HIE-SQL: History Information Enhanced Network for Context-Dependent Text-to-SQL Semantic Parsing

Anonymous ACL submission

Abstract

Recently, context-dependent text-to-SQL semantic parsing which translates natural language into SQL in an interaction process has attracted a lot of attention. Previous works leverage context-dependence information either from interaction history utterances or the previous predicted SQL queries but fail in taking advantage of both since of the mismatch between natural language and logic-form SQL. In this work, we propose a **H**istory **I**nformation **E**nhanced text-to-**SQL** model (**HIE-SQL**) to exploit context-dependence information from both history utterances and the last predicted SQL query. In view of the mismatch, we treat natural language and SQL as two modalities and propose a bimodal pre-trained model to bridge the gap between them. Besides, we design a schema-linking graph to enhance connections from utterances and the SQL query to the database schema. We show our history information enhanced methods improve the performance of HIE-SQL by a significant margin, which achieves new state-of-the-art results on the two context-dependent text-to-SQL benchmarks, the SparC and CoSQL datasets, at the writing time.

U_1: List the name of the teachers and the courses assigned for them to teach.

S_1 : SELECT T3.Name, T2.Course FROM course_arrange AS T1
JOIN course AS T2 ON T1.Course_ID = T2.Course_ID
JOIN teacher AS T3 ON T1.Teacher_ID = T3.Teacher_ID

U_2: Arrange this list with the teachers name in ascending order.

S_2 : SELECT T3.Name, T2.Course FROM course_arrange AS T1
JOIN course AS T2 ON T1.Course_ID = T2.Course_ID
JOIN teacher AS T3 ON T1.Teacher_ID = T3.Teacher_ID
ORDER BY T3.Name

U_3: Include teachers id in the same list.

S_3 : SELECT T3.Name, T2.Course, T1.teacher_ID FROM course_arrange AS T1
JOIN course AS T2 ON T1.Course_ID = T2.Course_ID
JOIN teacher AS T3 ON T1.Teacher_ID = T3.Teacher_ID
ORDER BY T3.Name

Figure 1: An example of context-dependent text-to-SQL interaction in CoSQL where U_i is the utterance of turn i and S_i is the corresponding SQL query for U_i. The tokens with red color are the history information that should be considered in later predictions. It is context-independent if we just consider the prediction of S_1.

1 Introduction

Conversation user interfaces to databases have launched a new research hotspot in Text-to-SQL semantic parsing (Zhang et al., 2019; Guo et al., 2019; Wang et al., 2020; Lin et al., 2020; Xu et al., 2021; Cao et al., 2021; Hui et al., 2021; Yu et al., 2021b) and benefited us in industry (Dhamdhere et al., 2017; Weir et al., 2020). Most previous works focus on the context-independent text-to-SQL task and propose many competitive models. Some models (Wang et al., 2020; Scholak et al., 2021) even surprisingly work well on the context-dependent text-to-SQL task by just appending the interaction history utterances to the input. Especially, PICARD (Scholak et al., 2021) achieves state-of-the-art performances both in Spider (Yu et al., 2018b), a cross-domain context-independent text-to-SQL benchmark, and CoSQL (Yu et al., 2019a), a cross-domain context-dependent text-to-SQL benchmark, before our work. However, every coin has two sides. That implies underachievement of the exploration of context information in context-dependent text-to-SQL semantic parsing.

Compared with context-independent text-to-SQL semantic parsing, context-dependent text-to-SQL semantic parsing are more challenging since of the various types of dependence in utterances which make models vulnerable to parsing errors. As R^2SQL (Hui et al., 2021) considers, different context dependencies between two adjacent utterances require the model to establish dynamic connections between utterances and database schema carefully. However, context information is not only from the last utterance. Long-range dependence is also the case in CoSQL as the prediction of S_3 depends on "the name of the teachers and the courses" in U_1 in Figure 1. A workable proposition for long-range dependence is to inherit context information from previous predicted SQL queries. But it is not a piece of cake to take advantage of previously predicted queries since of the mismatch between natural language and logic-form SQL. As Liu et al. (2020) conclude, roughly encoding the last predicted SQL query and utterances takes the wooden spoon while easily concatenation of interaction history utterances and current utterance appears to be strikingly competitive in their evaluation of 13 existing context modeling methods.

In this paper, we propose a history information enhanced network to make full use of both history interactive utterances and previous predicted SQL queries. We first encode the last predicted SQL query by treating the logic-form query as another modality with natural language. We present SQL-BERT, a bimodal pre-trained model for SQL and natural language which is able to capture the semantic connection and bridge the gap between SQL and natural language. It produces general-purpose representations and supports our context-dependent text-to-SQL semantic parsing. As adopted in a majority of large pre-trained models, we develop SQL-BERT with the multi-layer Transformer (Vaswani et al., 2017).We pre-train it with the objective function of masked language modeling (MLM) on SQL.

