Learning Logical Representations from Natural Languages with Weak Supervision and Back-Translation

Kaylin Hagopian[†][*]
Georgia Institute of Technology
Atlanta, GA, USA
khagopian3@gatech.edu

Qing Wang[†]
IBM Research
Yorktown Heights, NY, USA
qing.wang1@ibm.com

Tengfei Ma
IBM Research AI
Yorktown Heights, NY, USA
tengfei.ma1@ibm.com

Yupeng Gao
IBM Research AI
Yorktown Heights, NY, USA
yupeng.gao@ibm.com

Lingfei Wu
IBM Research AI
Yorktown Heights, NY, USA
wuli@us.ibm.com

Abstract

Semantic parsing with strong supervision requires a large parallel corpus for training, whose production may be impractical due to the excessive human labour needed for annotation. Recently, weakly supervised approaches based on reinforcement learning have been studied, wherein only denotations, but not the logical forms, are available. However, reinforcement learning approaches based on, e.g., policy gradient training, suffer reward sparsity and progress slowly. In this work, we explore enriching the policy gradient framework with a back-translation mechanism that boosts the reward signal. In this vein, we jointly optimize the expected reward of obtaining the correct denotation and the likelihood of reconstructing the natural language utterance. Initial experimental results suggest the proposed method slightly outperforms a state-of-the-art semantic parser trained with policy gradient.

Introduction

Semantic parsing is the task of converting natural language utterances to logical representations that a machine can understand and execute as problem solving steps. There has recently been a surge of interest in developing machine learning methods for semantic parsing, owing to the creation of large-scale text corpora. There has also been a proliferation of logical representations used in this setting, including logical forms [28], domain specific languages [26], python code [24], SQL queries [31], and bash commands [20]. A grand vision in AI that this work supports is the automatic translation of natural languages to a logical formalism in which a machine may reason with knowledge.

Early work on semantic parsing applied supervised learning approaches, wherein a natural language utterance is paired with a gold-standard program, which serves as ground truth for training [28, 12, 29, 30]. Such strong supervision, however, requires expert human effort to annotate a text corpus, which becomes prohibitively expensive at the corpus scales needed.

Semantic parsing is, by its nature, a machine translation problem, and shares the difficulties faced for low-resource natural languages. Recently, several efforts have tackled these problems using

[*]This work was done during the author's internship in IBM Research.
[†]Equal contribution

only monolingual data [16, 1]. These approaches typically use a combination of back-translation and adversarial training to escape the requirement of parallel alignment between source and target languages. These ideas parallel extensions [22, 32, 33, 27, 21] of adversarial generative models applied in other fields such as computer vision, where image-to-image translation represents a similar scenario.

Semantic parsing, on the other hand, is differentiated by the fact that some natural language utterances, in the form of a question, may come furnished with an answer (sometimes called a *denotation*), even though the program or logical query that would yield it is unknown. Collecting question-answer pairs is much less labor-intensive than producing gold-standard code or logic, and generally does not require expertise in knowledge representation or programming. These denotations have been used as a source of weak supervision in much of the previous work on logical-form generation; see, e.g., [6, 19, 15, 3, 4, 2].

In this work, we follow this weakly supervised setting and explore the use of reinforcement learning to generate programs [17, 23, 11, 18, 9]. Since gold-standard programs are unknown, reinforcement learning matches the execution results of generated programs with ground-truth *denotata* to reward (either positively or negatively) the translation model as it learns. Such a system may be trained with policy gradient but this is known to be challenging because the gradient estimate exhibits high variance and positive reward signals are generally quite weak initially.

Hence, we propose to use back-translation to encourage the production of correct programs by boosting the availability of positive reward signals. Back-translation has been successfully applied in several other settings; see, e.g., [1] for unsupervised machine translation, [32] for image-to-image translation, and [13] for semi-supervised semantic parsing. In the setting of reinforcement learning, the likelihood of reconstruction through back-translation serves as a source of incremental rewards, which complement the rarely obtained positive reward resulting from correct program execution and drives faster progress in model training. Good reconstructions may also substitute for programs residing in the memory buffer in the memory-augmented policy gradient approach, MAPO [18], a state-of-the-art weakly supervised semantic parser. Our initial empirical results show that the proposed approach slightly improves over MAPO for SQL generation tasks.

