Data Synthesis and Iterative Refinement for Neural Semantic Parsing without Annotated Logical Forms

Shan Wu[1,3], Bo Chen[1], Xianpei Han[1,2], and Le Sun[1,2(✉)]

[1] Chinese Information Processing Laboratory, Chinese Academy of Sciences, Beijing, China
{wushan2018,chenbo,xianpei,sunle}@iscas.ac.cn
[2] State Key Laboratory of Computer Science Institute of Software, Chinese Academy of Sciences, Beijing, China
[3] University of Chinese Academy of Sciences, Beijing, China

Abstract. Semantic parsing aims to convert natural language utterances to logical forms. A critical challenge for constructing semantic parsers is the lack of labeled data. In this paper, we propose a data synthesis and iterative refinement framework for neural semantic parsing, which can build semantic parsers without annotated logical forms. We first generate a naive corpus by sampling logic forms from knowledge bases and synthesizing their canonical utterances. Then, we further propose a bootstrapping algorithm to iteratively refine data and model, via a denoising language model and knowledge-constrained decoding. Experimental results show that our approach achieves competitive performance on GEO, ATIS and OVERNIGHT datasets in both unsupervised and semi-supervised data settings.

Keywords: Semantic parsing · Data synthesis · Unsupervised methods

1 Introduction

Semantic parsing is the task of translating natural language (NL) utterances to their formal meaning representations (MRs), such as lambda calculus [42,50], FunQL [23,28], and SQL queries [5,8,16]. Currently, most neural semantic parsers [12,13] model semantic parsing as a sequence translation task via a encoder-decoder framework. For instance, given an utterance "*What is the length of river traverses state0*", a SEQ2SEQ parsing model obtains its FunQL representation by sequentially generating its tokens `answer(length(river(traverse_2(state0))))`.

One of the key challenges in building a semantic parser is the scarcity of annotated data. Since annotating utterances with MRs is time consuming and requires specialized expert knowledge. Witnessed the data bottleneck problem, there are many learning algorithms have been proposed, such as denotation-based weak supervised learning [29,30], dual learning [6], transfer learning [18,37]. There are also many studies focus on the quick construction of training data, such as OVERNIGHT [40]. However, these works still require some degree of human efforts.

M. Sun et al. (Eds.): CCL 2022, LNAI 13603, pp. 51–65, 2022.
https://doi.org/10.1007/978-3-031-18315-7_4

In this paper, we propose a data synthesis and iterative refinement framework, which can build semantic parsers without labeled data. Inspired by the idea that, a simple and noise corpus can be synthesized by a grammar-lexicon method, like the one used in OVERNIGHT, and can be refined by leveraging external knowledges, like language models and knowledge base constraints. So, we first obtain a naive corpus based on synchronous context-free grammars and a seed lexicon. Then we improve the corpus with the knowledge of language models and knowledge base constraints by iteratively refining data and model to obtain mature corpus. Finally, we use the refined corpus to train the semantic parser. Figure 1 shows the overview of our method.

Specifically, to get the naive corpus, we sample logical forms from knowledge bases, and then synthesize their corresponding canonical utterances using a grammar-based synthesizing algorithm. For example, like in Overnight, we can synthesize an unnatural utterance "*what is length river traverse state0*" from `answer(length(river(traverse_2(state0))))`. Although the synthesized utterance "*what is length river traverse state0*" is different from the real-world utterance "*what is the length of river traverse state0*", the naive corpus can provide a start for unsupervised learning, and can be used to pretrain a base semantic parser.

Then, to improve the synthesized naive corpus, we iteratively refine the model and the data via a bootstrapping process, using the knowledge of language models and knowledge base constraints. Due to the limitation of grammars and seed lexicon, the synthesized training instances in naive corpus are often noisy, differing from real-world utterances, and with limited diversity, which hinder the model from generalizing to natural data. To address these issues, we propose to iteratively refine the model and the synthesized data via a denoising language model and knowledge-constrained decoding. Firstly, we view synthesized canonical utterances as an artificial version of utterances which are often not as fluent as natural utterances, then leverage a denoising language model to rewrite the canonical utterances to be closer to natural utterances. Secondly, to address the noise problem, a knowledge-constrained decoding algorithm is employed to exploit constraints from knowledge bases, therefore meaning representations can be more accurately predicted even when semantic parser is not strong enough. Finally, the *data synthesization* and *semantic parsing* are iteratively refined to bootstrap both the corpus and the semantic parser: the refined corpus is used to train a better semantic parser, and the better semantic parser in turn is used to refine training instances.