Besides, we propose a history information enhanced schema-linking graph to represent the relations among current utterance, interaction history utterances, the last predicted query, and corresponding database schema. Considering it is weird to shift a topic back and forth in an interaction, we assume that the long-range dependence is successive. For example, that S_3 depends on U_1 implies that S_2 does too in Figure 1. In that case, we can leverage the long-range dependence from the last predicted query. In addition, the corresponding SQL queries of adjacent utterances tend to overlap (Zhang et al., 2019). Therefore, unlike the previous schema-linking graph just with utterances and database schema (Hui et al., 2021), the last predicted query takes part in our graph. Besides, we distinguish current utterance and interaction history utterances in the schema-linking graph. We encode the schema-linking relations with Relative Self-Attention Mechanism (Shaw et al., 2018).

In our experiments, the proposed methods of SQLBERT and the history information enhanced schema-linking substantially improve the performance of our model. At the time of writing, our model ranks first on both two large-scale cross-domain context-dependent text-to-SQL leaderboards, SparC (Yu et al., 2019b) and CoSQL (Yu et al., 2019a). Specifically, our model achieves a 64.6% question match and 42.9% interaction match accuracy on SparC, and a 53.9% question match and 24.6% interaction match accuracy on CoSQL.

2 Related Work

Text-to-SQL semantic parsing follows a long line of research on semantic parsing from natural language to logical language (Zelle and Mooney, 1996; Zettlemoyer and Collins, 2005; Wong and Mooney, 2007).

Recently, context-independent text-to-SQL semantic parsing has been well studied. Spider (Yu et al., 2018b) is a famous dataset for the complex and cross-domain context-independent text-to-SQL task. Some works (Bogin et al., 2019a,b; Chen et al., 2021) apply graph neural networks to encode database schema. Xu et al. (2021) succeed in appling deep transformers to the context-independent text-to-SQL task. Yu et al. (2018a) employ a tree-based decoder to match SQL grammar. Rubin and Berant (2021) improve the tree-based decoder by a bottom-up method. Scholak et al. (2021) refine the sequence-based decoder via carefully designed restriction rules. Guo et al. (2019) and Gan et al. (2021) propose SQL intermediate representations to bridge the gap between natural language and SQL. Lei et al. (2020) study the role of schema-linking in text-to-SQL semantic parsing. Wang et al. (2020) propose a unified framework to capture the schema-linking. Lin et al. (2020) represent the schema-linking as a tagged sequence. Cao et al. (2021) further integrate non-local and local features via taking advantage of both schema-linking graph and its corresponding line graph. Besides, many previous works (Deng et al., 2021; Yu et al., 2021a; Shi et al., 2021) focus on pre-train models for context-independent text-to-SQL semantic parsing.

With more attentions on context-dependent text-to-SQL semantic parsing, existing works have been devoted to the context-dependent text-to-SQL task. SparC (Yu et al., 2019b) and CoSQL (Yu et al., 2019a) datasets are specially proposed for the task. EditSQL (Zhang et al., 2019) and IST-SQL (Wang et al., 2021) focus on taking advantages of the last predicted query for the prediction of current query. EditSQL tries to copy the overlap tokens

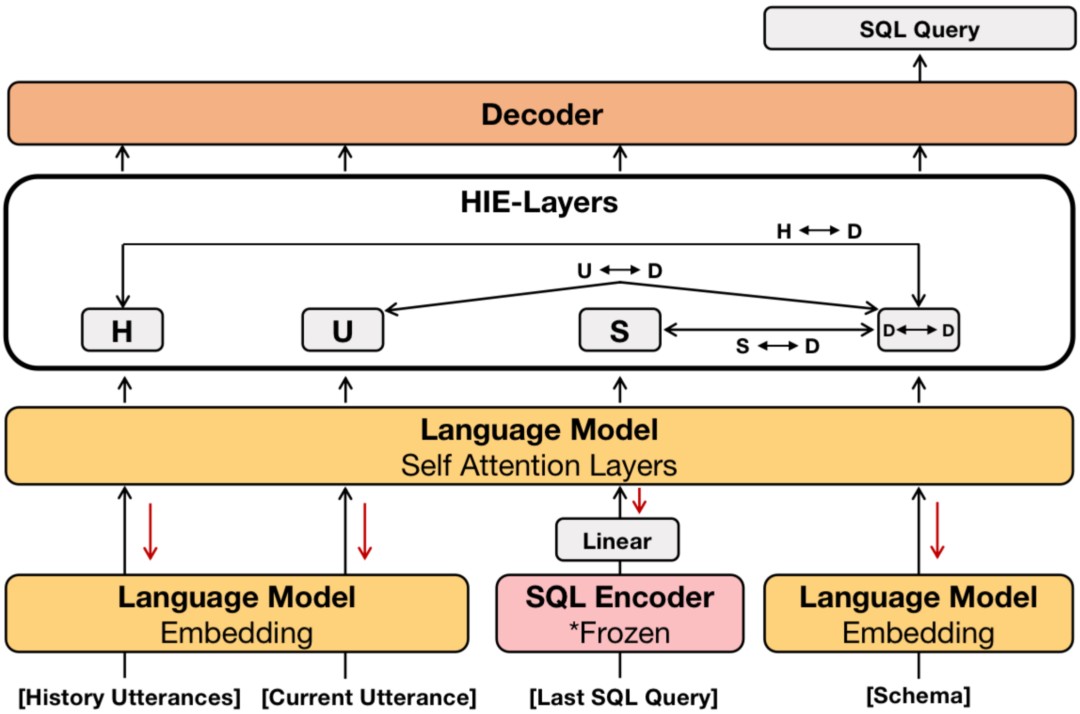

Figure 2: Structure and components of HIE-SQL. During the training stage, the parameters of SQL Encoder will not be updated.

from the last predicted query, while IST-SQL proposes an interaction state tracking method to encode the information from the last predicted query. IGSQL (Cai and Wan, 2020) and R^2SQL (Hui et al., 2021) leverages the contextual information among the current utterance, interaction history utterances and database schema via context-aware dynamic graphs. Notably, R^2SQL simulates the information by connecting the schema graphs with the tokens in interactive utterances. Yu et al. (2021b) creatively propose a context-aware pre-trained language model. However, the problem of making full use of both interaction history utterances and predicted queries for the context-dependent text-to-SQL task remains open.

