Dynamic Hybrid Relation Network for Cross-Domain Context-Dependent Semantic Parsing

Binyuan Hui,[1,2] Ruiying Geng,[2] Qiyu Ren,[3] Binhua Li,[2] Yongbin Li,[2*]
Jian Sun,[2] Fei Huang,[2] Luo Si,[2] Pengfei Zhu,[1*] Xiaodan Zhu[4]
[1] College of Intelligence and Computing, Tianjin University, Tianjin, China
[2] Alibaba Group, Beijing, China
[3] Beijing University of Posts and Telecommunications, Beijing, China
[4] Ingenuity Labs Research Institute & ECE, Queen's University
{huibinyuan,zhupengfei}@tju.edu.cn, zhu2048@gmail.com
{ruiying.gry,binhua.lbh,shuide.lyb,jian.sun,f.huang,luo.si}@alibaba-inc.com

Abstract

Semantic parsing has long been a fundamental problem in natural language processing. Recently, cross-domain context-dependent semantic parsing has become a new focus of research. Central to the problem is the challenge of leveraging contextual information of both natural language utterance and database schemas in the interaction history. In this paper, we present a dynamic graph framework that is capable of effectively modelling contextual utterances, tokens, database schemas, and their complicated interaction as the conversation proceeds. The framework employs a dynamic memory decay mechanism that incorporates inductive bias to integrate enriched contextual relation representation, which is further enhanced with a powerful reranking model. At the time of writing, we demonstrate that the proposed framework outperforms all existing models by large margins, achieving new state-of-the-art performance on two large-scale benchmarks, the SParC and CoSQL datasets. Specifically, the model attains a 55.8% question-match and 30.8% interaction-match accuracy on SParC, and a 46.8% question-match and 17.0% interaction-match accuracy on CoSQL.

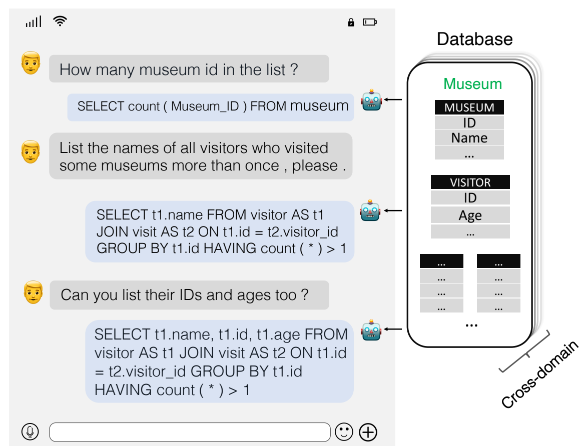

Figure 1: An example of cross-domain context-dependent Text-to-SQL interaction in the CoSQL dataset (Yu et al. 2019a).

Introduction

Mapping a nature language sentence into a logical form, known as semantic parsing, is a fundamental problem in natural language processing (Zelle and Mooney 1996; Zettlemoyer and Collins 2005; Wong and Mooney 2007; Zettlemoyer and Collins 2007; Li and Jagadish 2014; Yaghmazadeh et al. 2017; Iyer et al. 2017). Notably, the recent Text-to-SQL tasks have attracted considerable attention, which aims to convert nature language sentences into SQL queries. Large-scale datasets such as SParC (Yu et al. 2019b) and CoSQL (Yu et al. 2019a) have been made available to train more powerful models. Since relational databases store a great amount of structured data, improving the performance of Text-to-SQL conversion is important for many real-life applications.

Most existing work has focused on precisely converting individual utterances to SQL queries, with a strong assumption on context independence among queries. However, in real applications as shown in Figure 1, users interact with cross-domain databases through consecutively communication to exchange information with the databases. Unfortunately, the state-of-the-art approaches do not perform well on the newly released, cross-domain context-dependent Text-to-SQL benchmarks, SparC and CoSQL.

Central to the problem of context-dependent semantic parsing is leveraging interactive contextual information of both natural language utterance and database schemas available in the interaction history. Existing work (Suhr, Iyer, and Artzi 2018; Zhang et al. 2019; Liu et al. 2020) has mainly focused on integrating context information into the utterance encoding phase.

In this paper, we present a dynamic graph framework that is capable of effectively modelling contextual utterances,

*Corresponding authors: Yongbin Li and Pengfei Zhu

tokens, database schemas, and their complicated interaction when a conversation proceeds. In our evolving dynamic schema-linking graph network, the representation for nodes, edges, and relation weights adapt in the interactive context. The graph is built at both the utterance and word token level, rendering a flexible framework to model different levels of context in the multi-turn scenario. We propose different dynamic memory decay mechanisms to incorporate inductive bias that encourages forgetting the part of dynamic graphs of a longer history, with which dynamic context representation is constructed to leverage both implicit and explicit relations. The framework can effectively leverage features that are powerful in context-independent parsing such as explicit relations (Wang et al. 2020). We show that such information is also effective for the context-dependent parsing.

We further design a feature enhanced reranker that integrates external knowledge and task-related representation. The model can identify the correct queries by filtering out those that do not conform to the grammar of SQL and further improve the performance of Text-to-SQL models.

We evaluate our proposed model on two large-scale cross-domain context-dependent benchmarks, SParC (Yu et al. 2019b) and CoSQL (Yu et al. 2019a). At the time of writing this paper, the proposed model achieve new state-of-the-art performance on both datasets, substantially outperforming all existing models by large margins. Specifically, our model attains a 55.8% question-match and 30.8% interaction-match accuracy on SParC, and a 46.8% question-match and 17.0% interaction-match accuracy on CoSQL. We provide detailed analysis and visualization to further investigate how each component contributes to the entire framework.