Related Work

Semantic parsing with weak supervision has recently attracted growing interest; a basic architecture used is a sequence-to-sequence (seq2seq) model, but because of the lack of gold-standard programs, models are often trained using reinforcement learning, the simplest algorithm for which being REINFORCE. Neural Symbolic Machines [17] augment this algorithm by interleaving it with maximum-likelihood training, forming an iterative procedure that resembles expectation maximization. Additionally, a key-variable memory is introduced to store intermediate execution results, ensuring program compositionality. MAPO [18] extends Neural Symbolic Machines with additional memory that stores correctly executed programs. This memory is used to reduce the variance of the policy gradient estimates and to compose the marginal likelihood objective. [11] establish a connection between the gradient estimates of REINFORCE and those of the marginal likelihood and propose an improved algorithm to tackle the problem of spurious programs, through randomizing beam search and promoting equal update weights for correct programs. [9] also focuses on the challenge of spurious programs. Their solution is to lift natural language utterances to abstract forms so that utterances with different surface forms but similar meaning are more likely to be decoded into correct programs simultaneously. Beyond combining with marginal likelihood, [23] improves the basic REINFORCE training through the use of paired neural networks, one serving as a distributed executor and the other a symbolic one. The authors use the distributed executor's intermediate execution results to pretrain the symbolic executor for an initial policy and then use the REINFORCE algorithm to improve the policy. Such an approach is well suited for answering table queries, where the programs are simply row and column operations.

If denotations are not present, semantic parsing may be approached in an unsupervised manner. In this vein, machine translation without parallel data is most relevant. In [16], the idea is to map sentences in two different languages into the same latent space by using a single encoder and single decoder. The training loss includes the reconstruction error in autoencoding, the cross entropy in cross-domain translation, and the domain prediction error of an adversarial discriminator. [1] propose a similar use of a shared encoder, but the decoders are different among languages. The training techniques include the use of pre-trained cross-lingual embedding and on-the-fly back-translation.

Lack of parallel data is also being actively addressed in computer vision, where a proliferation of work extends the basic generative adversarial network (GAN) architecture [10] to fuse cross-domain information. CoGAN [22] learns a joint distribution of images from different domains by sharing the same latent space and same generator weights in the first few layers. The weight-sharing scheme limits the network capacity and favors a joint distribution solution over a product-of-marginal-distributions one. If x is an image and G is a mapping from one domain to the other, CycleGAN [32] imposes a cycle consistency through jointly training an inverse mapping F such that $F(G(x)) \approx x$. Cycle consistency is extensively explored in image-to-image translation, models including DualGAN [27], BiCycleGAN [33], and an improvement [21] of CoGAN; as well as in text style transfer [8].

In addition to the weakly supervised and unsupervised settings, many semantic parsing methods trained in a semi-supervised or supervised manner offer inspiration, particularly in terms of the neural architecture. SEQ4 [13], a chain of two seq2seq models with neural attention, is the basis of our back-translation scheme. The first model translates a logical form to natural language whereas the second one translates it back to the logical form. A key to this architecture is the reparameterization trick used for decoding the natural language so that the gradient can flow through. [7] propose another approach that also employs two seq2seq models, but in a supervised manner. This approach defines sketches, which are a coarse representation of the logical form. The first seq2seq is used to generate the sketch, whereas the second one completes the logical form conditioned on the sketch and the natural language input. Such a coarse-to-fine scheme ensures the correct translation of key information captured by the sketch.

Method

We propose using back-translation to enhance policy gradient training. In this section, we first briefly review the standard seq2seq model and the basic reinforcement learning setting. Then, we introduce the proposed application of back-translation.

Basic Seq2seq Model

Given an input sequence $x = (x_1, x_2, \ldots, x_T)$, seq2seq defines a neural architecture that emits a sequence $z = (z_1, z_2, \ldots, z_{T'})$ with probability model $q(z|x)$. Through successive conditioning, the probability model is decomposed as

$$q(z|x) = \prod_{t=1}^{T'} q(z_t | z_{<t}, x). \tag{1}$$

In this work, x denotes the natural language utterance and z denotes the program. The architecture consists of an encoder, and a decoder with attention.

Reinforcement Learning

Denote by ϕ the parameters of the above seq2seq model $q(z|x)$. Given a natural language utterance x and its corresponding denotation y, the ground-truth program z is unknown. Hence, it is infeasible to learn ϕ only by optimizing the probability model $q_\phi(z|x)$ with maximum likelihood, because the resulting decoded program z does not necessarily execute to y. However, a reward $R(z)$ can be defined for any decoded program z:

$$R(z) = \begin{cases} 1, & \text{if } z \text{ executes to } y, \\ 0, & \text{otherwise.} \end{cases}$$

Hence, we may learn the model through reinforcement learning: maximize the expected reward with respect to the probability model $q_\phi(z|x)$, termed *policy*.