The main contributions of this paper are:

- We propose a data synthesis and iterative refinement framework to build neural semantic parsers without labeled logical forms, in which we generate naive corpus from scratch and improve them with the knowledge of language models and knowledge base constraints via an iterative data-model refinement.
- Experimental results on GEO, ATIS and OVERNIGHT datasets show that our approach achieves competitive performance without using annotated data.

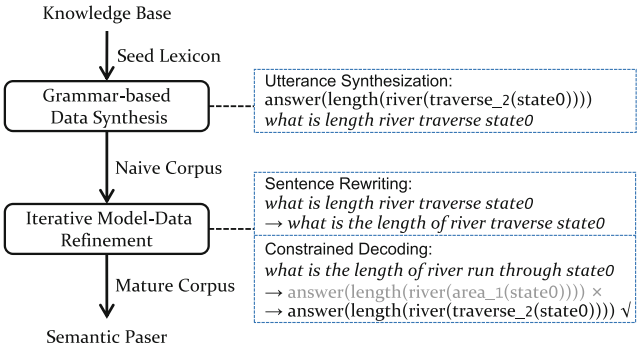

Fig. 1. The overview of our approach.

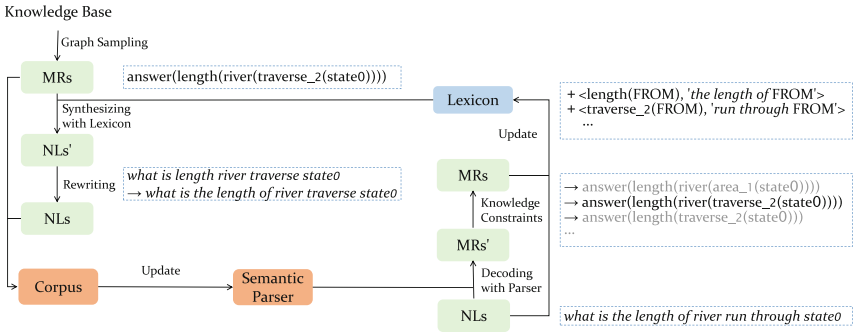

Fig. 2. The illustration of our approach. MRs denotes meaning representations, NLs denotes natural language sentences. The naive corpus is synthesized by seed lexicon. In each bootstrapping iteration, the corpus is refined via denoising language model and knowledge-constrained decoding. The data and the models are improved iteratively.

2 Background

2.1 Base Semantic Parsing Model

We employ the SEQ2SEQ semantic parser as our base model [12], which has shown its simplicity and effectiveness. Notice that our method is not specialized to SEQ2SEQ model and it can be used for any neural semantic parsers.

Encoder. Given a sentence $\mathbf{x} = w_1, w_2, ..., w_n$, the SEQ2SEQ model encodes $\mathbf{x}$ using a bidirectional RNN. Each word w_i is mapped to a fixed-dimensional vector by a word embedding function $\phi(\cdot)$ and then fed into a bidirectional LSTM [19]. The hidden states in two directions are concatenated $\mathbf{h}_i = [\overrightarrow{\mathbf{h}}_i; \overleftarrow{\mathbf{h}}_i]$, and the encoding of the whole sentence is: $\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_n$.

