

GRAPPA: Grammar-Augmented Pre-Training for Table Semantic Parsing

Tao Yu^{†*}, Chien-Sheng Wu[§], Xi Victoria Lin[§], Bailin Wang^{‡*}, Yi Chern Tan[†],
 Xinyi Yang[§], Dragomir Radev^{†§}, Richard Socher[§], Caiming Xiong[§]

[§] Salesforce Research

[†] Yale University, [‡]University of Edinburgh

{tao.yu, yichern.tan, dragomir.radev}@yale.edu, bailin.wang@ed.ac.uk
 {jason.wu, xilin, x.yang, rsocber, cxiong}@salesforce.com

Abstract

We present GRAPPA, an effective pre-training approach for table semantic parsing that learns a compositional inductive bias in the joint representations of textual and tabular data. We construct synthetic question-SQL pairs over high-quality tables via a synchronous context-free grammar (SCFG) induced from existing text-to-SQL datasets. We pre-train our model on the synthetic data using a novel text-schema linking objective that predicts the syntactic role of a table field in the SQL for each question-SQL pair. To maintain the model’s ability to represent real-world data, we also include masked language modeling (MLM) on several existing table-and-language modeling datasets to regularize our pre-training process. On four popular fully supervised and weakly supervised table semantic parsing benchmarks, GRAPPA significantly outperforms RoBERTa_{LARGE} as the feature representation layers and establishes new state-of-the-art results on all of them.

1 Introduction

Recent pre-training language models (LMs) such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) achieve tremendous success in a spectrum of natural language processing tasks, including semantic parsing (Zettlemoyer and Collins, 2005; Zhong et al., 2017; Yu et al., 2018c). For example, by incorporating the BERT encoder, Hwang et al. (2019) is able to boost the parser accuracy on WIKISQL (Zhong et al., 2017) by more than 15%, approaching the upper-bound performance. These advances have shifted the focus from building domain-specific semantic parsers (Zelle and Mooney, 1996; Zettlemoyer and Collins, 2005; Artzi and Zettlemoyer, 2013; Berant and Liang,

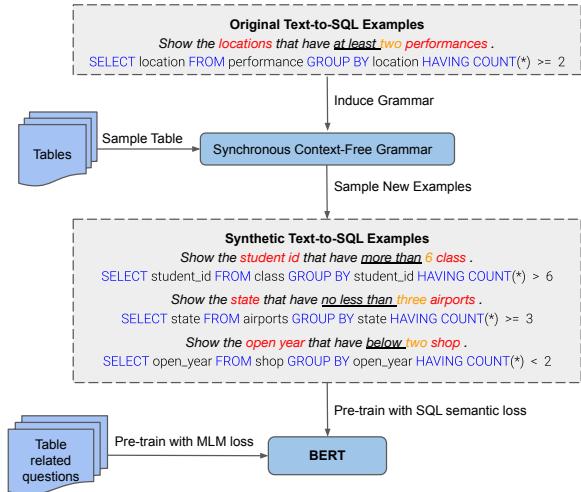


Figure 1: An overview of GRAPPA pre-training approach. We first induce a SCFG given some examples in SPIDER. We then sample from this grammar given a large amount of tables to generate new synthetic examples. Finally, GRAPPA is pre-trained on the synthetic data using SQL semantic loss and a small amount of table related utterances using MLM loss.

2014; Li and Jagadish, 2014) to cross-domain semantic parsing (Zhong et al., 2017; Yu et al., 2018c; Herzig and Berant, 2018; Dong and Lapata, 2018; Wang et al., 2020; Lin et al., 2020).

Despite significant gains on WIKISQL, the overall improvement on the SPIDER benchmark (Yu et al., 2018c) is limited. One possible reason is that SPIDER contains databases with multiple tables and more complex SQL queries. Language models pre-trained using unstructured text data such as Wikipedia and Book Corpus are exposed to a significant domain shift when applied to such tasks. To close this gap, our goal is to learn contextual representations jointly from structured tabular data and unstructured natural language sentences.

Closest to our work, Yin et al. (2020a) and Herzig et al. (2020a) use a large amount of web tables and their textual context (26M and 21M table-

* This work was mainly done during Tao and Bailin’s internship at Salesforce Research.

sentence pairs) for pre-training. However, these two approaches suffer from three disadvantages: 1) pre-training on such a large amount of noisy data is slow and expensive; 2) the natural language and tables in the training data are loosely connected; 3) the standard masked language modeling objective is weak at capturing the semantic groundings specific to tabular data.

In this paper, we propose a novel grammar-augmented pre-training framework for table semantic parsing (GRAPPA). Inspired by previous work on data synthesis for semantic parsing (Berant and Liang, 2014; Wang et al., 2015b; Jia and Liang, 2016; Yu et al., 2018b; Andreas, 2020), we induce a synchronous context-free grammar (SCFG) specific to mapping natural language to SQL queries from existing text-to-SQL datasets, which covers most commonly used question-SQL patterns. As shown in Figure 1, from a text-to-SQL example we can create a question-SQL template by abstracting over mentions of schema components (tables and fields) and values. By executing this template on randomly selected tables we can create a large number of synthetic question-SQL pairs. We train GRAPPA on these synthetic question-SQL pairs and their corresponding tables using a novel text-schema linking objective that predicts the syntactic role of a table column in the SQL for each pair. This way we encourage the model to identify natural language phrases that could be grounded to logical form constituents, which is critical for most table semantic parsing tasks.

GRAPPA usually achieves better downstream task performance by pre-training with fewer steps (less than 5 epochs). One possible reason is that it can easily overfit into the inductive biases introduced by the synthetic data, which is consistent with finding from Zhang et al. (2019b) and Herzig et al. (2020a). To prevent overfitting, we include the masked-language modelling (MLM) loss to regularize the pre-training over synthetic data. To this end, we utilize several large-scale, high-quality table-and-language datasets as input and carefully balance between preserving the original natural language representations and injecting the compositional inductive bias.

We pre-train GRAPPA using 475k synthetic examples and 391.5k examples from existing table-and-language datasets. Compared to Yin et al. (2020a) and Herzig et al. (2020a), our pre-training approach dramatically reduces the training time

and GPU cost. We evaluate on four popular semantic parsing benchmarks in both fully supervised and weakly supervised settings. GRAPPA consistently achieves new state-of-the-art results on all of them, significantly outperforming all previously reported results. Our pre-training method does not add any additional operations on the top of a BERT-like architecture. This enables our pre-trained model to be easily integrated with any existing state-of-the-art models as BERT does.¹

2 Methodology

2.1 Motivation

Semantic parsing data is compositional because utterances are usually related to some formal representations such as logic forms and SQL queries. Numerous prior works(Berant and Liang, 2014; Wang et al., 2015a; Jia and Liang, 2016; Iyer et al., 2017; Yu et al., 2018b; Andreas, 2020) have demonstrated the benefits of augmenting data using context-free grammar. The augmented examples can be used to teach the model to generalize beyond the given training examples.