Specifically, for semantic parsing, because the number of possible outcomes z is finite, we write the expected reward per training example x as

$$E_\phi(x) = \sum_z R(z) q_\phi(z|x).$$

Since optimization is typically done by using stochastic gradient methods, one needs an (unbiased) estimator of the gradient (called *policy gradient*)

$$\nabla_\phi E_\phi(x) = \sum_z R(z) \nabla_\phi q_\phi(z|x).$$

The standard approach is to apply the log-derivative trick $\nabla \log q = (\nabla q)/q$ and rewrite the policy gradient as

$$\nabla_\phi E_\phi(x) = \sum_z [R(z) - b] q_\phi(z|x) \nabla_\phi q_\phi(z|x),$$

were b is an arbitrary *baseline* value. The benefit is that even if the output space for z is so large that the summation is impossible to exactly evaluate, one may use Monte Carlo sampling to obtain an unbiased estimator of the gradient and use b to control the variance of the estimator:

$$\nabla_\phi E_\phi(x) \approx \frac{1}{L} \sum_{l=1}^{L} [R(z^{(l)}) - b] \nabla_\phi q_\phi(z^{(l)}|x), \quad z^{(l)} \sim q_\phi(z|x).$$

One working choice for b in practice is $\frac{1}{L}\sum_{l=1}^{L} R(z^{(l)})$.

Memory Augmented Policy Optimization (MAPO)

Reinforcement learning based algorithms often encountered the problems of cold start and sparse signals. The random initialized policies often lead to small rewards; and the random policy samples do not explore the search space efficiently. To solve these problems, [18] proposed to use a memory buffer to augment the policy optimization. The key idea is to re-express the expected return objective as the linear combination of two expectations: one term from the memory, and the other used to explore unknown trajectories.

$$E_\phi(x) = \sum_{z \in \mathcal{B}} R(z) q_\phi(z|x) + \sum_{z \notin \mathcal{B}} R(z) q_\phi(z|x)$$

They use enumeration to estimate the gradient of the first term and use Monte Carlo sampling with a rejection scheme to estimate the gradient of the second term.

Back-Translation

Although MAPO reduced the impact of the aforementioned issues, there are still some challenges in the training process. For example, a challenge for the policy gradient approach is that it is difficult to obtain programs $z^{(l)}$ that earn a non-zero reward $R(z^{(l)})$ initially, when the policy $q_\phi(z|x)$ is random (in the starting phase of training). Our approach is to train a second seq2seq model $p_\theta(x|z)$ jointly, which ensures that the decoded sequence z can be translated back to x. Such a cycle (Fig. 1) consistently filters out infeasible translations and narrows the space that must be searched for a reward-earning z.

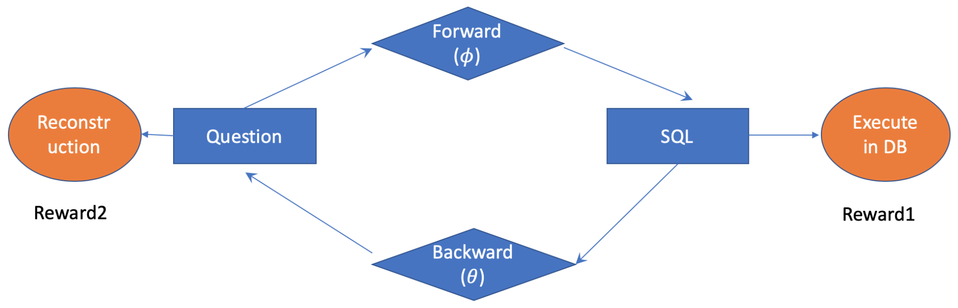

Figure 1: The additional back-translation rewards in our framework.

