A Globally Normalized Neural Model for Semantic Parsing

Chenyang Huang[1], **Wei Yang**[2], **Yanshuai Cao**[2], **Osmar Zaïane**[1], **Lili Mou**[1]
[1]Department of Computing Science,
Alberta Machine Intelligence Institute (Amii), University of Alberta
[2]Borealis AI
{chenyangh,zaiane}@ualberta.ca
{wei.yang,yanshuai.cao}@borealisai.com
doublepower.mou@gmail.com

Abstract

In this paper, we propose a globally normalized model for context-free grammar (CFG)-based semantic parsing. Instead of predicting a probability, our model predicts a real-valued score at each step and does not suffer from the label bias problem. Experiments show that our approach outperforms locally normalized models on small datasets, but it does not yield improvement on a large dataset.

1 Introduction

Semantic parsing has received much interest in the NLP community (Zelle and Mooney, 1996; Zettlemoyer and Collins, 2005; Jia and Liang, 2016; Guo et al., 2020). The task is to map a natural language utterance to executable code, such as λ-expressions, SQL queries, and Python programs.

Recent work integrates the context-free grammar (CFG) of the target code into the generation process. Instead of generating tokens of the code (Dong and Lapata, 2016), CFG-based semantic parsing predicts the grammar rules in the abstract syntax tree (AST). This guarantees the generated code complies with the CFG, and thus it has been widely adopted (Yin and Neubig, 2018; Guo et al., 2019; Bogin et al., 2019; Sun et al., 2019, 2020).

Typically, the neural semantic parsing models are trained by maximum likelihood estimation (MLE). The models predict the probability of the next rules in an autoregressive fashion, known as a locally normalized model.

However, local normalization is often criticized for the label bias problem (Lafferty et al., 2001; Andor et al., 2016; Wiseman and Rush, 2016; Stanojević and Steedman, 2020). In semantic parsing, for example, grammar rules that generate identifiers (e.g., variable names) have much lower probability than other grammar rules. Thus, the model will be biased towards such rules that can avoid predicting identifiers. More generally, the locally normalized model will prefer such early-step predictions that can lead to low entropy in future steps.

In this work, we propose to apply global normalization to neural semantic parsing. Our model scores every grammar rule with an unbounded real value, instead of a probability, so that the model does not have to avoid high-entropy predictions and does not suffer from label bias. Specifically, we use max-margin loss for training, where the ground truth is treated as the positive sample and beam search results are negative samples. In addition, we accelerate training by initializing the globally normalized model with the parameters from a pretrained locally normalized model.

We conduct experiments on three datasets: ATIS (Dahl et al., 1994), CoNaLa (Yin et al., 2018), and Spider (Yu et al., 2018). Compared with local normalization, our globally normalized model is able to achieve higher performance on the small ATIS and CoNaLa datasets with the long short-term memory (LSTM) architecture, but does not yield improvement on the massive Spider dataset when using a BERT-based pretrained language model.

2 Related Work

Early approaches to semantic parsing mainly rely on predefined templates, and are domain-specific (Zelle and Mooney, 1996; Zettlemoyer and Collins, 2005; Kwiatkowksi et al., 2010). Later, researchers apply sequence-to-sequence models to semantic parsing. Dong and Lapata (2016) propose to generate tokens along the syntax tree of a program. Yin and Neubig (2017) generate a program by predicting the grammar rules; our work uses the TranX tool (Yin and Neubig, 2018) with this framework.

Globally normalized models, such as the conditional random field (CRF, Lafferty et al., 2001), are able to mitigate the label bias problem. How-

Proceedings of the 5th Workshop on Structured Prediction for NLP, pages 61–66

ever, their training is generally difficult due to the global normalization process. To tackle this challenge, Daumé and Marcu (2005) propose learning as search optimization (LaSO), and Wiseman and Rush (2016) extend it to the neural network regime as beam search optimization (BSO). Specifically, they obtain negative partial samples whenever the ground truth falls out of the beam during the search, and "restart" the beam search with the ground truth partial sequence teacher-forced.

Our work is similar to BSO. However, we search for an entire output, and do not train with partial negative samples. This is because our decoder is tree-structured, and different partial trees cannot be implemented in batch efficiently. We instead perform locally normalized pretraining to ease the training of our globally normalized model.

3 Methodology

In this section, we first introduce the neural semantic parser TranX, which servers as the locally normalized base model in our work. We then elaborate how to construct its globally normalized version.

3.1 The TranX Framework

TranX is a context-free grammar (CFG)-based neural semantic parsing system (Yin and Neubig, 2018). TranX first encodes a natural language input X with a neural network encoder.