However, data augmentation becomes more complex and less beneficial if we want to apply it to generate data for a random domain. As mentioned in Section 1, Zhang et al. (2019b); Herzig et al. (2020a) note that this approach of utilizing augmented data doesn't result in a significant performance gain in cross-domain semantic parsing end tasks. The most likely reason for this is that models tend to overfit to the canonical input distribution especially the generated utterances are very different compared with the original ones.

To address this problem, in Section 2.3, we also include a small set of table related utterances in our pre-training data. We add an MLM loss on them as a regularization factor, which requires the model to balance between real and synthetic examples during the pre-training. We note that this consistently improves the performance on all downstream semantic parsing tasks (more detail in Section 4).

2.2 Data Augmentation with Synchronous Context-Free Grammar

We follow Jia and Liang (2016) to design our SCFG and apply it on a large amount of tables to populate new examples. For example, as shown in Figure

¹Our model is publicly available at http://huggingface.co/Salesforce/grappa_large_jnt.

1, by replacing substitutable column mentions (“locations”), table mentions (“performance”), values (“two”), and SQL logic phrases (“at least”) with the other possible candidates in the same group, our grammar generates new synthetic text-to-SQL examples with the same underlying SQL logic template. We then pre-train BERT on the augmented examples to force it to discover substitutable fragments and learn the underlying logic template so that it is able to generalize to other similar questions. Meanwhile, BERT also benefits from pre-training on a large number of different columns, table names, and values in the generated data, which could potentially improve schema linking in semantic parsing tasks.

Grammar induction To induce a cross-domain SCFG, we study examples in SPIDER since it is the only publicly available dataset that includes the largest number of complex text-to-SQL examples in different domains. To further show the generality of our approach, we do not develop different SCFG for each downstream task. Given a set of (x, y) pairs in SPIDER, where x and y are the utterance and SQL query respectively. We first define a set of non-terminal symbols for table names, column names, cell values, operations, etc. For example (see Table 1), we group operations such as MAX, MIN, COUNT, AVG, SUM as a non-terminals called AGG. We can also replace the entities/phrases with their non-terminal types in SQL query to generate a SQL production rule β . Then, we group (x, y) pairs by similar SQL production rule β . We select 90 most frequent β and randomly select roughly 4 samples each, and manually align entities/phrases with their corresponding non-terminal types to collect natural language templates, α .

Data augmentation With $\langle \alpha, \beta \rangle$ pairs, we can simultaneously generate pseudo natural questions and corresponding SQL queries given a new table or database. We first sample a production rule, and replace its non-terminals with one of corresponding terminals. For example, we can map the non-terminal AGG to MAX and “maximum” for the SQL query and the natural language sentence, respectively.

We use WIKITABLES (Bhagavatula et al., 2015), which contains 1.6 million high-quality relational Wikipedia tables. We remove tables with exactly the same column names and get about 340k tables and generate 413k question-SQL pairs given these

tables. Also, we generate another 62k question-SQL pairs using tables and databases in the training sets of SPIDER and WIKISQL. In total, our final pre-training dataset includes 475k question-SQL examples.

We note that SCFG is usually crude (Andreas, 2020) especially when it is applied to augment data for different domains. In this work we don’t focus on how to develop a better SCFG that generates more natural utterances. We see this as a very interesting future work to explore. Despite the fact that the SCFG is crude, our downstream task experiments show that it could be quite effective if some pre-training strategies are applied.

2.3 Table Related Utterances

As discussed in Section 2.1, in order to leverage the grammar-augmented data in table pre-training, we discover that combining MLM loss on real table related utterances significantly improves end task performance. We collected seven high quality datasets for textual-tabular data understanding (Table 8 in the Appendix), including TabFact (Chen et al., 2019), LogicNLG (Chen et al., 2020a), HybridQA (Chen et al., 2020b), WikiSQL (Zhong et al., 2017), WikiTableQuestions (Pasupat and Liang, 2015), ToTTo (Parikh et al., 2020), and Spider (Yu et al., 2018c). All of them contain Wikipedia tables or databases and the corresponding natural language utterances written by humans. Some utterances are questions over tables or databases, and some of them are descriptions of data content. We only use tables and contexts as a pre-training resource and discard all the other human labels such as answers and SQL queries.

2.4 Pre-Training GRAPPA

Unlike all the previous work where augmented data is used in the end task training, we apply the framework to language model pre-training. Training semantic parsers is usually slow, and augmenting a large amount of syntactic pairs directly to the end task training data can be prohibitively slow or expensive. In our work, we formulate text-to-SQL as a multi-class classification task for each column, which can be naturally combined with the MLM objective to pre-train BERT for semantic parsing. Moreover, in this way, the learned knowledge can be easily and efficiently transferred to downstream semantic parsing tasks in the exact same way as BERT (shown in Section 4).

Non-terminals	Production rules
$\text{TABLE} \rightarrow t_i$	1. $\text{ROOT} \rightarrow \langle \text{"For each COLUMN0 , return how many times TABLE0 with COLUMN1 OP0 VALUE0 ?"}, \text{SELECT COLUMN0 , COUNT (*) WHERE COLUMN1 OP0 VALUE0 GROUP BY COLUMN0 } \rangle$
$\text{COLUMN} \rightarrow c_i$	
$\text{VALUE} \rightarrow v_i$	
$\text{AGG} \rightarrow \langle \text{MAX, MIN, COUNT, AVG, SUM} \rangle$	
$\text{OP} \rightarrow \langle =, \leq, \neq, \dots, \text{LIKE}, \text{BETWEEN} \rangle$	2. $\text{ROOT} \rightarrow \langle \text{"What are the COLUMN0 and COLUMN1 of the TABLE0 whose COLUMN2 is OP0 AGG0 COLUMN2 ?"}, \text{SELECT COLUMN0 , COLUMN1 WHERE COLUMN2 OP0 (SELECT AGG0 (COLUMN2)) } \rangle$
$\text{SC} \rightarrow \langle \text{ASC, DESC} \rangle$	
$\text{MAX} \rightarrow \langle \text{"maximum"}, \text{"the largest"} \dots \rangle$	
$\leq \rightarrow \langle \text{"no more than"}, \text{"no above"} \dots \rangle$	
\dots	

Table 1: Examples of non-terminals and production rules in our SCFG. Each production rule $\text{ROOT} \rightarrow \langle \alpha, \beta \rangle$ is built from some $(x, y) \in \mathcal{D}$ by replacing all terminal phrases with non-terminals. t_i , c_i , and v_i stand for any table name, column name, entry value respectively.

GRAPPA is initialized by RoBERTa_{LARGE} (Liu et al., 2019) and further pre-train on the synthetic data with SQL semantic loss and table-related data with MLM loss. We follow Hwang et al. (2019) to concatenate a user utterance and the column headers into a single flat sequence separated by the $\langle /s \rangle$ token. The user utterance can be either one of the original human utterances collected from the aggregated datasets or the canonical sentences sampled from the SCFG. We add the table name at the beginning of each column if there are some complex schema inputs involving multiple tables. We employ two objective functions for language model pre-training: 1) masked-language modelling (MLM), and 2) SQL semantic prediction (SSP).