3 HIE-SQL

First, we formally define the conversational text-to-SQL semantic parsing problem. In the rest of the section, we detail the architecture of history information enhanced text-to-SQL model (HIE-SQL).

3.1 Preliminaries

Task Definition. Given the current user utterance u_τ, interaction history $h_\tau = [u_1, u_2, ..., u_{\tau-1}]$, the schema $D = \langle T, C \rangle$ of the target database such that the set of tables $T = \{t_1, ..., t_{|T|}\}$ and the set of columns $C = \{c_1, ..., c_{|C|}\}$, our goal is to generate the corresponding SQL query s_τ.

Model Architecture. Figure 2 shows the encoder-decoder framework of HIE-SQL. We will introduce it in four modules: (i) **Multimodal Encoder**, which encodes SQL query and natural language context in a multimodal manner, (ii) **SQLBERT**, a bimodal pre-trained encoder for SQL and natural language, (iii) **HIE-Layers**, which encode pre-defined schema-linking relations between all elements of the output of Language Model, and (iv) **Decoder**, which generates SQL query as an abstract syntax tree.

3.2 Multimodal Encoder

Since of the huge syntax structure differences between SQL and natural language, using a single language model to encode both languages at the same time increases the difficulty and cost of training the model. Inspired by the efficiency of the works (Kiela et al., 2019; Tsimpoukelli et al., 2021) to solve the multimodal problems, we build an additional pre-trained Encoder named SQLBERT (we will detail it in the following section) to pre-encode SQL query. Then we learn weights $W \in R^{N \times M}$ to project the N-dimensional SQL query embeddings to M-dimensional token input embedding space of

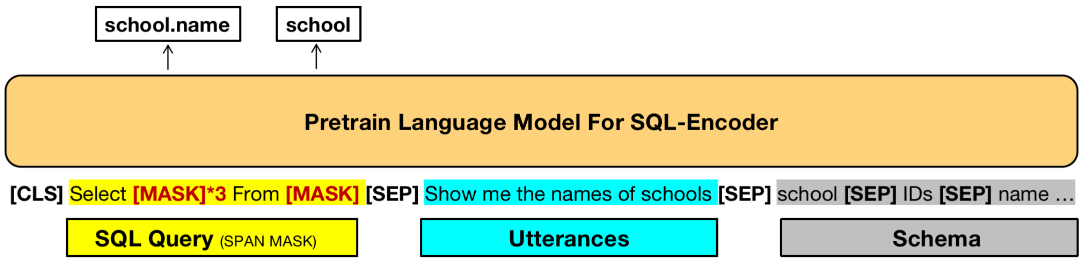

Figure 3: Input format and training objective of SQLBERT.

the language model:

$$\mathcal{S} = Wf(s_{\tau-1}), \tag{1}$$

where $f(\cdot)$ is the last hidden state output of SQLBERT.

We arrange the input format of HIE-SQL as $x = (\texttt{[CLS]}, \mathcal{U}, \texttt{[CLS]}, \mathcal{S}, \texttt{[SEP]}, \mathcal{T}, \texttt{[SEP]}, \mathcal{C})$ in which

$$\begin{aligned} \mathcal{U} &= (u_1, \texttt{[CLS]}, u_2, ..., \texttt{[CLS]}, u_\tau), \\ \mathcal{T} &= (t_1, \texttt{[SEP]}, t_2, ..., \texttt{[SEP]}, t_{|T|}), \\ \mathcal{C} &= (c_1, \texttt{[SEP]}, c_2, ..., \texttt{[SEP]}, c_{|C|}). \end{aligned} \tag{2}$$

All the special separator tokens and language word tokens in x are converted to the word embedding by embedding layer of the language model. Gathering the embeddings of natural language and SQL, we feed them to self-attention blocks in a language model. In the training stage, we directly take the golden SQL query of the last turn as an input SQL query. As for the inference stage, we apply the SQL query generated by HIE-SQL in the last turn. When predicting the first turn utterance, we just set $\mathcal{S}$ to empty.

3.3 SQLBERT

As mentioned above, we treat the SQL query as another modality that can provide information of the SQL query from the previous round as a reference for the model. So we need an encoder to extract the representation of the SQL query.

Model Architecture. Considering the success of multi-modal pre-trained models, such as ViLBERT (Lu et al., 2019) for language-image and CodeBERT (Feng et al., 2020) for natural language and programming language, we propose SQLBERT, a bimodal pre-trained model for natural language and SQL. We develop SQLBERT by using the same model architecture as RoBERTa (Liu et al., 2019). The total number of model parameters is 125M.