Then, the model generates a program by predicting the grammar rules (also known as actions) along the abstract syntax tree (AST) of the program. In Figure 1, for example, the rules generating the desired program include ApplyConstr(Expr.), ApplyConstr(Call), ApplyConstr(Attr.), and GenToken(sorted).

In TranX, these actions are predicted in an autoregressive way based on the input X and the partially generated tree, given by

$$P_L(a_t|\boldsymbol{a}_{<t}, X; \boldsymbol{\theta}_L) = \frac{\exp\{o(a_t|\boldsymbol{a}_{<t}, X; \boldsymbol{\theta}_L)\}}{\sum_{a'_t \in \mathcal{A}_t(\boldsymbol{a}_{<t})} \exp\{o(a'_t|\boldsymbol{a}_{<t}, X; \boldsymbol{\theta}_L)\}} \quad (1)$$

where $\boldsymbol{\theta}_L$ denotes the parameters of the neural network model, and the subscript L emphasizes that the probability is locally normalized. $o(\cdot)$ denotes the logit at this step, and a_t is an action (i.e., grammar rule) among all possible actions at this step $\mathcal{A}_t(\cdot)$, which is based on previous predicted rules $\boldsymbol{a}_{<t}$.

In other words, the prediction probability is normalized at every step, and the training objective is to maximize

$$P_L(\boldsymbol{a}_{1:n}|X; \boldsymbol{\theta}_L) = \prod_{t=1}^{n} P_L(a_t|\boldsymbol{a}_{<t}, X; \boldsymbol{\theta}_L) \quad (2)$$

where n is the total number of steps.

3.2 Globally Normalized Training

A locally normalized model may suffer from the label bias problem (Lafferty et al., 2001). This is because such a model normalizes the probability to 1 at every step. However, the candidate action set $\mathcal{A}_t(\boldsymbol{a}_{<t})$ may have different sizes, and the actions from a smaller $\mathcal{A}_t(\boldsymbol{a}_{<t})$ typically have higher probabilities. Thus, the model would prefer such actions $\boldsymbol{a}_{<t}$ that will yield smaller $\mathcal{A}_t(\boldsymbol{a}_{<t})$ in future steps.[1]

We propose to adapt TranX to a global normalized model to alleviate label bias. Instead of predicting a probability $P(a_t|\boldsymbol{a}_{<t}, X)$ as in (2), our globally normalized model predicts a positive score at a step as

$$s(a_t|\boldsymbol{a}_{<t}, X; \boldsymbol{\theta}_G) = \exp\{o(a_t|\boldsymbol{a}_{<t}, X; \boldsymbol{\theta}_G)\} \quad (3)$$

where $o(\cdot)$ is the same logit as (1), and $\boldsymbol{\theta}_G$ is the parameters.

The probability of the sequence $\boldsymbol{a}_{1:n}$ is normalized only once in a global manner, given by

$$P_G(\boldsymbol{a}_{1:n}, X; \boldsymbol{\theta}_G) = \frac{1}{Z_G} \prod_{t=1}^{n} s(a_t|\boldsymbol{a}_{<t}, X; \boldsymbol{\theta}_G) \quad (4)$$

where $Z_G = \sum_{\boldsymbol{a}'_{1:n}} \prod_{t=1}^{n} s(a'_t|\boldsymbol{a}'_{<t}; \boldsymbol{\theta}_G)$ is the partition function.

A globally normalized model alleviates the label bias problem, because it does not normalize the probability at every prediction step, as seen from (4). Thus, it is not biased by the size of $\mathcal{A}_t(\boldsymbol{a}_{<t})$.

The training objective is still to maximize the likelihood, albeit normalized in a global way. However, computing the partition function Z_G requires enumerating all combinations of actions $\boldsymbol{a}'_{1:n}$ in the partition function of (4), which is generally intractable.

In practice, the maximum likelihood training is approximated by max-margin loss between a positive sample $\boldsymbol{a}_{1:n}$ and a negative sample $\boldsymbol{a}^-_{1:n}$,

[1] Or more generally, the model prefers $\mathcal{A}_t(\boldsymbol{a}_{<t})$ with a smaller entropy.