MLM objective Intuitively, we would like to have a self-attention mechanism between natural language and table headers. We conduct masking for both natural language sentence and table headers. A small part of the input sequence is first replaced with the special token $\langle \text{mask} \rangle$. The MLM loss is then computed by the cross-entropy function on predicting the masked tokens. We follow the default hyperparameters from (Devlin et al., 2019) with a 15% masking probability.

SSP objective With our synthetic natural language sentence and SQL query pairs, we can add an auxiliary task to train our column representations. The proposed task is, given a natural language sentence and table headers, to predict whether a column appears in the SQL query and what operation is triggered. We then convert all SQL sequence labels into operation classification labels for each column. For example in the Figure 1, the operation classification label of the column “locations” is SELECT AND GROUP BY HAVING. For columns that appear in a nested query, we append the corresponding nested keywords before each operation. For instance, since the column “nation” is

in the INTERSECT nested sub-query, its labels are INTERSECT SELECT and INTERSECT GROUP BY. For each additional table name, we only predict if it is selected for constructing possible JOINS in the FROM clause. In total, there are 254 potential classes for operations in our experiments.

We use the encoding of the special token $\langle /s \rangle$ right before each column or table name to predict its corresponding operations. We also apply a 2-layer feed-forward network followed by a GELU activation layer (Hendrycks and Gimpel, 2016) and a normalization layer (Ba et al., 2016) to the output representations. Formally, we compute the final vector representation of each column y_i by:

$$\begin{aligned} \mathbf{h} &= \text{LayerNorm}(\text{GELU}(W_1 \cdot \mathbf{x}_i)) \\ \mathbf{y}_i &= \text{LayerNorm}(\text{GELU}(W_2 \cdot \mathbf{h})) \end{aligned}$$

The representation of the special token $\langle /s \rangle$ y_i for each column is then used to compute the cross-entropy loss. We sum losses from all columns in each training example for back-propagation. For samples from the aggregated datasets, we only compute the MLM loss to update our model. For samples from the synthetic data we generated, we compute only SSP loss to update our model.

3 Experiments

We conduct experiments on four *cross-domain* table semantic parsing tasks, where generalizing to unseen tables/databases at test time is required. Following previous work, we experiment with two different settings of table semantic parsing, fully supervised and weakly supervised setting. The data statistics and examples on each task are shown in Table 2 and Table 7 in the Appendix respectively.

3.1 Supervised Semantic Parsing

We first evaluate GRAPPA on two supervised semantic parsing tasks. In a supervised semantic

Task & Dataset	# Examples	Resource	Annotation	Cross-domain
SPIDER (Yu et al., 2018c)	10,181	database	SQL	✓
Fully-sup. WIKISQL (Zhong et al., 2017)	80,654	single table	SQL	✓
WIKITABLEQUESTIONS (Pasupat and Liang, 2015)	2,2033	single table	answer	✓
Weakly-sup. WIKISQL (Zhong et al., 2017)	80,654	single table	answer	✓

Table 2: Overview of four table-based semantic parsing and question answering datasets in fully-supervised (top) and weakly-supervised (bottom) setting used in this paper. More details in Section 3

parsing scenario, given a question and a table or database schema, a model is expected to generates the corresponding program.

SPIDER SPIDER (Yu et al., 2018c) is a large-scale complex and cross-domain text-to-SQL dataset. It consists of 10k complex question-query pairs where many of the SQL queries contain multiple SQL keywords. It also includes 200 databases where multiple tables are joined via foreign keys. For the baseline model, we use RAT-SQL + BERT (Wang et al., 2020) which is the state-of-the-art model according to the official leaderboard. We followed the official Spider evaluation to report set match accuracy.

Fully-sup. WIKISQL WIKISQL (Zhong et al., 2017) is a collection of over 80k questions and SQL query pairs over 30k Wikipedia tables. In the original setting, each question is annotated a SQL query in the context of a table. We use (Guo and Gao, 2019), a competitive model on WIKISQL built on SQLova (Hwang et al., 2019), as our base model. The model employs BERT to encode questions and column names. We adapt the same set of hyperparameters including batch size and maximum input length as in Guo and Gao (2019). For a fair comparison, we only consider single models without execution-guided decoding and report execution accuracy.

3.2 Weakly-supervised Semantic Parsing

We also consider weakly-supervised semantic parsing tasks, which are very different from SQL-guided learning in pre-training. In this setting, a question and its corresponding answer are given, but the underlying meaning representation (e.g., SQL queries) are unknown.

WIKITABLEQUESTIONS This dataset contains 18,496 question-denotation pairs over 2,018 tables (Pasupat and Liang, 2015). The questions involve a variety of operations such as comparisons, superlatives, and aggregations, where some of them are hard to answered by SQL queries. All tables

in WIKITABLEQUESTIONS are single Wikipedia tables as in WIKISQL.

We used the model proposed by Wang et al. (2019) which is the state-of-the-art parser on this task. This model is a two-stage approach that first predicts a partial “abstract program” and then refines that program while modeling structured alignments with differential dynamic programming. The original model uses GloVe (Pennington et al., 2014) as word embeddings. We modified their implementation to encode question and column names in the same way as we do in our fine-tuning method that uses RoBERTa and GRAPPA.

Weakly-sup. WIKISQL In the weakly-supervised setting of WIKISQL, only the answers (i.e., execution results of SQL queries) are available. We also employed the model proposed by Wang et al. (2019) as our baseline for this task. We made the same changes and use the same experiment settings as described in the previous section for WIKITABLEQUESTIONS.

3.3 Implementation

For fine-tuning RoBERTa, we modify the code of RoBERTa implemented by Wolf et al. (2019) and follow the hyperparameters for fine-tuning RoBERTa on RACE tasks and use batch size 24, learning rate $1e-5$, and the Adam optimizer (Kingma and Ba, 2014). We fine-tune GRAPPA for 300k steps on eight 16GB Nvidia V100 GPUs. The pre-training procedure can be done in less than 10 hours. For all downstream experiments using GRAPPA or RoBERTa, we always use a BERT specific optimizer to fine-tune them with a learning rate of $1e-5$, while using a model-specific optimizer with the respective learning rate for the rest of the base models.

4 Experimental Results

We conducted experiments to answer the following two questions: 1) Can GRAPPA provide better representations for table semantic parsing tasks? 2) What is the benefit of two pre-training objectives,

Models	Dev.	Test
Global-GNN (Bogin et al., 2019)	52.7	47.4
EditSQL w. BERT (Zhang et al., 2019b)	57.6	53.4
IRNet w. BERT (Guo et al., 2019)	61.9	54.7
RYANSQL w. BERT (Choi et al., 2020)	70.6	60.6
TranX w. TaBERT (Yin et al., 2020a)	64.5	-
RAT-SQL (Wang et al., 2019)	62.7	57.2
w. BERT-large	69.7	65.6
w. RoBERTa-large	69.57	-
w. GRAPPA (MLM)	71.08	-
w. GRAPPA (SSP)	73.57	67.72
w. GRAPPA (MLM+SSP)	73.43	69.63

Table 3: Performance on SPIDER. We use RAT-SQL + BERT (Wang et al., 2019) as our base model. We run each model three times by varying random seeds, and the average scores are shown.

namely MLM and SSP? Since GRAPPA is initialized by RoBERTa, we answer the first question by directly comparing the performance of base parser augmented with GRAPPA and RoBERTa on table semantic parsing tasks. For the second question, we report the performance of GRAPPA trained with MLM, SSP and also a variant with both of them (MLM+SSP).