Algorithm 1 Back-Translation Enhanced Weakly Supervised Text-to-SQL

```
Input: Data $\{x_i, y_i\}$ for $1 \leq i \leq n$; memory and exploration buffer $\mathcal{B}_i, \mathcal{B}_i^e$ for $1 \leq i \leq n$; constants $\alpha, \epsilon, M$
repeat for all actors:
  Initialize training batch $D \longleftarrow \Phi$
  Get a batch of inputs $C$
  for $(x_i, y_i, \mathcal{B}_i, \mathcal{B}_i^e) \in C$ do
    SystematicExploration($x_i, q_\phi^{old}, \mathcal{B}_i^e, \mathcal{B}_i$)
    Sample $z_i^+ \sim q_\phi^{old}$ over $\mathcal{B}_i$
    $w_i^+ = \max(q_\phi^{old}(\mathcal{B}_i), \alpha)$
    Add sample $(z_i^+, R_f(z_i^+), R_b(z_i^+), w_i^+)$ to $D$
    Sample another $z_i \sim q_\phi^{old}$
    if $z_i \notin \mathcal{B}_i$ then
      $w_i = 1 - w_i^+$
      Add $(z_i, R_f(z_i), R_b(z_i), w_i)$ to $D$
    end if
  end for
  Push $D$ to training queue
until converge
Repeat for the learner:
  Get a batch $D$ from training queue
  for $(z_i, R_f(z_i), R_b(z_i), w_i) \in D$ do
    $d\phi = d\phi + w_i((1-\alpha)R_f(z_i) + \alpha R_b(z_i))\nabla \log q_\phi(z_i)$
    $d\theta = d\theta + w_i R_b(z_i)\nabla \log q'_\theta(z_i)$
  end for
  Fix $\theta$, update $\phi = \phi + d\phi$, $\phi_\phi^{old} = q_\phi$
  Fix $\phi$, update $\theta = \theta + d\theta$
until: Converge or a certain number of steps is reached
Output: Final parameters $\phi$ and $\theta$
```

The architecture of the second seq2seq can be defined re-using Equation (1) with mathematical notations q, x, z, T, T', ϕ replaced by q', z, x, T', T, θ, respectively. We follow the MAPO framework, and add additional rewards for the generated samples outside the memory. So in the forward loop, the reward $R(z)$ is changed into $R(z) = (1-\alpha)R_f(z) + \alpha R_b(z)$.

We can update the forward SQL generation model and the backward-translation model iteratively. First, we fix the parameters of the back-translation model, the policy gradient for the forward model is thus

$$((1-\alpha)R_f(z_i) + \alpha R_b(z_i))\nabla \log q_\phi(z_i|x).$$

We can use this to update the forward parameters ϕ and the forward policy $q_\phi(z|x)$.

Then we fix the parameters for the forward model, and update the backward-translation model. In this case, we already have the generated sample z as the fixed input and we know the target translation x, so we do not need to use the forward reward score $R_f(z)$ here. Only the backward reward will impact the updating of the backward model parameters θ. The backward policy gradient is $R_b(z_i)\nabla \log q'_\theta(z_i)$. We summarize the algorithm in Algorithm 1.

Experiments

Dataset

We evaluated our model using the WikiTableQuestions dataset [25], which consists of 2,108 Wikipedia tables (containing column names and cell entries) and 22,033 <*question, answer*> pairs pertaining to the tables. The tables in the test set contain column names not seen in the training set. The goal of the dataset is to take an English *question* and produce a *program* that, when executed, produces the correct *answer*.

Model architecture

For the forward-translation model, we used the MAPO algorithm (as described in [18]). The back-translation seq2seq model (batch size: 10) was trained on $<$*question, MAPO-predicted program*$>$ pairs associated with a forward-translation (execution) reward of 1. The back-translation model then calculated a back-translation reward for each trajectory in the training queue which the MAPO learner linearly combined with the forward-translation reward before updating its model's parameters. The overall reward for a $<$*question, MAPO-predicted program*$>$ was as follows:

$$R = (1 - \alpha)R_f + \alpha R_b$$

where α denotes the back-translation reward weight (set to 0.01 based on experimentation), R_f is the forward-translation reward, and R_b indicates the back-translation reward.

To calculate the back-translation reward, the back-translation seq2seq model took $<$*question, MAPO-predicted program*$>$ pairs and translated the *MAPO-predicted programs* back into *predicted questions*. Back-translation reward was the similarity score between the *predicted question* and the actual *question*. Similarity was determined using a weighted linear combination of three accuracy metrics: BLEU score (using uni- and bigrams with equal weights), the Universal Sentence Encoder metric [5], and Word Mover's Distance [14]. Each accuracy metric had its own weight (heuristically set to 1, 0.1, and 0.01, respectively).