Related Work

Context-independent Semantic Parsing. Semantic parsing (Zelle and Mooney 1996; Zettlemoyer and Collins 2005; Wong and Mooney 2007; Zettlemoyer and Collins 2007) maps natural language utterances into logical forms. Recently, Text-to-SQL is a major focus of semantic parsing in which natural language sentence questioning tables are parsed into SQL queries. Deep learning has shown to achieve impressive results on context-independent Text-to-SQL datasets such as WikiSQL (Zhong, Xiong, and Socher 2017) and Spider (Yu et al. 2018b). For WikiSQL, Dong and Lapata (2018) propose the Coarse2Fine model which generates meaning sketches abstracted away from low-level information such as arguments and variable names and predicts missing details in order to obtain full meaning representations. McCann et al. (2018) propose MQAN, a model for general question answering that uses a multi-pointer-generator decoder to capitalize on questions as natural language descriptions of tasks. Furthermore, Hwang et al. (2019) introduce the large pretrained language model and demonstrate the effectiveness of a carefully designed architecture that combines previous approaches. Compared with Spider, WikiSQL does not involve the complexity of multiple tables. On Spider, most of work focuses on establishing schema linking, which dynamically obtains the relationships between natural language sentences and database schemas through attention mechanism. Guo et al. (2019) perform schema linking over a question and database schema using customized type vectors for alignment and adopts a grammar-based model (Yin and Neubig 2017) to synthesize an intermediate representation. Bogin, Berant, and Gardner (2019) provide a new perspective on schema linking, which converts the schema to a graph. Wang et al. (2020) propose the RAT-SQL framework, providing a unified way to encode arbitrary relational information among inputs. Unlike this work, we study context-dependent parsing.

Context-dependent Semantic Parsing. Miller et al. (1996) maps utterances to semantic frames, which are then mapped to SQL queries on the ATIS dataset (Hemphill, Godfrey, and Doddington 1990) that has only one database. Similar to ATIS, SCONE (Long, Pasupat, and Liang 2016; Guu et al. 2017; Fried, Andreas, and Klein 2018; Suhr and Artzi 2018; Huang, Choi, and Yih 2019) and SequentialQA (Iyyer, Yih, and Chang 2017) contain no logical form annotations and only denotation (Berant and Liang 2014) instead. Zettlemoyer and Collins (2009) propose a context-independent CCG parser and then applied it to do context-dependent substitution. Furthermore, Suhr, Iyer, and Artzi (2018) generate ATIS SQL queries from interactions by incorporating history with an interaction-level encoder and copy segments of previously generated queries. More recently, Yu et al. (2019a,b) construct two large-scale cross-domain context-dependent benchmarks for semantic parsing. Beyond that, Zhang et al. (2019) present an edit-based method that reuses the SQL query generated in the previous time step through editing, which achieves promising results. Liu et al. (2020) conduct a further exploratory study on semantic parsing in context and perform a fine-grained analysis to explore the sensitivity of input utterance and decoding to context. As discussed above, we present a dynamic graph framework that is capable of more effectively modelling contextual utterances, tokens, database schemas, and their complicated interaction.

Task Formulation and Notations

The context-dependent semantic parsing task consists of interactive dialogues in different domains, and its goal is to map nature language utterance in the interaction to the corresponding SQL queries. Let $\mathcal{I}$ be the set of all interactions, an interaction $I \in \mathcal{I}$ is a series of utterances $\langle x_1, \ldots, x_n \rangle$, and their corresponding SQL queries $\langle y_1, \ldots, y_n \rangle$, where n is the length of the interaction. In the cross-domain setting, each SQL query is grounded to a multi-table schema and each interaction uses different datasets. The schema involved in each interaction can be expressed as $S = \langle s_1, \ldots, s_m \rangle$, where m is the number of column headers. In order to describe the relationships between columns and tables more effectively, for each schema header s, the column and table name are formatted as [Table.Column] and [Table.*], respectively. Given the current utterance x_i, the involved schema S, and the interaction history of length $i-1$, formatted as $I[: i-1] = \langle (x_1, y_1), \ldots, (x_{i-1}, y_{i-1}) \rangle$, the goal is to generate SQL query y_i.

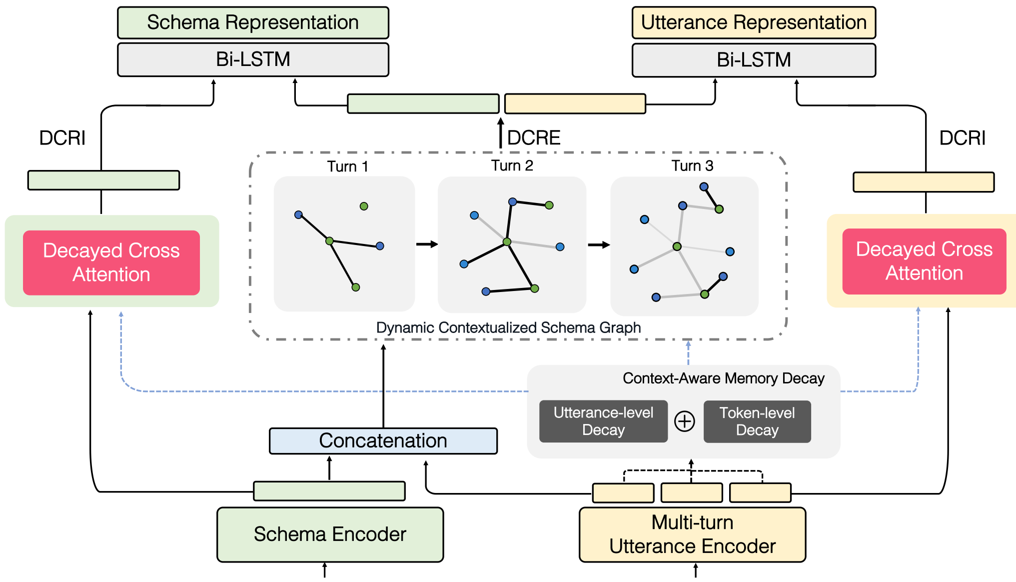

Figure 2: Illustration of the proposed model architecture.