We report results on the four aforementioned tasks in Tables 3, 4, 5, and 6 respectively. Overall, base models augmented with GRAPPA consistently outperforms the ones with RoBERTa across all four tasks. In most cases, the combined objective of MLM+SSP helps GRAPPA achieve better performance when compared with independently using MLM and SSP. Detailed results for each task are discussed as follows.

SPIDER Results on SPIDER are shown in Table 3. When augmented with GRAPPA, the model achieves significantly better performance compared with the baselines using BERT and RoBERTa. Our best model, GRAPPA with MLM+SSP achieves the new state-of-the-art performance, surpassing previous one (RAT-SQL+BERT-large) by a margin of 4%. Notably, most previous top systems use pre-trained contextual representations (e.g., BERT, TabERT), indicating the importance of such representations for the cross-domain parsing task.

Through qualitative analysis, we find that RAT-SQL+GRAPPA is better at handling more complex questions. Specifically, we manually inspected evaluation outputs. In some cases where the column name is unusual, GRAPPA can map it to the right position in the SQL output. For example, RoBERTa fails to link “founded” mentioned in the question to “independent year” in the SQL query,

Models	Dev.	Test
Zhong et al. (2017)	60.8	59.4
Xu et al. (2017)	69.8	68.0
Yu et al. (2018a)	74.5	73.5
Dong and Lapata (2018)	79.0	78.5
Shi et al. (2018)	84.0	83.7
Hwang et al. (2019)	87.2	86.2
He et al. (2019)	89.5	88.7
Lyu et al. (2020)	89.1	89.2
Guo and Gao (2019)	90.3	89.2
w. RoBERTa-large	91.2	90.6
w. GRAPPA (MLM)	91.4	90.7
w. GRAPPA (SSP)	91.2	90.7
w. GRAPPA (MLM+SSP)	91.2	90.8
w. RoBERTa-large (10k)	79.6	79.2
w. GRAPPA (MLM+SSP) (10k)	82.3	82.2

Table 4: Performance on fully-sup. WIKISQL. All results are on execution accuracy without execution-guided decoding. We run each model twice and the results are the same. Results on models trained on 10k (18%) WIKISQL examples are shown at the bottom.

but GRAPPA is able to correctly predict the SQL for this example.

Fully sup. WIKISQL Results on WIKISQL are shown in Table 4. All GRAPPA models achieve nearly the same performance as RoBERTa. We suspect it is the relatively large training size and easy SQL pattern of WIKISQL make the improvement hard, comparing to SPIDER. Hence, we set up a low-resource setting where we only use 10k examples from the training data. As shown in the bottom two lines of Table 4, GRAPPA improves the performance of the SQLova model by 3.0% compared to RoBERTa, indicating that GRAPPA can make the base parser more sample-efficient.

Moreover, we analyze the predictions in detail, and find that GRAPPA significantly improves the accuracies on column selection in WHERE (from 88.0% to 91.7%) and operator prediction (from 85.7% to 93.2%). This demonstrates that GRAPPA enhances RoBERTa’s capability on schema mapping and logic operation prediction.

WIKITABLEQUESTIONS Results on WIKITABLEQUESTIONS are shown in Table 5. By using RoBERTa and GRAPPA to encode question and column inputs, the performance of Wang et al. (2019) can be boosted significantly (>6%). Compared with RoBERTa, our best model with GRAPPA (MLM+SSP) can further improve the performance by 1.8%, leading to a new state-of-the-art performance on this task. Similar to the low-resource experiments for WIKISQL, we also show the performance of the model when trained with only 10% of the training data. As

Models	Dev.	Test
Pasupat and Liang (2015)	37.0	37.1
Neelakantan et al. (2016)	34.1	34.2
Haug et al. (2017)	-	34.8
Zhang et al. (2017)	40.4	43.7
Liang et al. (2018)	42.3	43.1
Dasigi et al. (2019)	42.1	43.9
Agarwal et al. (2019)	43.2	44.1
Herzig et al. (2020b)	-	48.8
Yin et al. (2020b)	52.2	51.8
Wang et al. (2019)	43.7	44.5
w. RoBERTa-large	50.7(+7.0)	50.9(+6.4)
w. GRAPPA (MLM)	51.5(+7.8)	51.7(+7.2)
w. GRAPPA (SSP)	51.2(+7.5)	51.1(+6.6)
w. GRAPPA (MLM+SSP)	51.9(+8.2)	52.7(+8.2)
w. RoBERTa-large $\times 10\%$	37.3	38.1
w. GRAPPA (MLM+SSP) $\times 10\%$	40.4(+3.1)	42.0(+3.9)

Table 5: Performance on WIKITABLEQUESTIONS. We use Wang et al. (2019) as a base model. Results trained on 10% of the data are shown at the bottom.

Models	Dev.	Test
Liang et al. (2018)	72.2	72.1
Agarwal et al. (2019)	74.9	74.8
Min et al. (2019)	84.4	83.9
Herzig et al. (2020b)	85.1	83.6
Wang et al. (2019)	79.4	79.3
w. RoBERTa-large	82.3 (+2.9)	82.3 (+3.0)
w. GRAPPA (MLM)	83.3 (+3.9)	83.5 (+4.2)
w. GRAPPA (SSP)	83.5(+4.1)	83.7 (+4.4)
w. GRAPPA (MLM+SSP)	85.9 (+6.5)	84.7 (+5.4)

Table 6: Performance on weakly-sup. WIKISQL. We use Wang et al. (2019) as our base model.

shown at the bottom two lines Table 5, GRAPPA (MLM + SSP) obtains much better performance than RoBERTa, again showing its superiority of providing better representations.

Weakly sup. WIKISQL Results on weakly supervised WIKISQL are shown in Table 6. GRAPPA with MLM+SSP again achieves the best performance when compared with other baselines, obtain the new state-of-the-art results of 84.7% on this task. It is worth noting that our best model here is also better than many models trained in the fully-supervised setting in Table 4. This suggests that inductive biases injected in pre-trained representation of GRAPPA can significantly help combat the issue of spurious programs introduced by learning from denotations (Pasupat and Liang, 2015; Wang et al., 2019) when gold programs are not available.

5 Analysis

Pre-training objectives GRAPPA trained with both MLM and SSP loss consistently outperforms the one trained with one of them. For example, we can see a 1% gain on the SPIDER dev set by using

GRAPPA (MLM) in Table 3. By pre-training on the synthetic text-to-SQL examples, GRAPPA (SSP) enlarges the margin by another 2.5%, resulting in a performance comparable to GRAPPA (MLM+SSP). However, GRAPPA (SSP) performs significantly worse than GRAPPA (MLM+SSP) on the hidden test set, which is what we expected. By combining the MLM loss on real natural language and SQL semantic loss on the synthetic data, the pre-trained model can benefit from both training signals and consistently improve the end task performance.

