, Moxin Chen, Xinyu Dai, Yinggong Zhao, Shujian Huang, and Jiajun Chen. 2020. Dialogue state tracking with explicit slot connection modeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 34–40, Online. Association for Computational Linguistics.

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

Bowen Qin, Lihan Wang, Binyuan Hui, Bowen Li, Xiangpeng Wei, Binhua Li, Fei Huang, Luo Si, Min Yang, and Yongbin Li. 2022b. Sun: Exploring intrinsic uncertainties in text-to-sql parsers. *arXiv preprint arXiv:2209.06442*.

Torsten Scholak, Nathan Schucher, and Dzmitry Bahdanau. 2021. PICARD: Parsing incrementally for constrained auto-regressive decoding from language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9895–9901, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Nino Shervashidze, Pascal Schweitzer, Erik Jan Van Leeuwen, Kurt Mehlhorn, and Karsten M Borgwardt. 2011. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research*, 12(9).

Peng Shi, Patrick Ng, Zhi guo Wang, Henghui Zhu, Alexander Hanbo Li, J. Wang, C. D. Santos, and Bing Xiang. 2021. Learning contextual representations for semantic parsing with generation-augmented pre-training. In *AAAI*.

Chang Shu, Yusen Zhang, Xiangyu Dong, Peng Shi, Tao Yu, and Rui Zhang. 2021. Logic-consistency text generation from semantic parses. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4414–4426, Online. Association for Computational Linguistics.

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.

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.

Run-Ze Wang, Zhen-Hua Ling, Jingbo Zhou, and Yu Hu. 2021a. Tracking interaction states for multi-turn text-to-sql semantic parsing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13979–13987.

Runze Wang, Zhenhua Ling, Jing-Bo Zhou, and Yu Hu. 2021b. A multiple-integration encoder for multi-turn text-to-sql semantic parsing. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:1503–1513.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv preprint*, abs/1910.03771.

Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. 2020. TaBERT: Pretraining for joint understanding of textual and tabular data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8413–8426, Online. Association for Computational Linguistics.

Tao Yu, Chien-Sheng Wu, Xi Victoria Lin, Bailin Wang, Yi Chern Tan, Xinyi Yang, Dragomir R. Radev, Richard Socher, and Caiming Xiong. 2021a. Grappa: Grammar-augmented pre-training for table semantic parsing. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Tao Yu, Rui Zhang, Heyang Er, Suyi Li, Eric Xue, Bo Pang, Xi Victoria Lin, Yi Chern Tan, Tianze Shi, Zihan Li, Youxuan Jiang, Michihiro Yasunaga, Sungrok Shim, Tao Chen, Alexander Fabbri, Zifan Li, Luyao Chen, Yuwen Zhang, Shreya Dixit, Vincent Zhang, Caiming Xiong, Richard Socher, Walter Lasecki, and Dragomir Radev. 2019a. CoSQL: A conversational text-to-SQL challenge towards cross-domain natural language interfaces to databases. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1962–1979, Hong Kong, China. Association for Computational Linguistics.

Tao Yu, Rui Zhang, Alex Polozov, Christopher Meek, and Ahmed Hassan Awadallah. 2021b. Score: Pre-training for context representation in conversational semantic parsing. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

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.

Tao Yu, Rui Zhang, Michihiro Yasunaga, Yi Chern Tan, Xi Victoria Lin, Suyi Li, Heyang Er, Irene Li, Bo Pang, Tao Chen, Emily Ji, Shreya Dixit, David Proctor, Sungrok Shim, Jonathan Kraft, Vincent Zhang, Caiming Xiong, Richard Socher, and Dragomir Radev. 2019b. SParC: Cross-domain semantic parsing in context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4511–4523, Florence, Italy. Association for Computational Linguistics.

Rui Zhang, Tao Yu, Heyang Er, Sungrok Shim, Eric Xue, Xi Victoria Lin, Tianze Shi, Caiming Xiong, Richard Socher, and Dragomir Radev. 2019. Editing-based SQL query generation for cross-domain context-dependent questions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5338–5349, Hong Kong, China. Association for Computational Linguistics.