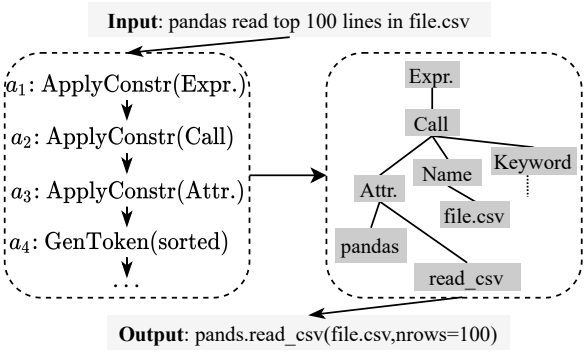

Figure 1: An example of generating a Python program with TranX.

given by

$$\mathcal{L}(\boldsymbol{a}^-_{1:n}, \boldsymbol{a}_{1:n}) = \max\{0, o(\boldsymbol{a}^-_{1:n}|X) - o(\boldsymbol{a}_{1:n}|X) + \Delta\} \tag{5}$$

where $o(\boldsymbol{a}_{1:n}|X) = \frac{1}{n}\sum_{t=1}^{n} o(a_t|\boldsymbol{a}_{<t})$ is the average of logits. Δ is a positive constant.

The positive sample is simply the ground truth actions, whereas the negative samples are obtained by beam search. In other words, we perform beam search inference during training, and the sequences in the beam (other than the ground truth) serve as the negative samples.

Similar to MLE training for (4), the max-margin loss increases the logits of the ground truth sample, while decreasing the logits for others. It is noted that the quality of negative samples will largely affect the max-margin training, as only a few samples are used to approximate Z_G.

To address this issue, we initialize the parameters of the globally normalized model $\boldsymbol{\theta}_G$ with $\boldsymbol{\theta}_L$ in a pretrained locally normalized model. Thus, our negative samples are of higher quality, so that the max-margin training is easier and more stable.

3.3 Handling the Copy Mechanism

TranX has a copy mechanism (Gu et al., 2016) as an important component for predicting the terminal nodes of the AST, as the target program largely overlaps with the source utterance, especially for entities (e.g., "file.csv" in Figure 1). In the locally normalized TranX, the copy mechanism marginalizes the probability of generating a token in the vocabulary and copying it from the source:

$$\begin{aligned} &P_L(a_t = \text{GenToken}[v] \,|\, \boldsymbol{a}_{<t}, X) \\ &= P(\text{gen} \,|\, \boldsymbol{a}_{<t}, X) P(v \,|\, \text{gen}, \boldsymbol{a}_{<t}, X) \\ &\quad + P(\text{copy} \,|\, \boldsymbol{a}_{<t}, X) P(v \,|\, \text{copy}, \boldsymbol{a}_{<t}, X) \end{aligned}$$

where $\text{GenToken}[\cdot]$ denotes generating a terminal token v. $P(\text{copy}|\cdot)$ is the predicted probability of copying the token v from the source utterance, and $P(\text{gen}|\cdot) = 1 - P(\text{copy}|\cdot)$ is the probability of generating v from the vocabulary.

However, the copy mechanism cannot be directly combined with global normalization, because we use unbounded, real-valued logits instead of probabilities. This would not make much sense when both logits are negative, whereas their product is positive.

Therefore, we propose a variant of copy mechanisms in the globally normalized setting. Specifically, we keep the probabilities $P(\text{copy}|\cdot)$ and $P(\text{gen}|\cdot)$, and use them to weight the logits of generating and copying a token v, given by

$$\begin{aligned} &o(a_t = \text{GenToken}[v] \,|\, \boldsymbol{a}_{<t}, X) \\ &= P(\text{gen} \,|\, \boldsymbol{a}_{<t}, X) o(v \,|\, \text{gen}, \boldsymbol{a}_{<t}, X) \\ &\quad + P(\text{copy} \,|\, \boldsymbol{a}_{<t}, X) o(v \,|\, \text{copy}, \boldsymbol{a}_{<t}, X) \end{aligned}$$

Here, $o(a_t = \text{GenToken}[v] \,|\, \cdot)$ is a linear interpolation of two logits, and thus fits the max-margin loss (5) naturally.

4 Experiments

Datasets. We conduct experiments on three benchmark parsing datasets: ATIS (Zettlemoyer and Collins, 2007), CoNaLa (Yin et al., 2018), and Spider (Yu et al., 2018), which contain 4473, 2379, and 8695 training samples, respectively.

It should be pointed out that much work adopts data anonymization techniques to replace entities with placeholders (Dong and Lapata, 2016; Yin and Neubig, 2017, 2019; Sun et al., 2020). This unfortunately causes a large number of duplicate samples between training and test. This is recently realized in Guo et al. (2020), and thus, in our work, we only compare the models using the original, correct ATIS dataset.