Generalization As mentioned in Section 2.2, we design our SCFG solely based on SPIDER, and then sample from it to generate synthetic examples. Despite the fact that GRAPPA pre-trained on such corpus is optimized to the SPIDER data distribution, which is very different from WIKISQL and WIKITABLEQUESTIONS, GRAPPA is still able to improve performance on the two datasets. In particular, for WIKITABLEQUESTIONS where the underlying distribution of programs (not necessarily in the form of SQL) are latent, GRAPPA can still help a parser generalize better, indicating GRAPPA can be beneficial for general table understanding even though it is pre-trained on SQL specific semantics.

We only see a relatively small performance gain in WIKISQL. One of the reasons is that questions in WIKISQL are relatively simple compared to the other two tasks, strong models are able to learn good representations by only training on the large amount of WIKISQL data. If only 10k of the data is available in WIKISQL training (See Table 4), GRAPPA significantly outperform RoBERTa by 3%. We believe that higher performance can be achieved if a SCFG is developed specifically for WIKISQL and WIKITABLEQUESTIONS.

Pre-training time and data Our experiments on the SPIDER task show that longer pre-training doesn't improve and can even hurt the performance of the pre-trained model. This also indicates that synthetic data should be carefully used in order to balance between preserving the original BERT encoding ability and injecting compositional inductive bias. The best result on SPIDER is achieved by using GRAPPA pre-trained for only 5 epochs on our relatively small pre-training dataset. We directly use this checkpoint to report results on WIKISQL and WIKITABLEQUESTIONS. Therefore, the conclusion might change for other tasks. Moreover, we encourage future work on studying how the size

and quality of synthetic data would affect the end task performance.

6 Related Work

Textual-tabular data understanding Real-world data exist in both structured and unstructured forms. Recently the field has witnessed a surge of interest in joint textual-tabular data understanding problems, such as table semantic parsing (Zhong et al., 2017; Yu et al., 2018c), question answering (Pasupat and Liang, 2015; Chen et al., 2020b), retrieval (Zhang et al., 2019a), fact-checking (Chen et al., 2019) and summarization (Parikh et al., 2020; Radev et al., 2020). While most work focus on single tables, often obtained from the Web, some have extended modeling to more complex structures such as relational databases (Finegan-Dollak et al., 2018; Yu et al., 2018c; Wang et al., 2020). All of these tasks can benefit from better representation of the input text and different components of the table, and most importantly, an effective contextualization across the two modalities. Our work aims at obtaining high-quality cross-modal representation via pre-training to potentially benefit all downstream tasks.

The closest work to ours are TaBERT (Yin et al., 2020a) and TAPAS (Herzig et al., 2020a). TaBERT is a language model (LM) for joint text-table representation that incorporates table rows most relevant to the text. It shows superior performances on both strong- and weakly- supervised semantic parsing tasks. TAPAS is a text-table LM that models the complete table structure and support arithmetic operations over the cells for weakly supervised table semantic parsing. Both are trained over millions of web tables and relevant but noisy textual context. In comparison, GRAPPA is pre-trained with a novel training objective, over synthetic data plus a much smaller but cleaner collection of text-table datasets. Another notable difference is that GRAPPA does not use table content during training. Table2Vec (Zhang et al., 2019a) and TURL (Deng et al., 2020) are also closely related but both of them focus on table population and retrieval tasks, and their pre-training objectives focus on the contextualization between the table and its caption.

Pre-training for NLP tasks GRAPPA is inspired by recent advances in pre-training for text (Peters et al., 2018; Devlin et al., 2019; Liu et al., 2019;

Lewis et al., 2020b,a; Guu et al., 2020). Seminal work in this area including Elmo (Peters et al., 2018) and BERT (Devlin et al., 2019) shows that textual representation trained using conditional language modeling objectives significantly improves performance on various downstream tasks with further fine-tuning, creating an “ImageNet moment” for NLP. This triggered an exciting line of research work under the themes of (1) cross-modal pre-training that involves text (Lu et al., 2019; Peters et al., 2019; Yin et al., 2020a; Herzig et al., 2020a) and (2) pre-training architectures and objectives catering subsets of NLP tasks (Lewis et al., 2020b,a; Guu et al., 2020). GRAPPA extends these two directions further.

Data augmentation for semantic parsing Our work was inspired by existing work on data augmentation for semantic parsing (Berant and Liang, 2014; Wang et al., 2015a; Jia and Liang, 2016; Iyer et al., 2017; Yu et al., 2018b). Berant and Liang (2014) employed a rule-based approach to generate canonical natural language utterances given a logical form. A paraphrasing model was then used to choose the canonical utterance that best paraphrases the input and to output the corresponding logical form. In contrast, Jia and Liang (2016) and Yu et al. (2018b) used prior knowledge in structural regularities to induce an SCFG and then directly use the grammar to generate more training data, which resulted in a significant improvement on the tasks. Unlike these works which augment a relatively small number of data and use them directly in end task training, we synthesize a large number of texts with SQL logic grounding to each table cheaply and use them for pre-training.

7 Conclusion

In this paper, we proposed a novel and effective pre-training approach for table semantic parsing. We developed a context-free grammar to automatically generate a large amount of question-SQL pairs. Then, we introduced GRAPPA, which is an LM that is pre-trained on the synthetic examples with SQL semantic loss. We discovered that, in order to better leverage augmented data, it is important to add MLM loss on a small amount of table related utterances. Results on four semantic parsing tasks demonstrated that GRAPPA significantly outperforms RoBERTa. While the pre-training method is surprisingly effective in its current form, we view these results primarily as an invitation for more fu-

ture work in this direction. For example, this work relies on hand-crafted grammar which often generates unnatural questions; Further improvements are likely to be made by applying more sophisticated data augmentation techniques.