Input format. As the training method showed in Figure 3, we set the same input as CodeBERT (Feng et al., 2020) does. To alleviate the difficulty of training and resolve inconsistencies between natural language and schema, we append the question-relevant database schema to the concatenation of SQL query and question. We represent the whole input sequence into the format as $x = (\texttt{[CLS]}, s_1, s_2, ..s_n, \texttt{[SEP]}, q_1, q_2, ..q_m, \texttt{[SEP]}, t_1 : c_{11}, c_{12}, ..., \texttt{[SEP]}, t_2 : c_{21}, ..., \texttt{[SEP]}, ...)$, in which s, q, t, and c are the tokens of SQL query, question, tables, and columns respectively.

Training Objective. The main training objective of SQLBERT is the masked language modeling (MLM). It's worth noting that we only mask the tokens of SQL query because we only need SQLBERT to encode SQL query in the downstream task. Specifically, we utilize a special objective referenced span masking (Sun et al., 2019) by sampling 15% independent span in SQL clause except the reserved word (e.g., SELECT, FROM, WHERE), which aims to avoid leaking answers and help SQLBERT learn the information structure of SQL better. In the training stage, we adopt a dynamic masking strategy via randomly shuffling the order of tables and columns in the original schema. We describe the masked span prediction loss as

$$\mathcal{L}(\theta) = \sum_{k=1}^{n} -log\mathcal{P}_\theta(s_k^{mask} | s^{\backslash mask}, q, t, c), \tag{3}$$

where θ stands for the model parameters, s_k^{mask} is the masked span of SQL input, $s^{\backslash mask}$ is the unmasked part.

Training data. We train SQLBERT with the open-source Text-to-SQL datasets including Spider, SparC and CoSQL, whose data structures and annotation styles are quite similar. For each sample, we only use its question, SQL query, and the corresponding database schema. As for SparC and CoSQL, which is a context-dependent version, we simply concatenate the current utterance with the

	Current Utterance	Interaction History	SQL Query
Columns	U−C−EM (Exact Match)	H−C−EM	S−C−EC (Equal Columns)
	U−C−PM (Partial Match)	H−C−PM	S−C−UC (Unequal Columns)
	U−C−VM (Value Match)	H−C−VM	
Tables	U−T−EM	H−T−EM	S−T−ET (Equal Tables)
	U−T−PM	H−T−PM	S−T−UT (Unequal Tables)

Table 1: Edge types between current utterance U, interaction history H, SQL S, and database schema D (Columns C and Tables T). We omit the pre-existing relations in schema such as the foreign-key relation (C-C-FK) in the table. We set a default "no relation" edge type for every node pair.

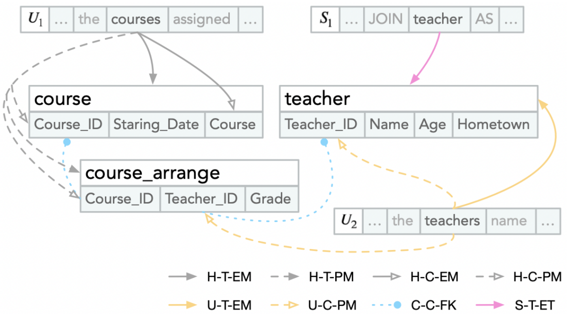

Figure 4: An example of the schema-linking graph for the prediction of S_2 in Figure 1. The graph is a sub-graph of the whole schema-linking graph. We only respectively choose one token in the history utterance (U_1), the current utterance (U_2), and the last predicted SQL query (S_1) in the example. Besides, we omit all unequal relation edges (S-C-UC and S-T-UT) and default "no relation" edges.

history utterances to build the question input. The size of the training dataset is 34,175.

3.4 HIE-Layers

Schema-Linking Graph. To explicitly encode the complex relational database schema. We convert it to a directed graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$, where $\mathcal{V} = C \cup T$ and $\mathcal{E}$ represents the set of pre-existing relations within columns and tables such as the foreign-key relation. In addition, we also consider the unseen linking to the schema in the contexts of current utterance, interaction history utterances, and the last predicted SQL query. Specifically, we define the context-dependent schema-linking graph $\mathcal{G}_c = \langle \mathcal{V}_c, \mathcal{E}_c \rangle$ where $\mathcal{V}_c = C \cup T \cup U \cup H \cup S$ and $\mathcal{E}_c = \mathcal{E} \cup \mathcal{E}_{U \leftrightarrow D} \cup \mathcal{E}_{H \leftrightarrow D} \cup \mathcal{E}_{S \leftrightarrow D}$. The additional relation edges are listed in Table 1. We set two match types between the language tokens of U, H, and D: EM for Exact Match, PM for Partial Match. When using database contents, we set VM (Value Match) for exactly matching the value of columns. As for SQL S, we simply match the words of tables and columns that appear in it to the target database schema: EC (Equal Columns) for column match, ET (Equal Tables) for table match. In Figure 4, we show an example of the proposed schema-linking graph.