Model

The overall architecture of our proposed model is depicted in Figure 2. In the following sections, we will discuss the components in detail.

Encoder

BERT Embedding Input The pretrained language models have shown superior performance in many tasks. We utilize BERT (Devlin et al. 2019) to encode both utterances and schema-related input simultaneously. Same as in (Hwang et al. 2019), we concatenate all utterances in one conversation and all the schema-related input using `[SEP]` as the delimiter:

$$[\texttt{CLS}], [x_1, \ldots, x_i], [\texttt{SEP}], s_1, [\texttt{SEP}], \ldots, s_m, [\texttt{SEP}]. \quad (1)$$

As such, we obtain BERT's utterance and schema representation by feeding the sequence to a pretrained BERT.

Multi-turn Utterance Encoder To encode the current utterance and effectively integrate information from conversation history, at each turn i, we employ an utterance-level Bi-LSTM (Hochreiter and Schmidhuber 1997) to produce embedding from a contextual hidden state:

$$\mathbf{h}^U_{i,k} = \textbf{Bi-LSTM}^U \left(x_{i,k}, \mathbf{h}^U_{i,k-1}\right), \quad (2)$$

We use the concatenation of the first and last hidden vector as the utterance encoding. To take advantage of the utterances in the history, we employ the popular interaction-level encoder (Suhr, Iyer, and Artzi 2018). For i-th utterance, the interaction-level encoder merges the current utterance embedding $\mathbf{h}^U_i$ with the preceding interaction-level encoding $\mathbf{h}^I_{i-1}$:

$$\mathbf{h}^I_i = \textbf{LSTM}^I \left(\mathbf{h}^U_i, \mathbf{h}^I_{i-1}\right). \quad (3)$$

This state is maintained and updated over the entire interaction. Moreover, we use interaction-level embedding $\mathbf{h}^I_i$ to further enrich utterance encoding:

$$\mathbf{h}^U_{i,k} = \textbf{LSTM}^U \left([x_{i,k}, \mathbf{h}^I_i], \mathbf{h}^U_{i,k-1}\right). \quad (4)$$

Schema Encoder To make the model capable of modelling cross-domain information, in addition to utterance encoding, we encode the schema involved in the current interaction. For each schema input s, which consists of table and column names, the schema embedding $\mathbf{h}^S_i$ are processed by a Bi-LSTM layer:

$$\mathbf{h}^S_{i,k} = \textbf{Bi-LSTM}^S \left(s_{i,k}, \mathbf{h}^S_{i,k-1}\right). \quad (5)$$

Dynamic Contextualized Schema Graph

For each conversation that contains multiple turns, we design dynamic contextualized schema graphs, inspired by the recent studies on dynamically evolving structures (Pareja et al. 2020). Unlike the original model that was not used for natural language, we construct our dynamic contextualized schema linking graphs, which will change as the conversation proceeds. This graph will be used with dynamic weight decay modules we discuss below to learn enriched contextual representation.

We would like to jointly learn our representation for utterance $\mathcal{X}$ and schema $\mathcal{S}$ in context $\mathcal{C}$, in particular considering modeling the alignment between them. At the i-th turn, given the interaction $X = \{x_1^{0\ldots k_1}, x_2^{0\ldots k_2}, \ldots, x_i^{0\ldots k_i}\}$ and the related set of schemas $S = \{s_1, \ldots, s_m\}$, we define the *dynamic contextualized schema graph* to be $\mathcal{G}_\mathcal{C} = \langle \mathcal{V}_\mathcal{C}, \mathcal{E}_\mathcal{C} \rangle$, where $\mathcal{V}_\mathcal{C} = X \cup S$, and $\mathcal{E}_\mathcal{C}$ are schema linking edges among the context words and schema members such as table and column names. Especially, the relationships can be divided into two categories:

- Internal relations: relations within a database schema, such as a *foreign key*.
- Interactive relations: relations that align entity references in utterances to the schema columns or tables.

Context-Aware Memory Decay

As the conversation proceeds and more queries are asked by users, the contextualized schema graph grows. Note that users' concerns and intention may change frequently. We would like the model to forget unrelated turns that are far away. We propose to integrate memory decay mechanism to introduce the desired inductive bias. Specifically, we construct both token-level and utterance-level weight decay framework to model the influence of context at different granularities. For each of the granularities, we provide two approaches: gate-based and schedule-based decay.

Token-level Decay We first propose a gate-based decay mechanism to automatically compute the importance of each word token. The decay weight is computed by:

$$m^T = \mathbf{Sigmoid}(V_{gate}\mathbf{ReLU}(U_{gate}\mathbf{h}^U_{i,k})), \quad (6)$$

where V_{gate} and U_{gate} are learned parameters and $\mathbf{h}^U_{i,k}$ are k-th token embeddings of the utterance of the i-th turn. In addition, for schedule-based decay, we investigate three explicit scheduling functions to compute decay weights, borrowed from schedule sampling (Bengio et al. 2015):

- Linear decay: $m^T_t = k - c * t$, where k is the base decay weight, t is the position where the decay happens, and c is a constant controlling the slope of decay;
- Exponential decay: $m^T_t = k^t$, where $k < 1$;
- Inverse sigmoid decay: $m^T_t = k/(k + \exp^{t/k})$, where $k \geq 1$;

We use word tokens' positions in the concatenated sequences (Eq. 1) to compute their distances t.