References

- Rishabh Agarwal, Chen Liang, Dale Schuurmans, and Mohammad Norouzi. 2019. Learning to generalize from sparse and underspecified rewards. In *ICML*.
- Jacob Andreas. 2020. [Good-enough compositional data augmentation](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7556–7566, Online. Association for Computational Linguistics.
- Yoav Artzi and Luke Zettlemoyer. 2013. Weakly supervised learning of semantic parsers for mapping instructions to actions. *Transactions of the Association for Computational Linguistics*.
- Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization. *ArXiv*, abs/1607.06450.
- Jonathan Berant and Percy Liang. 2014. Semantic parsing via paraphrasing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1415–1425, Baltimore, Maryland. Association for Computational Linguistics.
- Chandra Bhagavatula, Thanapon Noraset, and Doug Downey. 2015. Tabel: Entity linking in web tables. In *International Semantic Web Conference*.
- Ben Bogin, Matt Gardner, and Jonathan Berant. 2019. Global reasoning over database structures for text-to-sql parsing. *ArXiv*, abs/1908.11214.
- Wenhu Chen, Jianshu Chen, Yu Su, Zhiyu Chen, and William Yang Wang. 2020a. Logical natural language generation from open-domain tables. *arXiv preprint arXiv:2004.10404*.
- Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyu Zhou, and William Yang Wang. 2019. Tabfact: A large-scale dataset for table-based fact verification. *arXiv preprint arXiv:1909.02164*.
- Wenhu Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Wang. 2020b. Hybridqa: A dataset of multi-hop question answering over tabular and textual data. *arXiv preprint arXiv:2004.07347*.
- Donghyun Choi, Myeong Cheol Shin, Eunggyun Kim, and Dong Ryeol Shin. 2020. Ryansql: Recursively applying sketch-based slot fillings for complex text-to-sql in cross-domain databases. *ArXiv*, abs/2004.03125.
- Pradeep Dasigi, Matt Gardner, Shikhar Murty, Luke S. Zettlemoyer, and Eduard H. Hovy. 2019. Iterative search for weakly supervised semantic parsing. In *NAACL-HLT*.
- Xiang Deng, Huan Sun, Alyssa Lees, You Wu, and Cong Yu. 2020. Turl: Table understanding through representation learning. *arXiv preprint arXiv:2006.14806*.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.
- Li Dong and Mirella Lapata. 2018. [Coarse-to-fine decoding for neural semantic parsing](#). In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 731–742. Association for Computational Linguistics.
- Catherine Finegan-Dollak, Jonathan K. Kummerfeld, Li Zhang, Karthik Ramanathan Dhanalakshmi Ramanathan, Sesh Sadasivam, Rui Zhang, and Dragomir Radev. 2018. Improving text-to-sql evaluation methodology. In *ACL 2018*. Association for Computational Linguistics.
- 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 *ACL*.
- Tong Guo and Huilin Gao. 2019. Content enhanced bert-based text-to-sql generation. *ArXiv*, abs/1910.07179.
- Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. *ArXiv*, abs/2002.08909.
- Till Haug, Octavian-Eugen Ganea, and Paulina Grnarova. 2017. Neural multi-step reasoning for question answering on semi-structured tables. In *ECIR*.
- Pengcheng He, Yi Mao, Kaushik Chakrabarti, and Weizhu Chen. 2019. X-sql: reinforce schema representation with context. *ArXiv*, abs/1908.08113.
- Dan Hendrycks and Kevin Gimpel. 2016. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *ArXiv*, abs/1610.02136.
- Jonathan Herzig and Jonathan Berant. 2018. Decoupling structure and lexicon for zero-shot semantic parsing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1619–1629. Association for Computational Linguistics.

- Jonathan Herzig, P. Nowak, Thomas Müller, Francesco Piccinno, and Julian Martin Eisenschlos. 2020a. Tapas: Weakly supervised table parsing via pre-training. In *ACL*.
- Jonathan Herzig, Paweł Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Martin Eisenschlos. 2020b. Tapas: Weakly supervised table parsing via pre-training. *arXiv preprint arXiv:2004.02349*.
- Wonseok Hwang, Jinyeung Yim, Seunghyun Park, and Minjoon Seo. 2019. A comprehensive exploration on wikisql with table-aware word contextualization. *ArXiv*, abs/1902.01069.
- Srinivasan Iyer, Ioannis Konstas, Alvin Cheung, Jayant Krishnamurthy, and Luke Zettlemoyer. 2017. Learning a neural semantic parser from user feedback. *CoRR*, abs/1704.08760.
- Robin Jia and Percy Liang. 2016. Data recombination for neural semantic parsing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.
- Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.
- Mike Lewis, Marjan Ghazvininejad, Gargi Ghosh, Armen Agajanyan, Sida Wang, and Luke Zettlemoyer. 2020a. Pre-training via paraphrasing. *arXiv preprint arXiv:2006.15020*.
- Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020b. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.
- Fei Li and HV Jagadish. 2014. Constructing an interactive natural language interface for relational databases. *VLDB*.
- Chen Liang, Mohammad Norouzi, Jonathan Berant, Quoc V. Le, and Ni Lao. 2018. Memory augmented policy optimization for program synthesis and semantic parsing. In *NeurIPS*.
- 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 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP Findings 2020, November 16th-20th, 2020*.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke S. Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/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 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 13–23.
- Qin Lyu, Kaushik Chakrabarti, Shobhit Hathi, Souvik Kundu, Jianwen Zhang, and Zheng Chen. 2020. Hybrid ranking network for text-to-sql. Technical Report MSR-TR-2020-7, Microsoft Dynamics 365 AI.
- Sewon Min, Danqi Chen, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019. A discrete hard em approach for weakly supervised question answering. In *EMNLP*.
- Arvind Neelakantan, Quoc V. Le, Martín Abadi, Andrew McCallum, and Dario Amodei. 2016. Learning a natural language interface with neural programmer. *ArXiv*, abs/1611.08945.
- Ankur P. Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and Dipanjan Das. 2020. Totto: A controlled table-to-text generation dataset. *arXiv preprint arXiv:2004.14373*.
- Panupong Pasupat and Percy Liang. 2015. Compositional semantic parsing on semi-structured tables. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 1470–1480.
- Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543. ACL.
- Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 2227–2237. Association for Computational Linguistics.
- Matthew E. Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge enhanced contextual word representations. 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 43–54, Hong Kong, China. Association for Computational Linguistics.

- Dragomir Radev, Rui Zhang, Amrit Rau, Abhinand Sivaprasad, Chiachun Hsieh, Nazneen Fatema Rajani, Xiangru Tang, Aadit Vyas, Neha Verma, Pranav Krishna, Yangxiaokang Liu, Nadia Irwanto, Jessica Pan, Faiaz Rahman, Ahmad Zaidi, Murori Mutuma, Yasin Tarabar, Ankit Gupta, Tao Yu, Yi Chern Tan, Xi Victoria Lin, Caiming Xiong, and Richard Socher. 2020. Dart: Open-domain structured data record to text generation. *arXiv preprint arXiv:2007.02871*.
- Tianze Shi, Kedar Tatwawadi, Kaushik Chakrabarti, Yi Mao, Oleksandr Polozov, and Weizhu Chen. 2018. Incsql: Training incremental text-to-sql parsers with non-deterministic oracles. *arXiv preprint arXiv:1809.05054*.
- 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.
- Bailin Wang, Ivan Titov, and Mirella Lapata. 2019. Learning semantic parsers from denotations with latent structured alignments and abstract programs. In *Proceedings of EMNLP*.
- Yushi Wang, Jonathan Berant, and Percy Liang. 2015a. **Building a semantic parser overnight**. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1332–1342, Beijing, China. Association for Computational Linguistics.
- Yushi Wang, Jonathan Berant, Percy Liang, et al. 2015b. Building a semantic parser overnight. In *ACL (1)*, pages 1332–1342.
- Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierrick Cistac, Tim Rault, R’emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface’s transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.
- Xiaojun Xu, Chang Liu, and Dawn Song. 2017. Sqlnet: Generating structured queries from natural language without reinforcement learning. *arXiv preprint arXiv:1711.04436*.
- Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. 2020a. **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.
- Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. 2020b. Tabert: Pretraining for joint understanding of textual and tabular data. *arXiv preprint arXiv:2005.08314*.
- Tao Yu, Zifan Li, Zilin Zhang, Rui Zhang, and Dragomir Radev. 2018a. Typesql: Knowledge-based type-aware neural text-to-sql generation. In *Proceedings of NAACL*. Association for Computational Linguistics.
- Tao Yu, Michihiro Yasunaga, Kai Yang, Rui Zhang, Dongxu Wang, Zifan Li, and Dragomir Radev. 2018b. Syntaxsqlnet: Syntax tree networks for complex and cross-domain text-to-sql task. In *Proceedings of EMNLP*. Association for Computational Linguistics.
- 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. 2018c. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. In *EMNLP*.
- John M. Zelle and Raymond J. Mooney. 1996. Learning to parse database queries using inductive logic programming. In *AAAI/IAAI*, pages 1050–1055, Portland, OR. AAAI Press/MIT Press.
- Luke S. Zettlemoyer and Michael Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. *UAI*.
- Li Zhang, Shuo Zhang, and Krisztian Balog. 2019a. Table2vec: Neural word and entity embeddings for table population and retrieval. In *Proceedings of the 42Nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR’19*, pages 1029–1032, New York, NY, USA. ACM.
- Rui Zhang, Tao Yu, He Yang Er, Sungrok Shim, Eric Xue, Xi Victoria Lin, Tianze Shi, Caiming Xiong, Richard Socher, and Dragomir Radev. 2019b. 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 9th International Joint Conference on Natural Language Processing*. Association for Computational Linguistics.
- Yuchen Zhang, Panupong Pasupat, and Percy Liang. 2017. Macro grammars and holistic triggering for efficient semantic parsing. In *EMNLP*.
- Victor Zhong, Caiming Xiong, and Richard Socher. 2017. Seq2sql: Generating structured queries from natural language using reinforcement learning. *CoRR*, abs/1709.00103.