Pre-Training Back-translation Model

In the initial phase, the back-translation model has random parameters, so the additional back-translation reward may take in a lot of noise. To make the back-translation module helpful and easily updated, we first pre-train a SQL2Text model using some intermediate results and then initialize the back-translation model using the pre-trained model. To train this model, we used a MAPO learner (trained on the WikiTableQuestions training dataset) to generate $<$*question, program*$>$ pairs with forward-translation rewards of 1 (N=8,076). The tokens of these pairs were converted into subwords using OpenNMT's Byte Pair Encoding method (see the Appendix for a description of the experiment used to deem subwording was appropriate/useful). An 80-20 training/validation split was used to select the best model, which was subsequently used as the pre-trained back-translation seq2seq model. The best model had a BLEU score (focusing on uni- and bigrams) of 0.3805 for the validation set.

Results

When training its learner, the MAPO algorithm only considers forward-translation (execution) reward associated with a *predicted program* – either 1 (correct *answer* produced) or 0 (incorrect *answer* produced). To determine the effect of back-translation on performance, we compared our model's performance with that of MAPO's and other reinforcement learning based methods: REINFORCE, MML (Maximum Marginal Likelihood), Hard EM, IML [18]. All the baseline results are cited from their original paper.

Table 1 summarizes the results for the baselines and MAPO+back-translation model, reporting the results based on 5 runs. Figure 2 provides one example for comparison of the performance curves for MAPO and MAPO+back-translation.

Table 1: The mean accuracy and standard deviation on the dev set of WikiTable based on 5 runs.

REINFORCE	≤ 10
MML (Soft EM)	39.7 ± 0.3
Hard EM	39.3 ± 0.6
IML	36.8 ± 0.5
MAPO	42.3 ± 0.3
MAPO+bt	42.6 ± 0.5

Discussion

Overall, the performance of the original forward-translation and the new forward+back-translation models are relatively similar, with the model that incorporates back-translation slightly outperforming

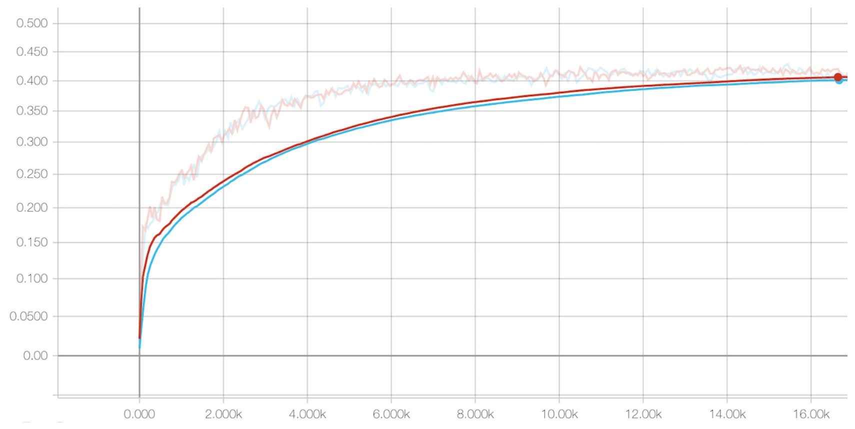

Figure 2: Performance curves for the MAPO (blue) and MAPO+back-translation (red) models. The results have been smoothed by a factor of 0.9.

the baseline. It is likely that part of the reason the back-translation model's performance is so close to the baseline is because the final back-translation model's performance was still relatively low (obtaining BLEU scores <0.4), thus not a great measure for the additional reward. However, as mentioned previously, the validation performance for the MAPO+back-translation model is slightly higher than the baseline MAPO implementation, and from the learning curves we can see it learns a little bit faster – suggesting back-translation may be useful. We hope this work can be a good starting point. In the future, we will explore ways to improve the back-translation model, or to use other methods to obtain additional rewards.

References

[1] Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. Unsupervised neural machine translation. In *ICLR*, 2018.

[2] Yoav Artzi and Luke Zettlemoyer. Weakly supervised learning of semantic parsers for mapping instructions to actions. *Transactions of the Association for Computational Linguistics*, 1:49–62, 2013.

[3] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on freebase from question-answer pairs. In *EMNLP*, 2013.

[4] Qingqing Cai and Alexander Yates. Large-scale semantic parsing via schema matching and lexicon extension. In *ACL*, 2013.

[5] Daniel Cer, Yinfei Yang, Sheng yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. Universal sentence encoder. arXiv:1803.11175v2, 2018.

[6] James Clarke, Dan Goldwasser, Ming-Wei Chang, and Dan Roth. Driving semantic parsing from the world's response. In *CoNLL*, 2010.

