SQL Generation via Machine Reading Comprehension

Zeyu Yan[*], **Jianqiang Ma**[*], **Yang Zhang, Jianping Shen**
{yanzeyu751,majianqiang554,zhangyang147,shenjianping324}@pingan.com.cn
AI Team, Ping An Life Insurance Company of China, Ltd

Abstract

Text-to-SQL systems offers natural language interfaces to databases, which can automatically generates SQL queries given natural language questions. On the WikiSQL benchmark, state-of-the-art text-to-SQL systems typically take a slot-filling approach by building several specialized models for each type of slot. Despite being effective, such modularized systems are complex and also fall short in jointly learning for different slots. To solve these problems, this paper proposes a novel approach that formulates the task as a question answering problem, where different slots are predicted by a unified machine reading comprehension (MRC) model. For this purpose, we use a BERT-based MRC model, which can also benefit from intermediate training on other MRC datasets. The proposed method can achieve competitive results on WikiSQL, suggesting it being a promising direction for text-to-SQL.

1 Introduction

Text-to-SQL systems generate SQL queries according to given natural language (NL) queries, as shown in example (1), where the headers in the table schema are {PLAYER, COUNTRY, YEAR(S) WON, TOTAL, TO PAR, FINISH}. Text-to-SQL technology is very useful as it can empower humans to naturally interact with relational databases, which serve as foundations for the digital world today. As a subarea of semantic parsing (Berant et al., 2013), text-to-SQL is known to be difficult due to the flexibility in natural language.

(1) a. **NL Query**: Who is the player from the United States with a total less than 293?
b. **SQL Query**: `SELECT` Player `FROM` T `WHERE` Country = 'United States' `AND` Total $<$ 293

Recently, by the development of deep learning, significant advances have been made in text-to-SQL. On the WikiSQL (Zhong et al., 2018) benchmark for multi-domain, single table text-to-SQL task, state-of-the-art systems (Hwang et al., 2019; He et al., 2019) can predict more than 80% of entire SQL queries correctly. Most of such systems take a slot-filling approach (Xu et al., 2018) that builds several (e.g. 6) specialized models, each of which is dedicated to predicting a particular type of slots, such as the column in `SELECT`, or the filter value in `WHERE`. For practical applications, however, such methods have two drawbacks: First, it is complex and delicate in architecture to rely on many dedicated modules working together to generate SQLs, which poses challenges in (joint) training, deployment and maintenance. Second, since most slot types are modeled with no or only limited dependencies on other slots, it is difficult for such models to leverage inter-dependencies of SQL slots. To deal with such problems, this paper formulates text-to-SQL as a question answering task (Section 3). In this formulation, we use a unified BERT-based (Devlin et al., 2019) machine reading comprehension (MRC) model to predict each type of SQL slots by answering template-generated questions. Then the SQL query is synthesized in the way as in slot-filling approaches. For instance, the SQL query for the example (1) can be re-constructed by answering questions, some of which are shown in Table 1.

[*]Equal contributions.

Proceedings of the 28th International Conference on Computational Linguistics, pages 350–356
Barcelona, Spain (Online), December 8-13, 2020

Context: *Query*: Who is the player from the United States with a total less than 293? *Header*: Player, Country, Year(s) won, Total, To par, Finish, *AGG*: empty maximum minimum count sum average.	
Question	**Answer Set**
What is the select column?	{Player}
What is the aggregation function?	{empty}
What are the values?	{United States, 293}
What is the filter column for "United States"?	{Country}
What is the filter column for "293"?	{Total}

Table 1: **Context, questions and answers in the MRC formulation for the example (1)**.

In the QA formulation, as all slot types are predicted by the same MRC model in the same manner, we arrive at a much simpler architecture with the benefits of easier training, deployment and maintenance. Moreover, with well-designed question generation strategy, important prior information for slot predictions can be added into the questions to leverage the power of BERT even more. Besides, our MRC-based model can naturally benefit from supplementary training on intermediate-labeled tasks (STILTs) (Phang et al., 2018).

The main contribution of this paper is an MRC approach to text-to-SQL. To the best of our knowledge, this is the *first* work that casts sketch-based text-to-SQL into question answering. We show that the proposed method can achieve competitive results on the WikiSQL dataset.