Utterance-level Decay We further propose the utterance-level decay to model the influence of history at the utterance level. Specifically, the gate-decay weight is computed by:

$$m^U = \mathbf{Sigmoid}(V^U_{gate}\mathbf{ReLU}(U^U_{gate}\mathbf{h}^I_i)), \quad (7)$$

where V^U_{gate} and U^U_{gate} are learned parameters and h^I_i is the utterance embedding of the i-th utterance. Similarly, the scheduling function could be used at the utterance-level decay. The final memory decay can be empirically selected or by a hyper-parametric weighted combination.

Dynamic Context Representation

We represent the interaction context based on the dynamic contextualized schema graph $\mathcal{G}_\mathcal{C}$ and memory decay m.

Dynamic Context Representation over Implicit Relations (DCRI) We propose a decayed attention mechanism to model implicit relations in the schema graph, which is formulated as $\mathbf{DAttn}(\mathbf{h}^Q, \mathbf{h}^K, m^a)$:

$$\begin{aligned} \alpha &= \mathbf{softmax}(\mathbf{h}^Q W_{\text{att}}\, \mathbf{h}^K \odot m^a) \\ \mathbf{DAttn}(\mathbf{h}^Q, \mathbf{h}^K, m^a) &= \sum_l \alpha_l \times \mathbf{h}^K_l, \end{aligned} \quad (8)$$

where $\mathbf{h}^Q$ and $\mathbf{h}^K$ are query and key embeddings, respectively; W_{att} are learnable parameters and l is the index of key; m^a is the memory decay values used for the current procedure. We split m into $[m^{IU}, m^S]$ to represent the utterance and schema decay values. First, we utilize the attention mechanism among all headers to explore internal relations in schema, called Schema-Inner Attention module, to update the schema $\mathbf{h}^S_i$:

$$\mathbf{h}^S = \mathbf{DAttn}(\mathbf{h}^S, \mathbf{h}^S, m^S). \quad (9)$$

It worth noting that the decay value in m^S positions are all set to 1, since there is no series definition in schema parts.

We then build a decayed attention structure to model the implicit interaction relationships between context utterances and table schemas. We use $\mathbf{h}^S$ to obtain the most relevant columns or tables:

$$\begin{aligned} \mathbf{r}^{IU} &= \mathbf{DAttn}(\mathbf{h}^U, \mathbf{h}^S, m^{IU}) \\ \mathbf{r}^{IS} &= \mathbf{DAttn}(\mathbf{h}^S, \mathbf{h}^U, m^S) \end{aligned} \quad (10)$$

where $\mathbf{r}^{IU}$ and $\mathbf{r}^{IS}$ are implicit exploration representation for utterances and schemas, respectively.

Dynamic Context Representation over Explicit Relations (DCRE) Here we further introduce dynamic context representation over the schema graphs with explicit relations, where the relation weights are influenced by the memory decay mechanism discussed above. First, we concatenate the node embedding of context with that of schema: $\mathbf{h}^R = \mathbf{Concat}\left(\mathbf{h}^U_1; ...; \mathbf{h}^U_t; \mathbf{h}^S\right)$ as the input. Then we perform a relation decayed graph transformer to obtain the structured representations:

$$e^{(h)}_{ij} = \frac{\mathbf{h}^R_i W^{(h)}_Q (\mathbf{h}^R_j W^{(h)}_K + g_{ij} \odot m_i)^\top}{\sqrt{d_z/H}}. \quad (11)$$

where $W^h_Q, W^h_K \in \mathbb{R}^{d^R_h \times (d^R_h/H)}$ are learnable parameters, the H is the number of head, g_{ij} is the explicit relationship embedding between the two element $\mathbf{h}^R_i$ and $\mathbf{h}^R_j$ from $\mathcal{E}_\mathcal{C}$ in $\mathcal{G}_\mathcal{C}$, and m_i is the memory decay value in the i-th position.

Inspired by Wang et al. (2020), we use the following internal relations: (1) *whether columns in the database belong to the same table*; (2) *whether they are foreign keys*. For the interactive relations, we determine (1) *whether utterance exactly matches the name of column/table*; (2) *whether the n-gram is subsequence of the name of a column/table*. We will show that our model can effectively incorporate these features that have been shown to be useful in static graphs in context-independent parsing, and demonstrate they can further improve the performance of our context-independent parsing.

In addition, the attention aggregation operation also needs the decay weights:

$$\begin{aligned} \alpha_{ij}^{(h)} &= \mathbf{softmax}\left(e_{ij}^{(h)}\right) \\ \mathbf{z}_i^{(h)} &= \sum_{j=1}^{n} \alpha_{ij}^{(h)} \left(\mathbf{h}_j^R W_V^{(h)} + g_{ij} \odot m_i\right). \end{aligned} \tag{12}$$

Then we can accumulate the final explicit exploration representation followed by an $\mathbf{FFN}$ operation (Vaswani et al. 2017):

$$\begin{aligned} \mathbf{z}_i &= \mathbf{Concat}\left(\mathbf{z}_i^{(1)}, \cdots, \mathbf{z}_i^{(H)}\right) \\ \mathbf{r}_i^E &= \mathbf{FFN}(\mathbf{LayerNorm}\left(\boldsymbol{x}_i + \boldsymbol{z}_i\right)). \end{aligned} \tag{13}$$