Settings. Our globally normalized semantic parser is developed upon the open-sourced TranX[2]. We adopt the CFG grammars provided by TranX to convert lambda calculus and Python programs into ASTs and sequence of grammar rules (actions). For ATIS and CoNaLa datasets, we use long LSTM models as both the encoder and the decoder. Their dimensions are set to 256. For the Spider dataset, we use a pretrained BERT model[3] (Devlin et al.,

[2] https://github.com/pcyin/tranX

[3] Specifically, we use the RoBERTa-base model (Liu et al., 2019) as we find it performs better than the original BERT-base model (Devlin et al., 2019).

	Dev	Test
Jia and Liang (2016)		
No copy	N/A	69.90%
Copy	N/A	76.30%
Copy + data recombination	N/A	**83.30%**
Guo et al. (2020)		
No copy	N/A	68.70%
Copy	N/A	75.70%
Ours		
No copy	80.00%	71.49%
Copy	79.15%	75.63%
Copy + global	81.61%	76.32%
Copy + global + emb	**84.53%**	**78.16%**

Table 1: Exact match accuracy on the ATIS dataset.

	Dev	Test
Yin and Neubig (2018)	N/A	24.35%
+ Reranking	N/A	30.11%
Ours (local)	33.46%	25.84%
+ Reranking	**35.82%**	**28.39%**
+ Global	34.75%	27.08%

Table 2: BLEU score on the CoNaLa dataset.

	Dev Acc.
Rubin and Berant (2020)	73.4%
Yu et al. (2021)	74.7%
Ours (local)	73.79%
+ Global	73.69%

Table 3: Exact match accuracy on the Spider dataset. Test performance requires submissions to the official website. We report validation performance instead.

2019) and the relation-aware Transformer (Wang et al., 2020) as the encoder and an LSTM as the decoder. The architecture generally follows the work by Xu et al. (2021).

The beam size is set to 20 to search for negative samples, and is set to 5 for inference. The margin Δ in (5) is set to 0.1. We use the Adam optimizer (Kingma and Ba, 2015) with a learning rate of 5e-4 for training.

For both ATIS and CoNaLa datasets, we report the best results on the development sets and the corresponding results on the test set. For the Spider dataset, we only report the results on the development set as the ground truth of the test set is not publicly available.

5 Results

ATIS dataset. Following Yin and Neubig (2017); Sun et al. (2020), we report the exact match accuracy for ATIS. We first replicate locally normalized models with and without the copy mechanism and achieve similar results to Jia and Liang (2016) and Guo et al. (2020), shown in Table 1. This verifies that we have a fair implementation and are ready for the study of global normalization.

We observe that the copy mechanism largely affects the accuracy on the test set, although it has little effect on the development set. This is because the training and validation distributions closely resemble each other, whereas the test distribution differs largely. Therefore, the copy mechanism is important for handling unseen entities in the test set, and our proposed copy variant in Section 3.3 is also essential to globally normalized models.

We then train our model with the max-margin loss. Our globally normalized model consistently improves the accuracy on both development and test sets, compared with its locally normalized counterpart. This shows the effectiveness of our approach.

In addition, we notice that a large number of entities in ATIS have a form like "ap:denvor" (Denver Airport). We thus use the combination of character-level ELMo embeddings (Peters et al., 2018) and word-level GloVe embeddings (Pennington et al., 2014). This further improves the accuracy, which outperforms the previous methods by $\sim$1.9% in the setting without data augmentation.

CoNaLa dataset. For CoNaLa, BLEU is treated as the main metric in previous work (Yin and Neubig, 2019), because accuracy is generally very low (<3%) on this dataset. From Table 2, we observe that our globally normalized model improves the BLEU scores on both the development and test sets compared with the locally normalized baseline. Such improvement is consistent with that on ATIS.

We further compare our model with Yin and Neubig (2019), which reranks beam search results by heuristics. Our method is outperformed by the reranking approach. Note that reranking can be considered as alleviating label bias with postprocessing, as the locally normalized model fails to assign the correct sequence with the highest joint probability. However, the reranking method requires training several reranking scorers, combined with an ad hoc feature (namely, length). By contrast, our global normalization does not rely on ad hoc human engineering.

Spider dataset. Table 3 lists the results on the Spider dataset. Here, our locally normalized model

uses BERT as the encoder, and its performance is on par with that from the recent state-of-the-art approaches (Rubin and Berant, 2020; Yu et al., 2021). However, our global normalization does not improve the performance. It is noted that BERT is a more powerful model than LSTM, and Spider has a much larger training set than CoNaLa and ATIS. We conjuncture that BERT learns the step-by-step local prediction probability very well, which in turn yields a satisfying joint probability and largely mitigates label bias by itself. Therefore, the globally normalized model does not exhibit its superiority on the Spider dataset.