2 Related Work

Text-to-SQL is a sub-area of semantic parsing (Berant et al., 2013), which maps natural language utterances to machine-interpretable representations, such as logic forms (Dong and Lapata, 2016), program codes (Yin and Neubig, 2017), and SQLs. For single-table, simple query text-to-SQL task of WikiSQL, many earlier work (Dong and Lapata, 2016; Krishnamurthy et al., 2017; Sun et al., 2018; Wang et al., 2018) follow a neural sequence-to-sequence architecture (Sutskever et al., 2014) with attention mechanism (Bahdanau et al., 2014). This approach often suffers the "ordering issue" when the WHERE-clause has more than one conditions. Xu et al. (2018) introduces a sketch based method, which treats text-to-SQL as slot-filling, by decomposing the SQL synthesis into several independent classification sub-tasks. Specifically, the aggregation function, the column in SELECT-clause, number of conditions, and each element in <column, operator, cell value> triplets in WHERE-clause are predicted separately. Recent advances (Yu et al., 2018a; Dong and Lapata, 2018; Hwang et al., 2019; He et al., 2019) mostly follow this approach and achieve competitive results on WikiSQL. However, sketch-based models, most of which are based on SQLNet (Xu et al., 2018), usually consist of six or more sub-modules and thus complex. By contrast, our question answering-based approach uses a unified MRC model to make predictions for all the SQL slots, thus enjoys a much *simpler architecture* and provides a natural way to *jointly modeling* different slots types.

Many recent work (Krishnamurthy et al., 2017; Guo et al., 2019; Wang et al., 2020; Choi et al., 2020) focused on multi-table and complex queries setting of text-to-SQL, as in the Spider task (Yu et al., 2018b). State-of-the art methods on Spider typically fall into two categories: *grammar-based approach* (Guo et al., 2019; Wang et al., 2020), and *sketch-based approach*, such as RYANSQL (Choi et al., 2020).Sketch-based methods also have slot prediction modules similar to SQLNet for the WikiSQL, while recursion modules are developed to handle the generation of complex SQL sketches, a characteristic in Spider but absent in WikiSQL. At a high level, our method is along the same line of SQLNet-RYANSQL, yet differs with them, as our method recognize slots in a unified way rather than using dedicated modules to predict each slot type. We can extend our method to the Spider task by following existing sketch construction methods as in RYANSQL, while replacing their slot classification modules with our MRC-based methods.

Machine reading comprehension MRC models (Seo et al., 2017; Wang et al., 2016; Xiong et al., 2018) are typically trained to answer questions by extracting a text span from the given context passage.

Thus it is often reduced to predicting the start and end position of the answer in the context passage. Recently, there is a trend to cast non-QA NLP tasks, such as information extraction (Levy et al., 2017; Li et al., 2019; Li et al., 2020), text classification and more (McCann et al., 2018; Keskar et al., 2019) into MRC, which can achieve comparable or improved results on the original task, thanks to the flexible and unified modeling of MRC formulation. Our work is inspired by these previous work, but tackles a new and sophisticated scenario of semantic parsing.

3 Method

Task formulation and dataset conversion Given a *question* $Q = q_1 q_2 .. q_L$, and a *context* passage $C = c_1 c_2 ... c_M$, were $|Q| = L$ and $|C| = M$ are their token numbers. The task is to find the start token C_{start} and end token C_{end} in the context passage for the given question. Some example questions and their context are shown in Table 1. To fit such task formulation and apply MRC models, we first convert standard text-to-SQL annotations, in the form of NL query-SQL query pairs as shown in example (1), together with table headers, into a set of <*question*, *answer*, *context*> triples, similar to the SQuAD dataset (Rajpurkar et al., 2016). For each type of slot $y \in Y$ that we would like to predict, we use template to generate a *question* Q. For all the generated questions associated with the same SQL, we provide a *context* C, which consists of the original NL query, the table headers and the textual description of the aggregation functions. The context is constructed in such a way that the answer for Q, i.e. the SQL slot to be predicted, is represented by a textual *span* in the context, we can denote as C_{start}-C_{end}. As in the standard MRC setting, all the predictions are reduced to predicting the start and end index of the textual span in the context passage.