Finally, we aggregate the embedding from encoder as well as the DCRI and DCRE into new representation for utterance and schema:

$$\begin{aligned} (\mathbf{r}^{EU}, \mathbf{r}^{ES}) &= \mathbf{Split}(\mathbf{r}^E) \\ \mathbf{h}^U &= \textbf{Bi-LSTM}([\mathbf{h}^U, \mathbf{r}^{IU}, \mathbf{r}^{EU}]) \\ \mathbf{h}^S &= \textbf{Bi-LSTM}([\mathbf{h}^S, \mathbf{r}^{IS}, \mathbf{r}^{ES}]). \end{aligned} \tag{14}$$

Here, we consider that DCRE and DCRI establish the schema linking from different perspectives: DCRE pays more attention to the provided prior relationships via exact or n-gram matching. Compared to that, DCRI focuses more on semantic relations between utterance and schemas that are not directly captured in surface-form matching.

Decoder

We use an LSTM decoder with attention to generate SQL queries at time step k:

$$\mathbf{h}_k^D = \mathbf{LSTM}^D\left(\left[\mathbf{q}_k; \mathbf{c}_k\right], \mathbf{h}_{k-1}^D\right), \tag{15}$$

where h^D is the hidden state of the decoder and c_k is the context vector with the utterance and schema attention:

$$\begin{aligned} \mathbf{c}^U &= \mathbf{Attn}(\mathbf{h}^D, \mathbf{h}^U) \\ \mathbf{c}^S &= \mathbf{Attn}(\mathbf{h}^D, \mathbf{h}^S) \\ \mathbf{c} &= \mathbf{Concat}(\mathbf{c}^U, \mathbf{c}^S) \end{aligned} \tag{16}$$

We apply separate layers to score SQL keywords and column headers and finally use $\mathbf{softmax}$ to generate the output probability distribution:

$$\begin{aligned} \mathbf{o}_k &= \tanh\left(\left[\mathbf{h}_k^D; \mathbf{c}_k\right] \mathbf{W}_o\right) \\ \mathbf{m}^{\mathrm{SQL}} &= \mathbf{o}_k \mathbf{W}_{\mathrm{SQL}} + \mathbf{b}_{\mathrm{SQL}} \\ \mathbf{m}^{\text{column}} &= \mathbf{o}_k \mathbf{W}_{\text{column}} \mathbf{h}^S \\ P\left(y_k\right) &= \mathbf{softmax}\left(\left[\mathbf{m}^{\mathrm{SQL}}; \mathbf{m}^{\text{column}}\right]\right) \end{aligned} \tag{17}$$

In addition, we use query editing mechanism (Zhang et al. 2019) in the decoder progress to edit the previously generated query while incorporating the context of user utterances and schemas.

Feature Enhanced Reranker During decoding, we use beam search to generate the N-best list of SQL candidates. The generated candidate set often contain the correct SQL, but it is not the one with the highest probability. Specifically we generate an N-best list using vanilla beam search and rerank the generated responses, which has been validated in other semantic parsing tasks (Yin and Neubig 2019). Furthermore, we design a novel neural reranker module, which integrates the knowledge of the external pretraining models and the task related hidden representation from Eq 14. Suppose that the current expectation is x, the corresponding prediction SQL of the model output is y', and the current task hidden vector is represented as

$$\mathbf{h}' = \mathbf{MaxPooling}([\mathbf{h}^U, \mathbf{h}^S]). \tag{18}$$

We take the last layer's hidden state for the first token [CLS] as the deep knowledge $\mathbf{k}$ by feeding the current utterance x_i together with history and prediction y' to the BERT module:

$$[\texttt{CLS}], x_1, [\texttt{SEP}], ..., [\texttt{SEP}], x_i, [\texttt{SEP}], y'. \tag{19}$$

Based on that, we combine the task related representation and deep knowledge for joint training:

$$P(B = 1 \mid x_{1,\ldots,i}, y') = \sigma\left(\mathbf{W}_{task}\mathbf{h}' + \mathbf{W}_{knowledge}\mathbf{k}\right) \tag{20}$$

where B is the binary class label; $\sigma(\cdot)$ is the sigmoid function; $\mathbf{W}_{task}$ and $\mathbf{W}_{knowledge}$ are learnable weights. In this way, we can use the reranker module to reorder the generated candidate sets to improve the probability that the prediction is ground truth but not at the top.

Experiment

Setup

Dataset. We evaluate the performance of the proposed model on two large-scale benchmark datasets, *i.e.* , SParC (Yu et al. 2019b) and CoSQL (Yu et al. 2019a). Table. 1 summarizes the statistics of SParC and CoSQL. Both contain 200 complex databases in 138 different domains. Compared with SParC, CoSQL has a larger vocabulary and significantly more turns with frequently semantic changes, making it a more challenging dataset.

Evaluation Metrics. Following (Yu et al. 2019b), we decompose the predicted SQL into clauses such SELECT, WHERE, GROUP BY, and ORDER BY and compute scores for each clause using set matching separately to avoid ordering issues. On both SParC and CoSQL, we use the two metrics for evaluation: *Question match* and *Interaction match*. When all predicted SQL clauses are correct, the exact set matching score is one for question match, and interaction match requires that each predicted SQL in the interaction is correct.

Implementation Details. We utilize PyTorch (Paszke et al. 2019) to implement our proposed model. For the model without BERT, we initialize word embedding using GloVe (Pennington, Socher, and Manning 2014) and model parameters from a uniform distribution and set the hidden size as

Dataset	# sequence	# user questions	# databases	# domain	# tables	Avg. len	Vocab	Avg. turns
SParC	4,298	12,726	200	138	1,020	8.1	3,794	3.0
CoSQL	3,007	15,498	200	138	1,020	11.2	9,585	5.2

Table 1: Comparison of the statistics of cross-domain context-dependent Text-to-SQL datasets.