A Appendices

A.1 Additional Analysis

Training coverage As shown in Figure 2, on the challenging end text-to-SQL SPIDER task, RAT-

Task	Question	Table/Database	Annotation
SPIDER	Find the first and last names of the students who are living in the dorms that have a TV Lounge as an amenity.	database with 5 tables e.g.student, dorm_amenity, ...	SELECT T1.FNAME, T1.LNAME FROM STUDENT AS T1 JOIN LIVES_IN AS T2 ON T1.STUDID=T2.STUD WHERE T2.DORMID IN (SELECT T3.DORMID FROM HAS_AMENITY AS T3 JOIN DORM_AMENITY AS T4 ON T3.AMENID=T4.AMENID WHERE T4.AMENITY_NAME= 'TV LOUNGE')
Fully-sup. WIKISQL	How many CFL teams are from York College?	a table with 5 columns e.g. player, position, ...	SELECT COUNT CFL TEAM FROM CFLDRAFT WHERE COLLEGE = 'YORK'
WIKITABLEQUESTIONS	In what city did Piotr's last 1st place finish occur?	a table with 6 columns e.g. year, event, ...	"Bangkok, Thailand"
Weakly-sup. WIKISQL	How many CFL teams are from York College?	a table with 5 columns e.g. player, position,...	2

Table 7: Examples of the inputs and annotations for four semantic parsing tasks. SPIDER and Fully-sup. WIKISQL require full annotation of SQL programs, whereas WIKITABLEQUESTIONS and Weakly-sup. WIKISQL only requires annotation of answers (or denotations) of questions.

	Train Size	# Table	Task
TabFact	92.2K	16K	Table-based fact verification
LogicNLG	28.5K	7.3K	Table-to-text generation
HybridQA	63.2K	13K	Multi-hop question answering
WikiSQL	61.3K	24K	Text-to-SQL generation
WikiTableQuestions	17.6K	2.1K	Question answering
ToTTo	120K	83K	Table-to-text generation
Spider	8.7K	1K	Text-to-SQL generation

Table 8: Aggregated datasets for table-and-language tasks.

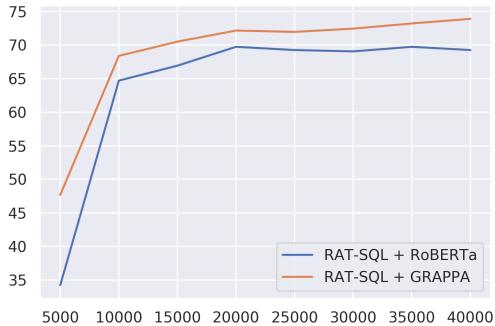
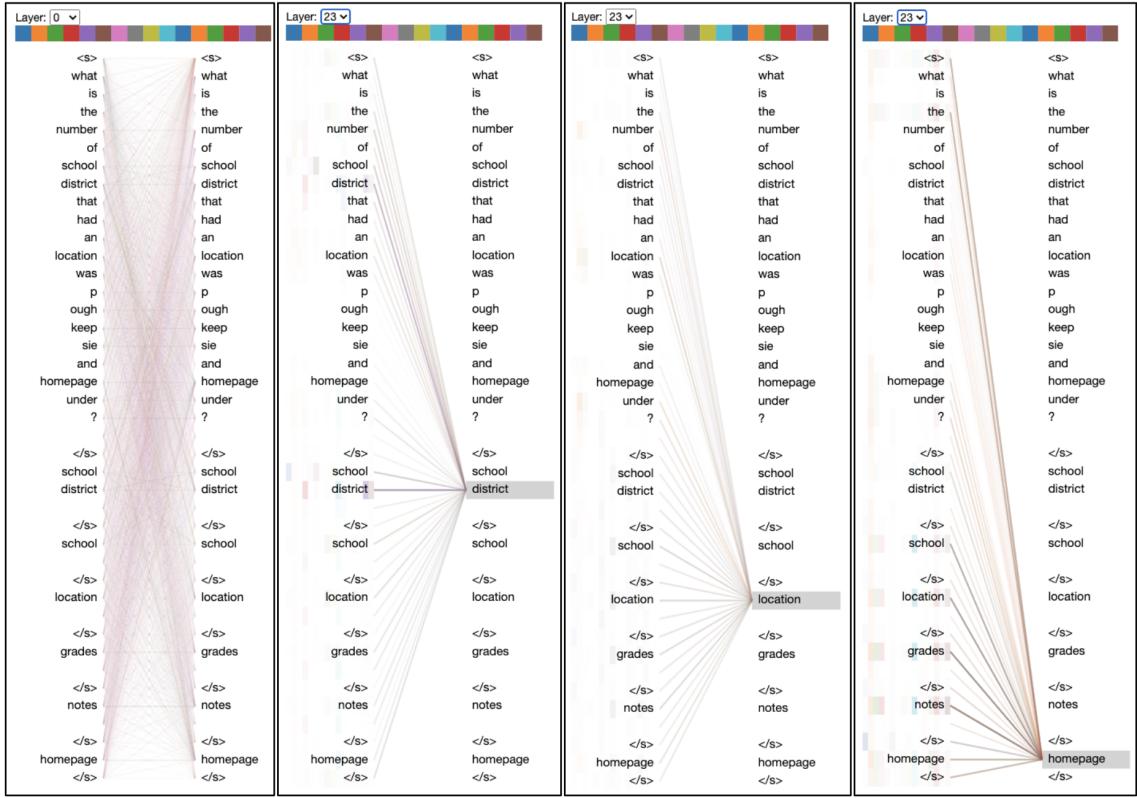
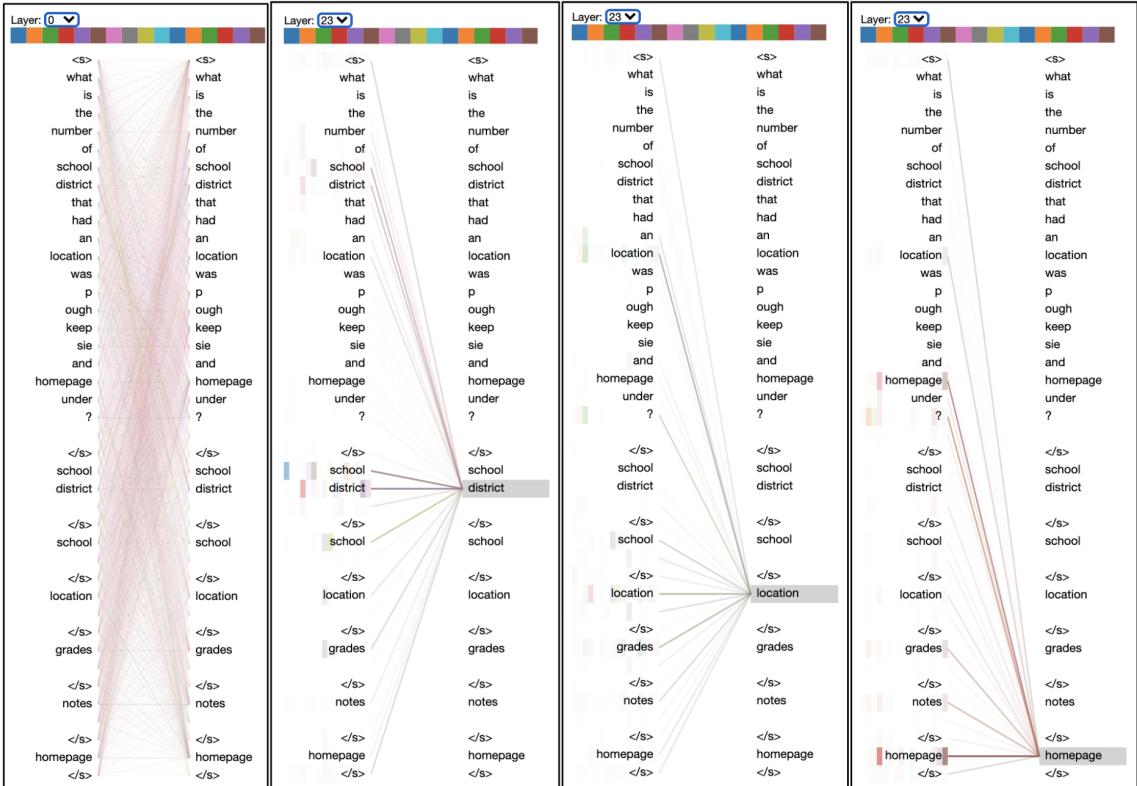