Since there are multiple filters in one SQL in WikiSQL dataset, the MRC method transforms into a multi-turn procedure. In each turn of this procedure, one type of slot will be asked according to results from previous turns. For example in Table 1, the question about select column will be answered and then the aggregation function without needing previous results. A multi-span extraction is conducted to answer the question about values in the data. Once the values are extracted by the MRC model, the filter column corresponding to each value is asked using the extracted value. For example, the question *What is the filter column for "United States"?* is asked only after the value "United States" is extracted in the previous step. Note that the questions are fed into MRC model follows the above order only in the prediction phase, where all predicted elements of SQL query are constructed one by one in SQL query format of WikiSQL. In the training phase, however, triples of different samples are shuffled into batches to feed into the MRC model while the questions about values are constructed with standard results in the dataset.

3.1 BERT-based MRC model

Given the question Q, we need to extract the text span from context C using an MRC model. For this purpose, we use BERT (Devlin et al., 2019). In particular, we resort to the standard question-answer usage of BERT, i.e. feeding the token sequence in the form of $\texttt{[CLS]}, q_1, q_2, .., q_L, \texttt{[SEP]}, c_1, c_2, ..., c_M$ as the input to the BERT model, where the special SEP token is inserted between the question Q and the context C to distinguish them. BERT then outputs a contextualized representation matrix $H \in R^{(L+M+2)\times d}$, where d is the vector dimension of the last layer of BERT. Following the MRC setup as in (Devlin et al., 2019), we use two additional trainable parameter vectors v_{start} and v_{end}, both of which are of dim d, to compute the probability of each token position being the start and end position of the answer span, respectively. The computation is simply by applying the softmax function over the multiplication of the BERT representation for each token H_i with the two vectors v_{start} and v_{end}, as shown in (1).

$$\begin{aligned} H^Q; H^C &= \text{BERT}([Q; C]) \\ p_{start}(i) &= softmax(H_i^C v_{start}) \\ p_{end}(i) &= softmax(H_i^C v_{end}) \end{aligned} \tag{1}$$

The MRC model described above fits scenarios of extraction only one pair of start position vector p_{start} and end position vector p_{end} from the context for the given question, such as the prediction of only

one SELECT column and aggregation function in the WikiSQL format, as well as the filter column prediction when the value is given. However, such model is not suitable for value predictions, as there can be multiple filter conditions, each of which has a corresponding value slot, i.e. multiple value spans in the query to be extracted simultaneously as answers. To overcome this limitation, we adopt sequence labeling to predict values using BIO tag-set. Specifically, The BERT representation for each token in the *context* part. i.e. H^C, is fed to a conditional random field (CRF) (Lafferty et al., 2001) layer to yield the output labels, which is shown in (2).

$$T = \text{CRF}(W^L H^C), |T| = |C| \tag{2}$$

Output T has the same length of the context C while each token in C is assigned a BIO label in T to show if it is a beginning(B) token of a value , or a continuation(I) token of a value, or even outside(O) of a value of value prediction. The results of value predictions can be extracted from such label sequence by combining one B-label and following I-labels with ignoring O-labels. Then the predictions of other SQL elements follow the MRC framework. Such treatment is similar to MRC-based entity-relation extraction work (Li et al., 2019), As future work, techniques in Hu et al. (2019) will be experimented to further unify value predictions into the MRC framework.

STILTs and AGG prediction enhancement STILTs (Phang et al., 2018) refers to the procedure that first fine-tunes a pre-trained language model on an intermediate task, before fine-tuning on the final task. The procedure is known to be effective on improving MRC models by intermediate fine-tuning on other QA datasets (Keskar et al., 2019). Thus we take advantage of STILTs to boost the performance of our BERT-based MRC model. An additional improvement focuses on the aggregation function (AGG) prediction. Analysis of preliminary results suggests that AGG prediction is a bottleneck for our system, which is partly attributed to the findings by Hwang et al. (2019) that AGG annotations in WikiSQL have up to 10% of errors. Since our unified MRC model has to take care of other types of questions, these extra constraints make it more challenging for our model to fit flawed data, compared with a dedicated AGG classifier, as in most SOTA methods. In such case, we improve the AGG results over the original MRC predictions, using only simple association signals in the training data. To this end, we adopt transformation-based learning algorithm (Brill, 1995) to update the AGG predictions based on simple association rules in the form of "change AGG from x to x', given certain word tuple occurrences." Such rules are mined and ranked from the training data by the algorithm.