Model	SParC				CoSQL			
	Question Match.		Interaction Match.		Question Match.		Interaction Match.	
	Dev	Test	Dev	Test	Dev	Test	Dev	Test
SyntaxSQL-con	18.5	20.2	4.3	5.2	15.1	14.1	2.7	2.2
CD-Seq2Seq	21.9	23.2	8.1	7.5	13.8	13.9	2.1	2.6
EditSQL	33.0	-	16.4	-	22.2	-	5.8	-
RichContext	41.8	-	20.6	-	33.5	-	9.6	-
Ours	42.4	-	21.9	-	34.5	-	11.0	-
EditSQL + BERT	47.2	47.9	29.5	25.3	39.9	40.8	12.3	13.7
RichContext + BERT	52.6	-	29.9	-	41.0	-	14.0	-
Ours + BERT	**54.1**	**55.8** (↑ 7.9)	**35.2**	**30.8** (↑ 5.5)	**45.7**	**46.8** (↑ 6.0)	**19.5**	**17.0** (↑ 3.3)

Table 2: Performance of various methods over questions (question match) and interactions (interaction match) in SParC and CoSQL.

300 for each **LSTM** layer. For the model with BERT, we use Adam (Kingma and Ba 2015) to minimize the token level cross-entropy loss and set the learning rate as 1e-3 on all modules except for the BERT fine-turn stage, for which a learning rate of 1e-5 is used instead. In particular, we use the pretrained small uncased BERT model with the 768 hidden size. For the reranker module, we first extract negative samples from the incorrect queries in the training set, and utilize the downsampling strategy to deal with label imbalance to make positive and negative samples balanced.

Reranker Details. As the generated SQL queries may not conform to the SQL grammar, the reranker model input is the utterance and generated SQL prediction, the aim is to classificate the current SQL whether matches the current utterance. The reranker leverage training data that extract from the intermediate results during downstream model training by BERT model. More specifically, after training the model for a period of time, when the performance on the entire training set is similar to that on the validation set, we extract negative samples from the incorrect queries in each small batch of beam in the training set, which ensure the distribution of candidates in the training set is similar to that in the validation set. We found this helped us get the best reranking performance. For reranker training, we utilize the down sampling strategy to deal with label imbalance by removing some negative examples to make positive and negative samples balanced. By training the reranker in the training set, the reranker module can decrease the matching scores of inconsistent SQL queries, and increase the consistent ones, improving the final results further in case that the correct query is generated in the beam search. The reranking module is trained after the main training model finishes, and is used to post-process the output generated in the beam during inference.

Compared Methods. We compare the proposed method against the following state-of-the-art models:

- **SyntaxSQL-con** is modified from the original context-independent SyntaxSQLNet (Yu et al. 2018a), which encodes the utterance and the associated SQL in interaction history. It employs a column attention mechanism to compute representations for the previous question and SQL.
- **CD-Seq2Seq** (Yu et al. 2019b) is based on sequence-to-sequence modelling extended with the turn-level history encoder proposed in (Suhr, Iyer, and Artzi 2018), which modifies the database schema encoder, and takes the bag-of-words representation for column headers as input to perform SQL generation.
- **EditSQL** (Zhang et al. 2019) is an editing-based encoder-decoder model, which utilizes the interaction history by editing the previous predicted query to improve generation quality. Note that it also uses the implicit relation exploration module based on general attention during the encoding stage.
- **RichContext** (Liu et al. 2020) reports state-of-the-art performance on the dev set of SParC and CoSQL and the model is based on rich context modeling methods. We select their best performance to compare with ours.

Overall Performance

As shown in Table 2, we compare the performance of the proposed model with other state-of-the-art models on the SParC and CoSQL datasets. We observe that our method outperforms all existing models on all evaluation metrics. Our model achieves the performance of 54.1%/45.7% in question match and 35.2%/19.5% in interaction match on

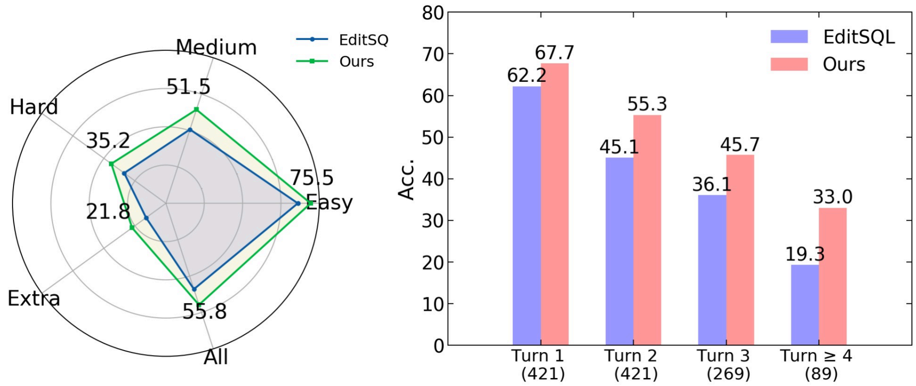

Figure 3: Performance split by different difficulty levels (left) and different turns (right) on SParC.