Attention-Based Decoder. Given the sentence representation, the SEQ2SEQ model sequentially generates the tokens of its logical form. Specifically, the decoder is first initialized with the hidden states of encoder $\mathbf{s}_0 = [\overrightarrow{\mathbf{h}}_n; \overleftarrow{\mathbf{h}}_1]$. Then at each step t, let $\phi(y_{t-1})$ be the vector of the previous predicted logical form token, the current hidden state $\mathbf{s}_t$ is obtained from $\phi(y_{t-1})$ and $\mathbf{s}_{t-1}$. Then we calculate the attention weights for the current step t, with the i-th hidden state in the encoder:

$$\alpha_t^i = \frac{\exp\left(\mathbf{s}_t \cdot \mathbf{h}_i\right)}{\sum_{i=1}^{n} \exp\left(\mathbf{s}_t \cdot \mathbf{h}_i\right)} \tag{1}$$

and the next token is generalized from the vocabulary distribution:

$$\begin{aligned} \mathbf{c}_t &= \sum_{i=1}^{n} \alpha_t^i \mathbf{h}_i \\ P(y_t | y_{<t}, \mathbf{x}) &= \text{softmax}(\mathbf{W}_o[\mathbf{s}_t; \mathbf{c}_t] + \mathbf{b}_o) \end{aligned} \tag{2}$$

where $\mathbf{W}_o \in \mathbb{R}^{|V_y| \times 3n}$, $\mathbf{b}_o \in \mathbb{R}^{|V_y|}$ and $|V_y|$ is the output vocabulary size.

Learning. Given a training corpus consisting of <utterance, logical form> pairs, the SEQ2SEQ model is trained by optimizing the objective function:

$$J = -\sum_{(\mathbf{x},\mathbf{y}) \in \mathbf{D}} \sum_{t=1}^{m} \log p(y_t | y_{<t}, \mathbf{x}) \tag{3}$$

where $\mathbf{D}$ is the corpus, $\mathbf{x}$ is the utterance, $\mathbf{y}$ is its logical form label.

2.2 SCFG for Data Synthesization

Wang, Berant, and Liang [40] use a synchronous context-free grammar(SCFG) to generate logical forms paired with canonical utterances, and use crowdsourcing to paraphrase these canonical utterances into natural utterances. The SCFG consists of a set of production rules (lexicon): $N \rightarrow \langle \alpha, \beta \rangle$, where N is a non-terminal, and α and β are sequence of terminal and non-terminal symbols. Any non-terminal symbol in α is aligned to the same non-terminal symbol in β, and vice versa. Therefore, SCFGs define a set of joint derivations of aligned pairs of strings. The seed lexicon in OVERNIGHT is specified by the builder containing types, entities, and properties in databases. Type checking is also performed to rule out some uninterpretable canonical utterances.

3 Approach

This section describes our data synthesis and iterative refinement method for semantic parsing. Firstly, we generate a naive training corpus by sampling meaning representations from knowledge bases and synthesizing their utterances using a grammar-based algorithm. Then, to reduce the noise and eliminate the gap with real corpus, we propose to iteratively refine the data and the model by rewriting synthesized utterances via a denoising language model and generating meaning representations via knowledge-constraint decoding. Figure 2 shows the overview of our approach and we describe all components in detail as follows.

3.1 Data Synthesis

In OVERNIGHT [40] and PARASEMPRE [3], they use simple grammars to generate logical forms paired with canonical utterances. To generate corpus from scratch, we also synthesize data via a grammar-based algorithm.

Specifically, we first sample MRs from knowledge bases via a graph sampling algorithm, then we synthesize their utterances by mapping predicates to words from a seed lexicon and composing these words using context free grammars. Different from the corpus generation method in OVERNIGHT, our method starts from not only grammar but also the knowledge base schema, and can be easier to extended to other datasets like GEO and ATIS.

Generating MRs via Graph Sampling. The graph sampling algorithm aims to sample meaning representations from knowledge bases. Given a knowledge base, Graph Sampling regards MRs as subgraphs of the knowledge base. To ensure the truthfulness and integrality of generated meaning representations, we sample subgraph-based MRs according to both the structure of MRs and the schemas of knowledge bases.

Specifically, to generate MRs, we start from the nonterminal token `root` and then recursively expand all nonterminal tokens in current MRs. For general/functional nonterminal tokens such as `root`, `argmax` and `count`, because they are domain-independent, we expand them using hand-crafted general production rules. For nonterminal tokens about entities and relations such as `river`, `state` and `city` for GEO, because they are domain dependent, we expand them by production rules sampled from knowledge base schemas.