6 Conclusion

In this work, we propose to apply global normalization for neural semantic parsing. Our approach predicts the score of different grammar rules at an autoregressive step, and thus it does not suffer from the label bias problem. We observe that our proposed method is able to improve performance on small datasets with LSTM-based encoders. However, global normalization becomes less effective on the large dataset with a BERT architecture.

Acknowledgments

We acknowledge the support of the Mitacs Accelerate Program (Ref: IT16065) and the Natural Sciences and Engineering Research Council of Canada (NSERC) under grant Nos. RGPIN-2020-04465 and RGPIN-2020-04440. Chenyang Huang is supported by the Borealis AI Graduate Fellowship Program. Lili Mou and Osmar Zaïane are supported by the Amii Fellow Program and the Canada CIFAR AI Chair Program. This research is also supported in part by Compute Canada (www.computecanada.ca).

References

Daniel Andor, Chris Alberti, David Weiss, Aliaksei Severyn, Alessandro Presta, Kuzman Ganchev, Slav Petrov, and Michael Collins. 2016. Globally normalized transition-based neural networks. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 2442–2452.

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

Deborah A. Dahl, Madeleine Bates, Michael Brown, William Fisher, Kate Hunicke-Smith, David Pallett, Christine Pao, Alexander Rudnicky, and Elizabeth Shriberg. 1994. Expanding the scope of the ATIS task: The ATIS-3 corpus. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.

Hal Daumé and Daniel Marcu. 2005. Learning as search optimization: Approximate large margin methods for structured prediction. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 169–176.

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*, pages 4171–4186.

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*, pages 33–43.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 1631–1640.

Jiaqi Guo, Qian Liu, Jian-Guang Lou, Zhenwen Li, Xueqing Liu, Tao Xie, and Ting Liu. 2020. Benchmarking meaning representations in neural semantic parsing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1520–1540.

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.

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*, pages 12–22.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

Tom Kwiatkowksi, Luke Zettlemoyer, Sharon Goldwater, and Mark Steedman. 2010. Inducing probabilistic CCG grammars from logical form with higher-order unification. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1223–1233.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543.

Matthew 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*, pages 2227–2237.

Ohad Rubin and Jonathan Berant. 2020. SmBoP: Semi-autoregressive bottom-up semantic parsing. *arXiv preprint arXiv:2010.12412*.

Miloš Stanojević and Mark Steedman. 2020. Max-margin incremental CCG parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4111–4122.

Zeyu Sun, Qihao Zhu, Lili Mou, Yingfei Xiong, Ge Li, and Lu Zhang. 2019. A grammar-based structural cnn decoder for code generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 7055–7062.

Zeyu Sun, Qihao Zhu, Yingfei Xiong, Yican Sun, Lili Mou, and Lu Zhang. 2020. TreeGen: A tree-based transformer architecture for code generation. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 8984–8991.

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.

Sam Wiseman and Alexander M. Rush. 2016. Sequence-to-sequence learning as beam-search optimization. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1296–1306.

Peng Xu, Dhruv Kumar, Wei Yang, Wenjie Zi, Keyi Tang, Chenyang Huang, Jackie Chi Kit Cheung, Simon JD Prince, and Yanshuai Cao. 2021. Optimizing deeper transformers on small datasets. *arXiv preprint arXiv:2012.15355*.

Pengcheng Yin, Bowen Deng, Edgar Chen, Bogdan Vasilescu, and Graham Neubig. 2018. Learning to mine aligned code and natural language pairs from stack overflow. In *Proceedings of the International Conference on Mining Software Repositories*, pages 476–486.

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*, pages 440–450.

Pengcheng Yin and Graham Neubig. 2018. TRANX: A transition-based neural abstract syntax parser for semantic parsing and code generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 7–12.

Pengcheng Yin and Graham Neubig. 2019. Reranking for neural semantic parsing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4553–4559.

Tao Yu, Chien-Sheng Wu, Xi Victoria Lin, bailin wang, Yi Chern Tan, Xinyi Yang, Dragomir Radev, richard socher, and Caiming Xiong. 2021. GraPPa: Grammar-augmented pre-training for table semantic parsing. In *The International Conference on Learning Representations*.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. 2018. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 3911–3921.

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

Luke Zettlemoyer and Michael Collins. 2007. Online learning of relaxed CCG grammars for parsing to logical form. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 678–687.

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