Model	SParC		CoSQL	
	Ques.	Int.	Que.	Int.
DCRI	48.0	29.9	40.0	12.3
DCRE	48.3	30.2	40.6	13.0
DCRI + DCRE	51.1	31.3	41.5	14.0
+Gate-based Decay	51.6	31.9	42.7	15.4
+Schedule-based Decay	52.0	32.3	42.6	16.1
Reranker	53.7	34.6	45.0	18.6
Feature Enhanced Reranker	**54.1**	**35.2**	**45.7**	**19.5**

Table 3: Ablation study of proposed method over Ques. (question match) and Int. (interaction match) in dev set.

the dev set, which is a strong model for the context-independent cross-domain Text-to-SQL generation. We can see that our proposed method outperforms the second published best algorithm (EditSQL) on the SParC / CoSQL test dataset by approximately 7.9%/5.5% in question match and 6.0%/3.3% in interaction match, presenting new state-of-the-art results on benchmarks.

In order to distinguish the performance of models of different complexities, following (Yu et al. 2018b), we evaluate the models on the dev dataset with four-levels of difficulty: easy, medium, hard, extra hard. Table 3(left) shows the comparison between our proposed model and EditSQL, which demonstrates that our method is better than EditSQL. In particular, in SparC extra hard type, we also achieve a significant improvement. Furthermore, to understand how the models perform as the interaction proceeds, Figure 3(right) shows the performance split by turns on the SparC dev set. As the conversation proceeds, it is becomes more difficult to generate SQL using historical information, which leads to a decrease on accuracy. It shows that although our model is still affected by turns, in SParC, our accuracy rate at the fourth turn is still 33%, which is close to that of the third turn of the previous method.

Detailed Analysis

We investigate the impact of different components on the performance. Specifically, we analyze the following three components: First, when the relationship is established, two methods are used, Dynamic Context Representation over Implicit Relations (DCRI) and Dynamic Context Representation over Explicit Relations (DCRE). Here we perform them without decay to understand the difference. Second, two methods are used for memory decay: gate-based and schedule-based. Third, two types of reranker models, with or without task-related vectors, are further compared and we show that the representation learned from the model is beneficial to the discrimination of the reranker. As presented in Table 3, we can conclude that each module in our proposed method improve the performance of generation. It is worth noting that the decay mechanism improves the interaction match of CoSQL more obviously, we attribute this to the more frequent switching of intentions in the CoSQL dataset.

Conclusions

This paper investigates context-dependent semantic parsing. We present a dynamic graph framework that can effectively model contextual utterances, tokens, database schemas, and their complicated relations as the interaction with databases proceeds. The framework employs dynamic memory decay mechanisms to introduce inductive bias to construct enriched contextual relation representation at both the utterance and token level, rendering a flexible framework to model different levels of context in the dynamic multi-turn scenario. The proposed model achieves new state-of-the-art performance on two large-scale benchmarks, the SParC and CoSQL datasets.

Acknowledgments

We would like to thank all of the anonymous reviewers for their invaluable suggestions and helpful comments. We thank Songhe Wang, Tao Yu, Bo Pang for their help with the evaluation on SparC and CoSQL. This work was supported by the National Natural Science Foundation of China under Grants 61732011 and 61876127. The last author's research is supported by NSERC Discovery Grants and Discovery Accelerator Supplements (DAS).

References

Bengio, S.; Vinyals, O.; Jaitly, N.; and Shazeer, N. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *NeurIPS*.

Berant, J.; and Liang, P. 2014. Semantic Parsing via Paraphrasing. In *ACL*.

Bogin, B.; Berant, J.; and Gardner, M. 2019. Representing Schema Structure with Graph Neural Networks for Text-to-SQL Parsing. In *ACL*.

Devlin, J.; Chang, M.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT*, 4171–4186.

Dong, L.; and Lapata, M. 2018. Coarse-to-Fine Decoding for Neural Semantic Parsing. In *ACL*.

Fried, D.; Andreas, J.; and Klein, D. 2018. Unified Pragmatic Models for Generating and Following Instructions. In *NAACL-HLT*.

Guo, J.; Zhan, Z.; Gao, Y.; Xiao, Y.; Lou, J.; Liu, T.; and Zhang, D. 2019. Towards Complex Text-to-SQL in Cross-Domain Database with Intermediate Representation. In *ACL*.

Guu, K.; Pasupat, P.; Liu, E. Z.; and Liang, P. 2017. From Language to Programs: Bridging Reinforcement Learning and Maximum Marginal Likelihood. In *ACL*.

Hemphill, C. T.; Godfrey, J. J.; and Doddington, G. R. 1990. The ATIS Spoken Language Systems Pilot Corpus. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley*.

Hochreiter, S.; and Schmidhuber, J. 1997. Long Short-Term Memory. *Neural Computation* .

Huang, H.; Choi, E.; and Yih, W. 2019. FlowQA: Grasping Flow in History for Conversational Machine Comprehension. In *ICLR*.

Hwang, W.; Yim, J.; Park, S.; and Seo, M. 2019. A Comprehensive Exploration on WikiSQL with Table-Aware Word Contextualization. *Corr* .

Iyer, S.; Konstas, I.; Cheung, A.; Krishnamurthy, J.; and Zettlemoyer, L. 2017. Learning a Neural Semantic Parser from User Feedback. In *ACL*.

Iyyer, M.; Yih, W.; and Chang, M. 2017. Search-based Neural Structured Learning for Sequential Question Answering. In *ACL*.

Kingma, D. P.; and Ba, J. 2015. Adam: A Method for Stochastic Optimization. In *ICLR*.

Li, F.; and Jagadish, H. V. 2014. Constructing an Interactive Natural Language Interface for Relational Databases. *Proc. VLDB Endow.* 8: 73–84.

Liu, Q.; Chen, B.; Guo, J.; Lou, J.; Zhou, B.; and Zhang, D. 2020. How Far are We from Effective Context Modeling? An Exploratory Study on Semantic Parsing in Context. In *IJCAI*.