Graph Encoding. The work (Wang et al., 2020) shows that Relative Self-Attention Mechanism (Shaw et al., 2018) is an efficient way to encode graphs whose nodes are at the token level. It rebuilds the calculation of the self-attention module in the transformer layers as follows:

$$
\begin{aligned}
e_{ij} &= \frac{x_i W^Q (x_j W^K + r_{ij}^K)^T}{\sqrt{d_z}}, \\
\alpha_{ij} &= \underset{j}{softmax}\{e_{ij}\}, \\
z_i &= \sum_{j=1}^{n} \alpha_{ij} (x_j W^V + r_{ij}^V).
\end{aligned} \tag{4}
$$

HIE-Layers consist of 8 transformer layers, whose self-attention modules are described above. Specifically, we initialize a learned embedding for each type of edge defined above. For every input sample, we build a relation matrix $\mathcal{R} \subseteq (L \times L)$ where L is the length of the input token. $\mathcal{R}^{(i,j)}$ represents the relation type between $i-$th and $j-$th input tokens. While computing the relative attention, we set the $r_{ij}^K = r_{ij}^V = \mathcal{R}_e^{(i,j)}$ where $\mathcal{R}_e^{(i,j)}$ is the corresponding embedding of $\mathcal{R}^{(i,j)}$.

3.5 Decoder

To build the decoder of HIE-SQL, we apply the same work (Yin and Neubig, 2017) as Wang et al. (2020) propose, which generates SQL as an abstract syntax tree in depth-first traversal order by using LSTM (Hochreiter and Schmidhuber, 1997) to output sequences of decoder actions. We recommend the reader to refer to the work (Yin and Neubig, 2017) for details.

Dataset	System Response	Interaction	Train	Dev	Test	User Questions	Vocab	Avg Turn
CoSQL	✔	3007	2164	293	551	15598	9585	5.2
SparC	✗	4298	3034	422	842	12726	3794	3.0

Table 2: Details of SparC and CoSQL datasets.

Model	SparC				CoSQL			
	Dev		Test		Dev		Test	
	QM	IM	QM	IM	QM	IM	QM	IM
EditSQL + BERT (Zhang et al., 2019)	47.2	29.5	47.9	25.3	39.9	12.3	40.8	13.7
IGSQL + BERT (Cai and Wan, 2020)	50.7	32.5	51.2	29.5	44.1	15.8	42.5	15.0
IST-SQL + BERT (Wang et al., 2021)	-	-	-	-	44.4	14.7	41.8	15.2
R^2SQL + BERT (Hui et al., 2021)	54.1	35.2	55.8	30.8	45.7	19.5	46.8	17.0
RAT-SQL† + SCoRe (Yu et al., 2021b)	62.2	42.5	62.4	38.1	52.1	22.0	51.6	21.2
T5-3B + PICARD† (Scholak et al., 2021)	-	-	-	-	**56.9**	24.2	**54.6**	23.7
HIE-SQL + GraPPa (ours)	**64.7**	**45.0**	**64.6**	**42.9**	56.4	**28.7**	53.9	**24.6**

Table 3: Performances of various models in SparC and CoSQL. QM and IM stand for question match and interaction match respectively. The models with † are proposed for the context-independent text-to-SQL task and applied to the context-dependent text-to-SQL task by just appending interaction history utterances to the input.

3.6 Regularization Strategy

We introduce R-Drop (Liang et al., 2021), a simple regularization strategy, to prevent the overfitting of the model. Concretely, we feed every input data x_i to go through our model twice and the loss function is as follows:

$$\begin{aligned}\mathcal{L}^i_{NLL} &= -\,log\mathcal{P}_1(y_i|x_i) - log\mathcal{P}_2(y_i|x_i),\\ \mathcal{L}^i_{KL} &= \frac{1}{2}(D_{KL}(\mathcal{P}_1(y_i|x_i)\|\mathcal{P}_2(y_i|x_i))\\ &\quad + D_{KL}(\mathcal{P}_2(y_i|x_i)\|\mathcal{P}_1(y_i|x_i))),\\ \mathcal{L}^i &= \mathcal{L}^i_{NLL} + \mathcal{L}^i_{KL},\end{aligned} \tag{5}$$

where -$log\mathcal{P}_1(y_i|x_i)$ and -$log\mathcal{P}_2(y_i|x_i)$ are two output distributions for input x_i at all decoder steps, $\mathcal{L}^i_{NLL}$ is the negative log-likelihood learning objective of decoder actions, and $\mathcal{L}^i_{KL}$ is the bidirectional Kullback-Leibler (KL) divergence between these two output distributions.

4 Experiment

4.1 Setup

Setting. We initialize the weights of Language Model with GraPPa (Yu et al., 2021a), an effective pre-training model for table semantic parsing that performs well on the context-independent text-to-SQL datasets (e.g. Spider). We stack 8 HIE-layers, which are introduced in section 3.4, on top of the Language Model. When training the model with R-Drop, we set the Dropout rate of 0.1 for the Language Model and HIE-Layers, 0.3 for the decoder. We use Adam optimizer to conduct the parameter learning and set the learning rate of $1e^{-5}$ for fine-tuning GraPPa and $1e^{-4}$ for HIE-Layers and Decoder. The learning rate linearly increases to the setting point at first $max\_steps/8$ steps, then decreases to 0 at max_steps, where $max\_steps = 50000$ with 24 training batch-size. As for SQL-BERT, we fine-tune CodeBERT$_{BASE}$ (Feng et al., 2020) on the dataset we described in Section 3.3. We set the learning rate as $1e^{-5}$, a batch size of 64, and train SQLBERT for 10 epochs. The shape of learned weights of the linear layer applied to the output of SQLBERT is 768×1024. While inferring, we set the beam size to 3.