To utilize the schema to produce MRs, we extend the original schema by adding the attribute value as value type nodes and the aggregation operations as self-loop edges. We provide the extended schema and sampling examples in the Fig. 3.

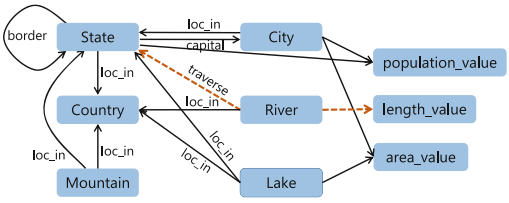

Fig. 3. The extended schema of GEO (partial). To sample the subgraph from the dotted edges, the root nonterminal token `root` is recursively extended by the production rules:
`root` → `answer(length_value)`
`length_value` → `length(river_set)`
`river_set` → `river(river_attri)`
`river_attri` → `traverse_2(state_set)`
`state_set` → `state0`,
generating the MR: `answer(length(river(traverse_2 (state0))))`

Based on the schema graph, the meaning representations can be effectively sampled by utilizing context-free grammar (i.e., the production rules) for grammatical correctness and knowledge base schemas for semantic correctness.

Synthesizing Utterances via SCFG-Based Algorithm. Based on canonical compositionality assumption in Wang, Berant, and Liang [40], we also use SCFG to generate utterances. We extend the context-free grammar in Graph Sampling to synchronous context-free grammar. For example in Fig. 2, based on the SCFG rules, we can synthesize the utterance "what is length river traverse state0" from the sampled MR:

$$\texttt{root} \rightarrow \langle \texttt{answer(}\textsc{Form}\texttt{)}, \text{what is } \textsc{Form} \rangle$$
$$\textsc{Form} \rightarrow \langle \texttt{length(}\textsc{Form}\texttt{)}, \text{length } \textsc{Form} \rangle$$
$$\textsc{Form} \rightarrow \langle \texttt{river(}\textsc{Form}\texttt{)}, \text{river } \textsc{Form} \rangle$$
$$\textsc{Form} \rightarrow \langle \texttt{traverse\_2(}\textsc{Form}\texttt{)}, \text{traverse } \textsc{Form} \rangle$$
$$\textsc{Form} \rightarrow \langle \texttt{state0}, \text{state0} \rangle$$

Seed Lexicon Construction. To synthesize utterances from sampled semantic representations, a lexicon is further needed for SCFG, which maps logical tokens to their natural language words. For OVERNIGHT, we simply use its original seed lexicon. For other datasets, we use the following simple way to build an initial lexicon:

For domain-general logical tokens we manually write their natural language templates. The number of domain-general rules is usually very small. Some examples of our domain-general rules are in Table 1.

Table 1. Examples of our domain-general rules on GEO (above) and ATIS (below). We write seed lexicon of domain-general grammar manually, the number of which is usually very small (only 5 needed in GEO and 12 in ATIS and 23 in OVERNIGHT).

Category	Domain-general rules	NL templates
Query	answer (FORM)	what is FORM
Count	count (FORM)	the number of FORM
Exclusion	exclude (FORM_1, FORM_2)	FORM_1 do not FORM_2
Superlative (max)	largest_one (VALUE (FORM))	FORM with largest VALUE
Filter (type)	λtλs: ($t $s)	$t $s
Filter (property)	λpλvλs: ($p $v $s)	$s whose $p is $v
Comparative ($<$)	λpλvλs: ($<$ ($p $v) $s)	$s whose $p is smaller than $v
Superlative(max)	λpλs: argmax $s ($p $s)	$s with largest $p

For domain-dependent entity tokens and relation tokens, we simply use the words in their logical tokens, with a simple preprocessing which removes numbers and underlines. For example, the `area_1` denotes the words "area" and `departure_time` denotes the words "departure time".

Using the above SCFG with seed lexicon, an initial training corpus can be synthesized. Although, this seed lexicon is obviously with limited coverage and lack of diversity. This naive corpus can still provide a helpful start for semantic parsing. Next, we describe how to iterative refine the parsing mode and data.