Figure 2: The development exact set match score in SPIDER vs. the number of training steps. RAT-SQL initialized with our pre-trained GRAPPA converges to higher scores in a shorter time than RAT-SQL w. BERT.

SQL initialized with GRAPPA outperforms RAT-SQL using RoBERTa by about 14% in the early training stage. This shows that GRAPPA already captures some semantic knowledge in pre-training. Finally, GRAPPA is able to keep the competitive edge by 4%.

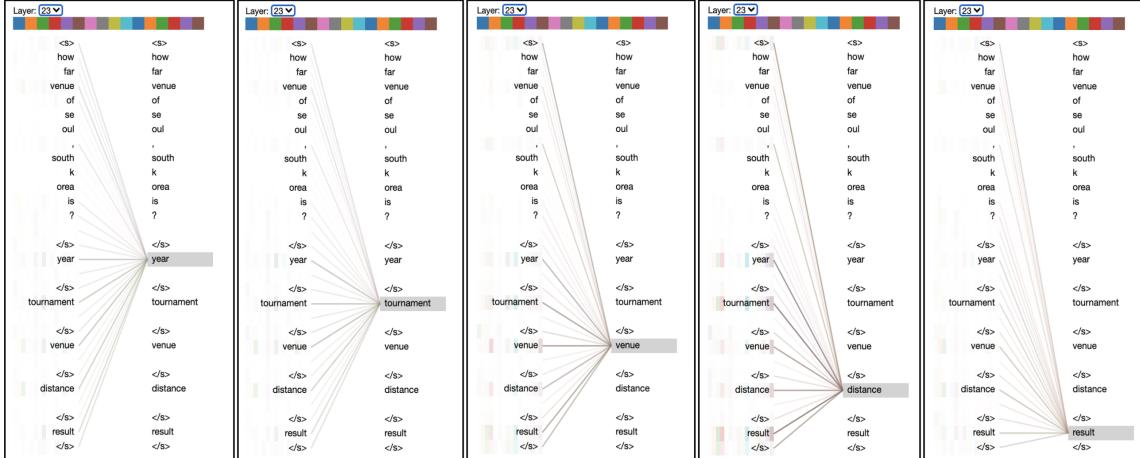


(a) RoBERTa-large

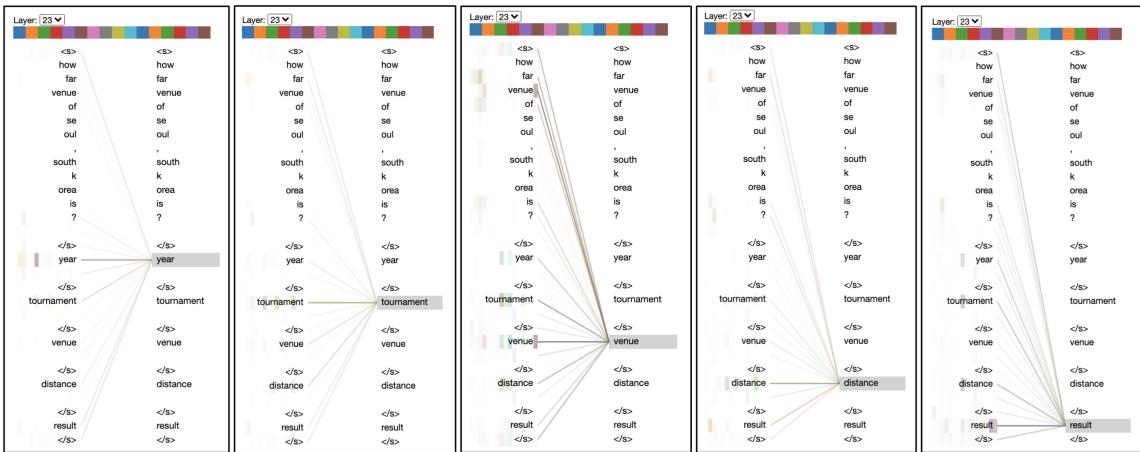


(b) GRAPPA

Figure 3: Attention visualization on the last self-attention layer.



(a) RoBERTa-large



(b) GRAPPA

Figure 4: Attention visualization on the last self-attention layer.