Datasets. We conduct experiments on two cross-domain context-dependent text-to-SQL datasets, SparC (Yu et al., 2019b) and CoSQL (Yu et al., 2019a). Table 2 depicts the statistic information of them.

Evaluation Metrics. The main metric we used to measure model performance in SparC and CoSQL is interaction match (IM), which requires all output SQL queries in interaction to be correct. We also

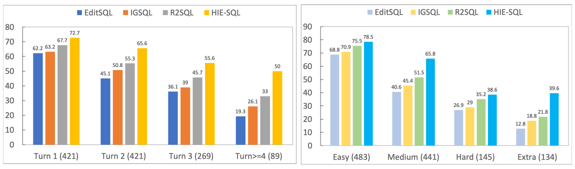

Figure 5: Performances of previous works and HIE-SQL in different turns (left) and different difficulty levels (right) on SparC.

use question match (QM) to evaluate the accuracy of every single question.

4.2 Experiment Result.

Results of our proposed HIE-SQL model are shown in Table 3. In terms of interaction match, our model achieves state-of-the-art performances on both development set and test set of SparC and CoSQL. For the test set of SparC, HIE-SQL outperforms the prior state-of-the-art (Yu et al., 2021b) by 4.8% in IM and 2.2% in QM. For CoSQL, compared with the previous state-of-the-art (Scholak et al., 2021), a rule-based auto-regressive method based on the large pre-trained model-T5-3B (Raffel et al., 2020) which is optimized for a GPU with 40GB of memory, HIE-SQL improves IM of development set by 4.5% and IM of the test set by 0.9%. Besides, HIE-SQL surpasses RAT-SQL + SCoRe in all metrics of SparC and CoSQL. This demonstrates that properly integrating interaction utterances and predicted SQL queries is an effective way to enhance the model's ability for Context-Dependent Text-to-SQL Semantic Parsing.

To further explore the advantages of HIE-SQL, we test the performance on different turns and at different difficulty levels of utterances. As shown in Figure 5, with the increase of turns, the lead of our model gets greater and greater. When the indexes of turns are greater than or equal to 4, the accuracy of HIE-SQL is 17% higher than that of R^2SQL. It demonstrates that the main contribution of introducing SQL query is to improve the robustness of the model to long interaction. HIE-SQL is also robust to the varying difficulty levels of utterances. Our model performs equally in hard and extra hard levels, and achieves 39.6% accuracy on the extra hard level, which is 17.8% higher than that of R^2SQL.

4.3 Ablation Study

We provide ablation studies to examine the contribution of each component of HIE-SQL. We want

	SparC		CoSQL	
Model	**QM**	**IM**	**QM**	**IM**
HIE-SQL	64.7	**45.0**	56.4	**28.7**
w/o SQL query	**65.8**	44.3	**56.5**	23.9
w/o SQLBERT	63.9	44.7	54.8	26.3
w/o $\mathcal{E}_{H\leftrightarrow D}$	64.0	44.3	56.0	26.3

Table 4: Ablation study of HIE-SQL in development sets of SparC and CoSQL. As for ablation on SQL query, we drop the SQL query and only feed utterances and database schema to the model. As for ablation on SQLBERT, we directly concatenate the tokens of SQL query and other context tokens for the input of the language model. And w/o $\mathcal{E}_{H\leftrightarrow D}$ means we treat historical utterances like the current utterance in our schema-linking.

Dataset	Model	T-F	F-T	T-T
SparC	HIE-SQL	125	88	**383**
	w/o SQL query	132	104	379
CoSQL	HIE-SQL	140	106	**278**
	w/o SQL query	161	128	254

Table 5: The counts of different switches in the pairs of adjacent predicted SQL queries. T-F stands for the match of the former predicted query and unmatch of the later predicted query with golden queries. F-T stands for the reverse case. T-T is the case of both matching.

to identify whether introducing the last SQL query has a significant impact on performance. Also, we would like to investigate whether the pre-trained SQL encoder, SQLBERT, can improve the model's ability to understand SQL queries. What's more, we conduct another ablation study regarding additional graph edges between historical utterances and database schema $\mathcal{E}_{H\leftrightarrow D}$ to check the necessity of the join of historical utterance information in schema-linking.

As shown in Table 4, Our full model achieves about 5 points and 1 point improvement of IM in CoSQL and SparC respectively compared with the model without the last SQL query input. The pre-encoding SQL query by SQLBERT can further improve the performance. It confirms SQLBERT's ability to efficiently represent SQL features. In addition, $\mathcal{E}_{H\leftrightarrow D}$ also plays a positive role.