4 Experiment

4.1 Dataset, Metric and Implementation Details

We use the largest human-annotated text-to-SQL dataset, WikiSQL (Zhong et al., 2018), which consists of 80,654 pairs of questions and human-verified SQL queries. Tables appeared either in train or dev set will never appear in the testset. As in previous work, the following two metrics are used for evaluating SQL query synthesis accuracy: (1) *Logical Form Accuracy*, denoted as LF, where LF = SQL with correct logic form / total # of SQL; and (2) *Execution Accuracy*, denoted as EX. where EX = SQL with correct execution / total # of SQL. Execution guidance decoding (Wang et al., 2018) is not evaluated.

The word embeddings are randomly initialized by BERT, and fine-tuned during the training. Adam is used (Kingma and Ba, 2014) to optimize the model with default hyper-parameters. We choose uncased BERT-base pre-trained model with default settings due to resource limitations. Codes are implemented in Pytorch 1.3 and will be made publicly available[1].

4.2 Results

We compare our method with notable published work that has reported results on WikiSQL, including Seq2SQL (Zhong et al., 2018), SQLNet (Xu et al., 2018), TypeSQL (Yu et al., 2018a), Coarse-to-Fine (Dong and Lapata, 2018), SQLova (Hwang et al., 2019), X-SQL (He et al., 2019) in Table 2. On the test set, our final model with BERT-base outperforms SQLova, the BERT-large based strong baseline,

[1] https://github.com/nl2sql/QA-SQL

and rivals the SOTA X-SQL with MT-DNN. For STILTs, we fine-tuned BERT on SQuAD 1.1 dataset for 3 epochs with hyper-parameters similar to Devlin et al. (2019), before fine-tuning on WikiSQL. As shown in Table 2 and 3, STILTS training can benefit our MRC-based model, especially the LF accuracy. We expect further improvement from chained STILTs as in Keskar et al. (2019) with more QA datasets.

Model	Dev		Test	
	LF	EX	LF	EX
Seq2SQL	49.5	60.8	48.3	59.4
SQLNet	63.2	69.8	61.3	68.0
TypeSQL	68.0	74.5	66.7	73.5
Coarse2Fine	72.5	79.0	71.7	78.5
SQLova	81.6	87.2	80.7	86.2
X-SQL	**83.8**	**89.5**	**83.3**	**88.7**
ours	79.4	85.9	79.3	85.9
ours + ST	80.2	86.2	79.5	86.0
ours + ST + AE	81.9	87.8	81.8	87.4

Table 2: **Accuracy of previous and this work**. ST: STILTs training, AE: AGG enhancement.

Analysis Table 3 shows slot type-wise results, implying aggregation function accuracy S_{agg} is the bottleneck to the pure MRC model (*ours*), which is probably due to that our unified model meets more obstacles in fitting partially erroneous data, for which the AGG enhancement method (*AE*) is very effective. Our error analysis on 100 randomly sampled errors shows that while 47% of the errors can be attributed to the model, 53% can be best described as data flaws or errors, the majority of which involves AGG. For example, "*What year* has a bronze of Valentin Novikov", "*What year* has a silver for Matthias Merz", and "*What is the year* of the Film Klondike Annie" are of same (WH-word + SELCT column) patterns, but gold AGGs are `AVG`, `MIN` and `SUM`, respectively. We further make an automatic analysis, finding that 8.91% of the data are cases where queries of the same pattern are annotated with at least 3 distinct AGG. Such inconsistency suggests that even higher accuracy means fitting both signal and noise.

Model	S_{col}	S_{agg}	$W_{no.}$	W_{col}	W_{op}	W_{val}
SQLova	96.8	90.6	98.5	94.3	97.3	95.4
X-SQL	**97.2**	91.1	**98.6**	95.4	97.6	96.6
ours	96.7	90.0	98.3	95.4	98.7	96.8
ours+ST	96.8	90.7	98.3	95.4	98.7	96.8
ours+ST+AE	96.8	**92.8**	98.3	**95.4**	**98.7**	**96.8**

Table 3: **Test accuracy for each slot type.** ST: STILTs training, AE: AGG prediction enhancement.

5 Conclusion

This paper proposes a question answering approach to text-to-SQL, where a BERT-based MRC model is trained to predict all the slots that are needed for SQL generation. Our approach enjoys advantages of easier deployment and maintenance in practice, as well as the potentials in leveraging other MRC datasets via the STILTs supplementary training. Capable of jointly learning slots with a simple, unified model, the proposed method proves to be a promising direction for text-to-SQL. As future work, We plan to extend our model to cope with multi-table text-to-SQL task, Spider.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544, Seattle, Washington, USA, October. Association for Computational Linguistics.