Yanzhao Zheng, Haibin Wang, Baohua Dong, Xingjun Wang, and Changshan Li. 2022. Hie-sql: History information enhanced network for context-dependent text-to-sql semantic parsing. *ArXiv preprint*, abs/2203.07376.

Victor Zhong, Mike Lewis, Sida I. Wang, and Luke Zettlemoyer. 2020. Grounded adaptation for zero-shot executable semantic parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6869–6882, Online. Association for Computational Linguistics.

Victor Zhong, Caiming Xiong, and R. Socher. 2017. Seq2sql: Generating structured queries from natural language using reinforcement learning. *CoRR*, abs/1709.00103.

A More Implementation Details

In the pre-training, STAR is initialized with ELECTRA (Clark et al., 2020). Similar to ELECTRA which is consist of a generator $\mathcal{G}$ and a discriminator $\mathcal{D}$, we also employ the replaced token detection objective to further improve the text-to-SQL pre-training. Concretely, given the input I_t (defined in Eq. (1)) of the t-th turn, the generator with masked language modeling (MLM) selects a random set of positions and replaces these positions with `[MASK]`, and then learns to predict the original tokens of the masked-out tokens. The chance of each token being masked out is 15%. We denote the loss function of the generator as $\mathcal{L}_{\text{MLM}}$. In addition, we also train the discriminator to predict whether the each token is the same as the original token. We denote the loss function for training the discriminator as $\mathcal{L}_{\text{Dis}}$. Finally, we combine the loss functions of the generator $\mathcal{G}$ and the discriminator $\mathcal{D}$ to form the overall objective function for replaced token detection (RTD) as:

$$\mathcal{L}_{\text{overall}} = \mathcal{L}_{\text{Dis}} + \gamma \mathcal{L}_{\text{MLM}} \quad (15)$$

We refer the readers to (Clark et al., 2020) for the implementation details of the RTD objective. γ is a hyperparameter controlling the impact of $\mathcal{L}_{\text{MLM}}$. In this work, the impact factor γ is set to 5. Our codebase is built on huggingface library (Wolf et al., 2019).

We use LGESQL as our downstream model. For a fair comparison, all LGESQL experiments are trained for 100 epoch. The learning rate is 1e-4 and weight decay is 0.1. And we adopt a more carefully optimization for our STAR encoder with layer-wise learning rate decay coefficient 0.8. Batch size is 10 and the maximum gradient norm is 5. Other hyperparameters are the same as in (Cao et al., 2021).

B Details of SPARC and COSQL

We evaluate the effectiveness of STAR on two context-dependent text-to-SQL parsing benchmarks: SPARC (Yu et al., 2019b) and COSQL (Yu et al., 2019a). Concretely, SPARC is a collection of cross-domain context-dependent dataset, which consists of about 4.3k question sequences and 12k+ individual questions annotated with SQL queries. COSQL is a conversational text-to-SQL corpus, which contains about 3k dialogues and 10k+ annotated SQL queries. Both SPARC and COSQL query 200 complex databases spanning across 138 domains. Table 5 reports the statistics of SPARC and COSQL datasets in detail.

	SPARC	COSQL
# Question Sequences	4,298	3,007
# Train	3,034	2,164
# Dev	422	293
# Test	842	551
# User Questions	12,726	15,598
# Databases	200	200
# Domain	138	138
Avg.len	8.1	11.2
Vocab	3,794	9,585
System Response	no	yes

Table 5: Details of SParC and CoSQL Dataset.

Model	SPARC		COSQL	
	QM	IM	QM	IM
ROBERTA	61.6	41.2	51.9	20.8
STAR (init. with ROBERTA)	**65.0**	**45.1**	**54.1**	**25.3**

Table 6: Results of STAR which is initialized with ROBERTA on the dev sets of both SPARC and COSQL.

C Generalization of STAR

We also evaluate the generalization of our pre-training objectives by using ROBERTA as our initialization model, rather than applying ELECTRA. The experimental results are shown in Table 6. In a similar trend, STAR that is initialized with ROBERTA performs significantly better than the original ROBERTA, which to some extent verifies the generalization of the proposed pre-training objectives, no matter what initialization models are used to train STAR.