Table 5 shows the continuity of performance of our model compared with that of the model without the last SQL query input. Our model has a higher rate of continuous match, but a lower rate

U_1	Which cartoon aired first?
HIE-SQL	SELECT title FROM cartoon ORDER BY original_air_date asc LIMIT 1
RAT-SQL	SELECT title FROM cartoon ORDER BY original_air_date asc LIMIT 1
U_2	What was the last cartoon to air?
HIE-SQL	SELECT title FROM cartoon ORDER BY original_air_date desc LIMIT 1
RAT-SQL	SELECT title FROM cartoon ORDER BY original_air_date desc LIMIT 1
U_3	What channel was it on?
HIE-SQL	SELECT channel FROM cartoon ORDER BY original_air_date desc LIMIT 1
RAT-SQL	SELECT channel FROM cartoon ORDER BY original_air_date desc LIMIT 1
U_4	What is the production code?
HIE-SQL	SELECT production_code FROM cartoon ORDER BY original_air_date desc LIMIT 1
RAT-SQL	SELECT production_code FROM cartoon ORDER BY original_air_date asc LIMIT 1

Table 6: An example in CoSQL. U_i is the input utterance of turn i with corresponding predictions of HIE-SQL and RAT-SQL following. All predictions of HIE-SQL are the ground truth queries in the case.

of switching from mismatch to match. It illustrates that our model does use the SQL information and is sensitive to the accuracy of the last predicted SQL query which explains the higher question match without SQL query input.

We regard R-Drop as a simple means of data augmentation which can improve the generalization of the model. As shown in Figure 6, the model with R-drop outperforms the model without R-Drop in both QM and IM. Additionally, the standard deviations of the IM in the last 20k steps are 0.014 and 0.015 of HIE-SQL and the one without R-Drop respectively even the curve of HIE-SQL has a more obvious upward trend. It shows that R-Drop improves the robustness of our model and stabilizes its performance in IM.

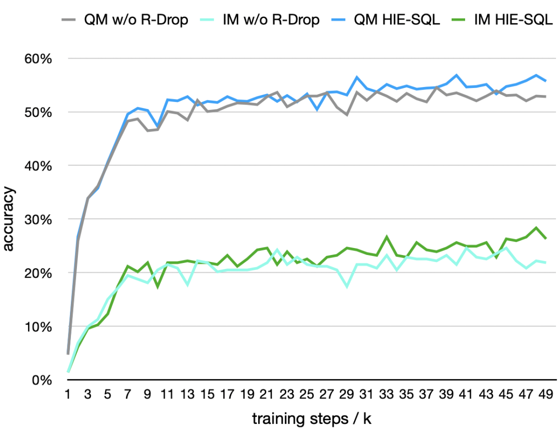

Figure 6: Ablation study result of regarding R-Drop in development set of CoSQL. We show the performances in QM and IM of two models at different training steps. We set the beam size = 1 in the inference stage.

4.4 Case Study

In Table 6, we show the predictions of HIE-SQL and RAT-SQL in an example of CoSQL. Here, HIE-SQL and RAT-SQL both fine-tune GraPPa on CoSQL. As the example shows, RAT-SQL fails to distinguish the right one from two long-range dependences in U_1 and U_2 in Table 6. By contrast, HIE-SQL inherits the right context-dependence from the last predicted query to avoid confusion between U_1 and U_2.

5 Conclusion

We present HIE-SQL, a history information enhanced context-dependent text-to-SQL model, which targets at explicitly capturing the context-dependence from both interaction history utterances and the last predicted SQL query. With the help of the proposed bimodal pre-trained model, SQLBERT, HIE-SQL bridge the gap between the utterances and predicted SQL despite the mismatch of natural language and logic-form SQL. Moreover, we also introduce a method of schema-linking to enhance the connections among utterances, SQL query, and database schema.

Taken together, HIE-SQL achieves consistent improvements on the context-dependent text-to-SQL task, especially in the interaction match metric. HIE-SQL achieves new state-of-the-art results on two famous context-dependent text-to-SQL datasets, SparC and CoSQL.

References

Ben Bogin, Jonathan Berant, and Matt Gardner. 2019a. Representing schema structure with graph neural networks for text-to-SQL parsing. In *Proceedings of the 57th Conference of the Association for Computational Linguistics (ACL)*, volume 1, pages 4560–4565.

Ben Bogin, Matt Gardner, and Jonathan Berant. 2019b. Global reasoning over database structures for text-to-SQL parsing. 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 3657–3662.

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.

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 (ACL/IJCNLP)*, volume 1, pages 2541–2555.

Zhi Chen, Lu Chen, Yanbin Zhao, Ruisheng Cao, Zihan Xu, Su Zhu, and Kai Yu. 2021. ShadowGNN: Graph projection neural network for text-to-SQL parser. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, volume 1, pages 5567–5577.

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 (NAACL-HLT)*, volume 1, pages 1337–1350.

Kedar Dhamdhere, Kevin S McCurley, Ralfi Nahmias, Mukund Sundararajan, and Qiqi Yan. 2017. Analyza: Exploring data with conversation. In *Proceedings of the 22nd International Conference on Intelligent User Interfaces (ACM IUI)*, pages 493–504.

Zhangyin Feng, Daya Guo, Duyu Tang, Nan Duan, Xiaocheng Feng, Ming Gong, Linjun Shou, Bing Qin, Ting Liu, Daxin Jiang, and Ming Zhou. 2020. CodeBERT: A pre-trained model for programming and natural languages. In *Findings of the Association for Computational Linguistics: EMNLP 2020 (EMNLP-Findings)*, pages 1536–1547.