[7] Li Dong and Mirella Lapata. Coarse-to-fine decoding for neural semantic parsing. In *ACL*, 2018.

[8] Cicero Nogueira dos Santos, Igor Melnyk, and Inkit Padhi. Fighting offensive language on social media with unsupervised text style transfer. In *ACL*, 2018.

[9] Omer Goldman, Veronica Latcinnik, Ehud Nave, Amir Globerson, and Jonathan Berant. Weakly supervised semantic parsing with abstract examples. In *ACL*, 2018.

[10] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, 2014.

[11] Kelvin Guu, Panupong Pasupat, Evan Zheran Liu, and Percy Liang. From language to programs: Bridging reinforcement learning and maximum marginal likelihood. In *ACL*, 2017.

[12] Rohit J. Kate, Yuk Wah Wong, and Raymond J. Mooney. Learning to transform natural to formal languages. In *AAAI*, 2005.

[13] Tomáš Kočiský, Gábor Melis, Edward Grefenstette, Chris Dyer, Wang Ling, Phil Blunsom, and Karl Moritz Hermann. Semantic parsing with semi-supervised sequential autoencoders. In *EMNLP*, 2016.

[14] Matt J Kusner, Yu Sun, Nicholas I Kolkin, and Kilian Q Weinberger. From word embeddings to document distances. In *JMLR*, 2015.

[15] Tom Kwiatkowski, Eunsol Choi, Yoav Artzi, and Luke Zettlemoyer. Scaling semantic parsers with on-the-fly ontology matching. In *EMNLP*, 2013.

[16] Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. Unsupervised machine translation using monolingual corpora only. In *ICLR*, 2018.

[17] Chen Liang, Jonathan Berant, Quoc Le, Kenneth D. Forbus, and Ni Lao. Neural symbolic machines: Learning semantic parsers on freebase with weak supervision. In *ACL*, 2017.

[18] Chen Liang, Mohammad Norouzi, Jonathan Berant, Quoc Le, and Ni Lao. Memory augmented policy optimization for program synthesis with generalization. arXiv:1807.02322, 2018.

[19] Percy Liang, Michael I. Jordan, and Dan Klein. Learning dependency-based compositional semantics. In *ACL*, 2011.

[20] Xi Victoria Lin, Chenglong Wang, Luke Zettlemoyer, and Michael D. Ernst. Nl2bash: A corpus and semantic parser for natural language interface to the linux operating system. In *LREC*, 2018.

[21] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks. In *NIPS*, 2017.

[22] Ming-Yu Liu and Oncel Tuzel. Coupled generative adversarial networks. In *NIPS*, 2016.

[23] Lili Mou, Zhengdong Lu, Hang Li, and Zhi Jin. Coupling distributed and symbolic execution for natural language queries. In *ICML*, 2017.

[24] Yusuke Oda, Hiroyuki Fudaba, Graham Neubig, Hideaki Hata, Sakriani Sakti, Tomoki Toda, and Satoshi Nakamura. Learning to generate pseudo-code from source code using statistical machine translation. In *Proceedings of the 30th IEEE/ACM International Conference on Automated Software Engineering*, 2015.

[25] Panupong Pasupat and Percy Liang. Compositional semantic parsing on semi-structured tables. In *ACL*, 2015.

[26] Chris Quirk, Raymond Mooney, and Michel Galley. Language to code: Learning semantic parsers for if-this-then-that recipes. In *ACL*, 2015.

[27] Zili Yi, Hao Zhang, Ping Tan, and Minglun Gong. Dualgan: Unsupervised dual learning for image-to-image translation. In *ICCV*, 2017.

[28] John M. Zelle and Raymond J. Mooney. Learning to parse database queries using inductive logic programming. In *AAAI*, 1996.

[29] Luke S. Zettlemoyer and Michael Collins. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *UAI*, 2005.

[30] Luke S. Zettlemoyer and Michael Collins. Online learning of relaxed CCG grammars for parsing to logical form. In *EMNLP/CoNLL*, 2007.

[31] Victor Zhong, Caiming Xiong, and Richard Socher. Seq2sql: Generating structured queries from natural language using reinforcement learning. arXiv:1709.00103, 2017.

[32] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017.

[33] Jun-Yan Zhu, Richard Zhang, Deepak Pathak, Trevor Darrell, Alexei A. Efros, Oliver Wang, and Eli Shechtman. Toward multimodal image-to-image translation. In *NIPS*, 2017.