D Details of Data Construction

In this paper, we synthesize a new large-scale pre-training dataset which consists of about 480K high-quality context-dependent text-to-SQL conversations. Specifically, we first generate single-turn question-SQL pairs by exploiting the SPIDER, SPARC and COSQL datasets.

D.1 Single-turn Question-SQL Pairs

To obtain sufficient high-quality single-turn question-SQL pairs, we carefully examine currently available sources and generate question-SQL pairs from SPIDER, SPARC and COSQL datasets. Specifically, we collect the original single-turn

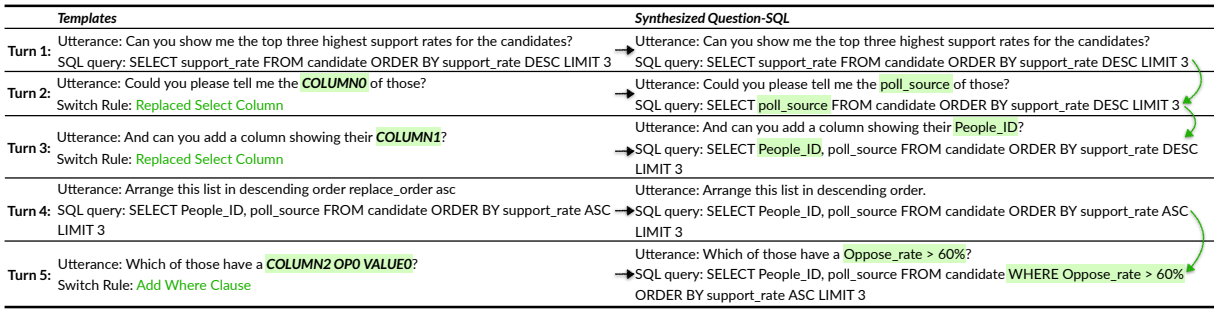


Figure 6: An example of synthetic text-to-SQL conversation.

question-SQL pairs from the dataset SPIDER which is one of the largest single-turn cross-domain text-to-SQL corpora. For the context-dependent text-to-SQL datasets SPARC and COSQL, we generate a new question for each SQL query instead of using the original NL questions since they may contain ellipsis and anaphora that refers to earlier items in the conversations, resulting in low-quality question-SQL pairs. In particular, we employ the SNOWBALL framework (Shu et al., 2021) with BART to generate the question based on each SQL query, which employs an iterative training procedure by recursively augmenting the training set with quality control.

D.2 Context-dependent Text-to-SQL Conversations

To expand the single-turn question-SQL pairs to context-dependent text-to-SQL conversations, we first convert SQL queries into their structured formats. For example, we convert the SQL query "*SELECT support_rate FROM candidate ORDER_BY support_rate LIMIT 3*" into a set of SQL states as {`SELECT`: `[support_rate]`, `FROM`: `[candidate]`, `ORDER_BY`: `[support_rate]`, other SQL keywords: `[NONE]`}.

Then, following (Yu et al., 2021b), we study 600 examples from the training set of both SPARC and CoSQL datasets, and induce about 100 follow-up question-grammar templates. Each template consists of a pair of (i) a context-free question template (e.g., "*Could you please tell me the* `[COLUMN0]` *of those?*") where the typed slot `[COLUMN0]` represents the mention of schema, and (ii) its corresponding operation grammar (e.g., "*replaced select column*") that contains context switch labels of the question templates.

Finally, for a single-turn question-SQL pair constructed in Section D.1 with database d, we randomly choose a created question-grammar template. We sample the values for typed slots in the template and get the synthesized NL question as well as its corresponding SQL query if the previous SQL query satisfies the constraints in the sampled template (e.g., the SQL query contains the mentioned schema); otherwise, another question-grammar temple is sampled until we successfully synthesize the next question-SQL pair. Then, we consider the synthesized question-SQL pair as a new start and repeat the above process until we obtain the context-dependent text-to-SQL conversation consisting of T turns of question-SQL pairs. Figure 6 shows an example of synthetic text-to-SQL conversation with five turns.