Yujian Gan, Xinyun Chen, Jinxia Xie, Matthew Purver, John R Woodward, John Drake, and Qiaofu Zhang. 2021. Natural SQL: Making SQL easier to infer from natural language specifications. In *Findings of the Association for Computational Linguistics: EMNLP 2021 (EMNLP-Findings)*.

Jiaqi Guo, Zecheng Zhan, Yan Gao, Yan Xiao, Jian-Guang Lou, Ting Liu, and Dongmei Zhang. 2019. Towards complex text-to-SQL in cross-domain database with intermediate representation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics (ACL)*, volume 1, pages 4524–4535.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

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 Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI)*, volume 35, pages 13116–13124.

Douwe Kiela, Suvrat Bhooshan, Hamed Firooz, and Davide Testuggine. 2019. Supervised multimodal bitransformers for classifying images and text. In *Visually Grounded Interaction and Language (ViGIL), NeurIPS 2019 Workshop*.

Wenqiang Lei, Weixin Wang, Zhixin Ma, Tian Gan, Wei Lu, Min-Yen Kan, and Tat-Seng Chua. 2020. Re-examining the role of schema linking in text-to-sql. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6943–6954.

Xiaobo Liang, Lijun Wu, Juntao Li, Yue Wang, Qi Meng, Tao Qin, Wei Chen, Min Zhang, and Tie-Yan Liu. 2021. R-Drop: Regularized dropout for neural networks. *arXiv preprint arXiv:2106.14448*.

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

Qian Liu, Bei Chen, Jiaqi Guo, Jian-Guang Lou, Bin Zhou, and Dongmei Zhang. 2020. How far are we from effective context modeling? an exploratory study on semantic parsing in context. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3580–3586.

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*.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. ViLBERT: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems (NeurIPS)*, pages 13–23.

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. *J. Mach. Learn. Res.*, 21:140:1–140:67.

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 (NAACL-HLT)*, volume 1, pages 311–324.

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 (EMNLP)*.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, volume 2, pages 464–468.

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

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*.

Maria Tsimpoukelli, Jacob Menick, Serkan Cabi, SM Eslami, Oriol Vinyals, and Felix Hill. 2021. Multimodal few-shot learning with frozen language models. *arXiv preprint arXiv:2106.13884*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems (NeurIPS)*, pages 5998–6008.

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 (ACL)*, pages 7567–7578.

Runze Wang, Zhen-Hua Ling, Jingbo Zhou, and Yu Hu. 2021. Tracking interaction states for multi-turn text-to-SQL semantic parsing. In *Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI)*, pages 13979–13987.

Nathaniel Weir, Prasetya Utama, Alex Galakatos, Andrew Crotty, Amir Ilkhechi, Shekar Ramaswamy, Rohin Bhushan, Nadja Geisler, Benjamin Hättasch, Steffen Eger, et al. 2020. DBPal: A fully pluggable nl2sql training pipeline. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, pages 2347–2361.

Yuk Wah Wong and Raymond J. Mooney. 2007. Learning synchronous grammars for semantic parsing with lambda calculus. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 960–967.

Peng Xu, Dhruv Kumar, Wei Yang, Wenjie Zi, Keyi Tang, Chenyang Huang, Jackie Chi Kit Cheung, Simon J. D. Prince, and Yanshuai Cao. 2021. Optimizing deeper transformers on small datasets. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL/IJCNLP)*, volume 1, pages 2089–2102.

Pengcheng Yin and Graham Neubig. 2017. A syntactic neural model for general-purpose code generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 440–450.

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)*.

Tao Yu, Michihiro Yasunaga, Kai Yang, Rui Zhang, Dongxu Wang, Zifan Li, and Dragomir R. Radev. 2018a. SyntaxSQLNet: Syntax tree networks for complex and cross-domain text-to-SQL task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1653–1663.

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 R. Fabbri, Zifan Li, Luyao Chen, Yuwen Zhang, Shreya Dixit, Vincent Zhang, Caiming Xiong, Richard Socher, Walter S. Lasecki, and Dragomir R. 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.

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)*.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir R. Radev. 2018b. 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 (EMNLP)*, 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, Emily Ji, Shreya Dixit, David Proctor, Sungrok Shim, Jonathan Kraft, Vincent Zhang, Caiming Xiong, Richard Socher, and Dragomir R. Radev. 2019b. SParC: Cross-domain semantic parsing in context. In *Proceedings of the 57th Conference of the Association for Computational Linguistics (ACL)*, volume 1, pages 4511–4523.

John M. Zelle and Raymond J. Mooney. 1996. Learning to parse database queries using inductive logic programming. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence and Eighth Innovative Applications of Artificial Intelligence Conference (AAAI/IAAI)*, pages 1050–1055.

Luke S. Zettlemoyer and Michael Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *Proceedings of the 21st Conference in Uncertainty in Artificial Intelligence (UAI)*, pages 658–666.

Rui Zhang, Tao Yu, Heyang Er, Sungrok Shim, Eric Xue, Xi Victoria Lin, Tianze Shi, Caiming Xiong, Richard Socher, and Dragomir R. 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 5337–5348.