Long, R.; Pasupat, P.; and Liang, P. 2016. Simpler Context-Dependent Logical Forms via Model Projections. In *ACL*.

McCann, B.; Keskar, N. S.; Xiong, C.; and Socher, R. 2018. The Natural Language Decathlon: Multitask Learning as Question Answering. *CoRR* abs/1806.08730.

Miller, S.; Stallard, D.; Bobrow, R. J.; and Schwartz, R. M. 1996. A Fully Statistical Approach to Natural Language Interfaces. In *ACL*.

Pareja, A.; Domeniconi, G.; Chen, J.; Ma, T.; Suzumura, T.; Kanezashi, H.; Kaler, T.; Schardl, T. B.; and Leiserson, C. E. 2020. EvolveGCN: Evolving Graph Convolutional Networks for Dynamic Graphs. In *AAAI*, 5363–5370.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; Desmaison, A.; Köpf, A.; Yang, E.; DeVito, Z.; Raison, M.; Tejani, A.; Chilamkurthy, S.; Steiner, B.; Fang, L.; Bai, J.; and Chintala, S. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *NeurIPS*.

Pennington, J.; Socher, R.; and Manning, C. D. 2014. Glove: Global Vectors for Word Representation. In *EMNLP*.

Suhr, A.; and Artzi, Y. 2018. Situated Mapping of Sequential Instructions to Actions with Single-step Reward Observation. In *ACL*.

Suhr, A.; Iyer, S.; and Artzi, Y. 2018. Learning to Map Context-Dependent Sentences to Executable Formal Queries. In Walker, M. A.; Ji, H.; and Stent, A., eds., *NAACL-HLT*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is All you Need. In *NeurIPS*.

Wang, B.; Shin, R.; Liu, X.; Polozov, O.; and Richardson, M. 2020. RAT-SQL: Relation-Aware Schema Encoding and Linking for Text-to-SQL Parsers. In *ACL*.

Wong, Y. W.; and Mooney, R. J. 2007. Learning Synchronous Grammars for Semantic Parsing with Lambda Calculus. In *ACL*.

Yaghmazadeh, N.; WANG, Y.; Dillig, I.; and Dillig, T. 2017. SQLizer: query synthesis from natural language. *Proceedings of the ACM on Programming Languages* 1: 1 – 26.

Yin, P.; and Neubig, G. 2017. A Syntactic Neural Model for General-Purpose Code Generation. In *ACL*.

Yin, P.; and Neubig, G. 2019. Reranking for Neural Semantic Parsing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics.

Yu, T.; Yasunaga, M.; Yang, K.; Zhang, R.; Wang, D.; Li, Z.; and Radev, D. R. 2018a. SyntaxSQLNet: Syntax Tree Networks for Complex and Cross-Domain Text-to-SQL Task. In *EMNLP*.

Yu, T.; Zhang, R.; Er, H.; Li, S.; Xue, E.; Pang, B.; Lin, X. V.; Tan, Y. C.; Shi, T.; Li, Z.; Jiang, Y.; Yasunaga, M.; Shim, S.; Chen, T.; Fabbri, A. R.; Li, Z.; Chen, L.; Zhang, Y.; Dixit, S.; Zhang, V.; Xiong, C.; Socher, R.; Lasecki, W. S.; and Radev, D. R. 2019a. CoSQL: A Conversational Text-to-SQL Challenge Towards Cross-Domain Natural Language Interfaces to Databases. In Inui, K.; Jiang, J.; Ng, V.; and Wan, X., eds., *EMNLP-IJCNLP*.

Yu, T.; Zhang, R.; Yang, K.; Yasunaga, M.; Wang, D.; Li, Z.; Ma, J.; Li, I.; Yao, Q.; Roman, S.; Zhang, Z.; and Radev, D. R. 2018b. Spider: A Large-Scale Human-Labeled Dataset for Complex and Cross-Domain Semantic Parsing and Text-to-SQL Task. In *EMNLP*.

Yu, T.; Zhang, R.; Yasunaga, M.; Tan, Y. C.; Lin, X. V.; Li, S.; Er, H.; Li, I.; Pang, B.; Chen, T.; Ji, E.; Dixit, S.; Proctor, D.; Shim, S.; Kraft, J.; Zhang, V.; Xiong, C.; Socher, R.; and Radev, D. R. 2019b. SParC: Cross-Domain Semantic Parsing in Context. In *ACL*.

Zelle, J. M.; and Mooney, R. J. 1996. Learning to Parse Database Queries Using Inductive Logic Programming. In *AAAI*.

Zettlemoyer, L. S.; and Collins, M. 2005. Learning to Map Sentences to Logical Form: Structured Classification with Probabilistic Categorial Grammars. In *UAI*.

Zettlemoyer, L. S.; and Collins, M. 2007. Online Learning of Relaxed CCG Grammars for Parsing to Logical Form. In *EMNLP-CoNLL*.

Zettlemoyer, L. S.; and Collins, M. 2009. Learning Context-Dependent Mappings from Sentences to Logical Form. In *ACL*.

Zhang, R.; Yu, T.; Er, H.; Shim, S.; Xue, E.; Lin, X. V.; Shi, T.; Xiong, C.; Socher, R.; and Radev, D. R. 2019. Editing-Based SQL Query Generation for Cross-Domain Context-Dependent Questions. In *EMNLP-IJCNLP*.

Zhong, V.; Xiong, C.; and Socher, R. 2017. Seq2SQL: Generating Structured Queries from Natural Language using Reinforcement Learning. *CoRR* .