Eric Brill. 1995. Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. *Computational linguistics*, 21(4):543–565.

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 preprint arXiv:2004.03125*.

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

Li Dong and Mirella Lapata. 2016. Language to logical form with neural attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 33–43, Berlin, Germany, August. Association for Computational Linguistics.

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, Melbourne, Australia, July. 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 *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4524–4535, Florence, Italy, July. Association for Computational Linguistics.

Pengcheng He, Yi Mao, Kaushik Chakrabarti, and Weizhu Chen. 2019. X-sql: reinforce schema representation with context. *arXiv preprint arXiv:1908.08113*.

Minghao Hu, Yuxing Peng, Zhen Huang, and Dongsheng Li. 2019. A multi-type multi-span network for reading comprehension that requires discrete reasoning. 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 1596–1606, Hong Kong, China, November. Association for Computational Linguistics.

Wonseok Hwang, Jinyeung Yim, Seunghyun Park, and Minjoon Seo. 2019. A Comprehensive Exploration on WikiSQL with Table-Aware Word Contextualization. *ArXiv*, abs/1902.01069.

Nitish Shirish Keskar, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Unifying question answering, text classification, and regression via span extraction. *arXiv: Computation and Language*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Jayant Krishnamurthy, Pradeep Dasigi, and Matt Gardner. 2017. Neural semantic parsing with type constraints for semi-structured tables. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1516–1526, Copenhagen, Denmark, September. Association for Computational Linguistics.

John D. Lafferty, Andrew McCallum, and Fernando C Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML*.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-shot relation extraction via reading comprehension. pages 333–342, August.

Xiaoya Li, Fan Yin, Zijun Sun, Xiayu Li, Arianna Yuan, Duo Chai, Mingxin Zhou, and Jiwei Li. 2019. Entity-relation extraction as multi-turn question answering. In *Proceedings of ACL*, pages 1340–1350, Florence, Italy, July. Association for Computational Linguistics.

Xiaoya Li, Jingrong Feng, Yuxian Meng, Qinghong Han, Fei Wu, and Jiwei Li. 2020. A unified MRC framework for named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online, July. Association for Computational Linguistics.

B. McCann, N. Keskar, Caiming Xiong, and R. Socher. 2018. The natural language decathlon: Multitask learning as question answering. *ArXiv*, abs/1806.08730.

Jason Phang, Thibault Févry, and Samuel R. Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *ArXiv*, abs/1811.01088.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas, November. Association for Computational Linguistics.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. *ArXiv*, abs/1611.01603.

Yibo Sun, Duyu Tang, Nan Duan, Jianshu Ji, Guihong Cao, Xiaocheng Feng, Bing Qin, Ting Liu, and Ming Zhou. 2018. Semantic parsing with syntax- and table-aware SQL generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Melbourne, Australia, July. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *NIPS*.

Zhiguo Wang, Haitao Mi, Wael Hamza, and Radu Florian. 2016. Multi-perspective context matching for machine comprehension. *ArXiv*, abs/1612.04211.

Chenglong Wang, Po-Sen Huang, Alex Polozov, Marc Brockschmidt, and Rishabh Singh. 2018. Execution-guided neural program decoding. *ArXiv*, abs/1807.03100.

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, July. Association for Computational Linguistics.

Caiming Xiong, Victor Zhong, and Richard Socher. 2018. Dcn+: Mixed objective and deep residual coattention for question answering. *ArXiv*, abs/1711.00106.

Xiaojun Xu, Chang Liu, and Dawn Xiaodong Song. 2018. SQLNet: Generating structured queries from natural language without reinforcement learning. *ArXiv*, abs/1711.04436.

Pengcheng Yin and Graham Neubig. 2017. A syntactic neural model for general-purpose code generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 440–450, Vancouver, Canada, July. Association for Computational Linguistics.

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 the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, New Orleans, Louisiana, June. 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. 2018b. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3911–3921, Brussels, Belgium, October-November. Association for Computational Linguistics.

Victor Zhong, Caiming Xiong, and Richard Socher. 2018. Seq2sql: Generating structured queries from natural language using reinforcement learning. *ArXiv*, abs/1709.00103.