3.2 Iterative Data-Model Refining

Due to the limitation of grammar and lexicon, the synthesized training instances in naive corpus are often noisy, differing from real-world sentences, and with limited diversity. To address these issues, we refine the corpus with the knowledge of language models and knowledge base constraints through a bootstrapping process: 1) we rewrite synthesized utterances via a denoising language model, so the utterances will be more fluent and closer to natural utterances; 2) we propose to exploit knowledge during decoding, so that meaning representations can be more accurately predicted even when the model is not strong enough; 3) we iteratively refine the data and the model via a bootstrapping process. After several iterations of refinement, we obtain the mature corpus and the final semantic parser.

Utterance Rewriting via Denoising Language Model. The synthesized utterances are often not fluent, differing from real-world sentences. For example, the synthesized utterance in Fig. 2: "*what is length river traverse state*" is very different to its natural expression "*what is the length of river traverses state0*". And this discrepancy misleads models to learn incorrect patterns.

Thanks to the current powerful language models, we can use a denoising language model to rewrite synthesized utterances to more natural sentences. Specifically, we regard the synthesized utterances as a noisy version of natural expressions, and then denoise them via neural language model-based language denoising techniques [26].

Specifically, we train a language model based on GPT2.0 [34], which is then used to denoise by minimizing:

$$\mathcal{L}^{lm} = \mathbb{E}_{x \sim \mathbf{X}}[-\log P(x|C(x))] \tag{4}$$

where C is a noise model with some words dropped and swapped as in Lample et al. [26].

Generating High-Quality Lexicon via Knowledge-Constrained Decoding. To obtain high-quality lexicon, which can be used to synthesize better ⟨MR, canonical utterance⟩ pairs, we use the current parser to generate parallel data. Without manually annotated corpus, the initial semantic parser is often not strong enough, therefore it is difficult to find high-quality meaning representations. So we also apply knowledge-constrained decoding.

Like previous work [25,44,47], we decode the meaning representations under the grammar we mentioned in Graph Sampling. Only the grammatical logical forms are generated during the decoding. Additionally, we leverage knowledge base schemas to effectively filter out illegal logical forms. Given a semantic parser, we first obtain the top K meaning representations for each sentence. Then if there exists an executing program or search engine for logical forms, we will only keep the executable logical forms. Otherwise, we verify whether the logical form is well-typed under the knowledge base schema constraints, and only preserve the eligible logical forms.

After obtaining the higher quality parallel data, following Wong and Mooney [41], we apply the GIZA++ on the parallel data to get the alignments between words and grammar rules and induce a new SCFG lexicon.

Table 2. Accuracies on OVERNIGHT. The previous methods with superscript $*$ means they use different unsupervised settings.

		Bas.	Blo.	Cal.	Hou.	Pub.	Rec.	Res.	Soc.	Avg.
Supervised										
SEQ2SEQ		84.3	57.9	78.1	69.9	76.2	80.7	78.0	80.5	75.7
RECOMBINATION [21]		85.2	58.1	78.0	71.4	76.4	79.6	76.2	81.4	75.8
CROSSDOMAIN [37]		86.2	60.2	79.8	71.4	78.9	84.7	81.6	82.9	78.2
SEQ2ACTION [9]		88.2	61.4	81.5	74.1	80.7	82.9	80.7	82.1	79.0
DUAL [6]		87.5	63.7	79.8	73.0	81.4	81.5	81.6	83.0	78.9
Unsupervised (with nonparallel data)										
Two-stage [7]		64.7	53.4	58.3	59.3	60.3	68.1	73.2	48.4	60.7
WmdSamples [7]		31.9	29.0	36.1	47.9	34.2	41.0	53.8	35.8	38.7
Mature Corpus + Samples		58.5	55.3	62.4	65.1	66.7	62.2	72.3	47.1	**61.2**
Unsupervised										
Cross-domain Zero Shot* [18]		-	28.3	53.6	52.4	55.3	60.2	61.7	-	-
GENOVERNIGHT [40]		15.6	27.7	17.3	45.9	46.7	26.3	61.3	9.7	31.3
Naive Corpus	EMBED BERT	15.9	24.6	18.6	44.1	46.9	27.0	62.2	9.7	31.1
	Glove	16.2	23.6	16.2	30.3	36.9	27.0	43.2	9.2	25.3
	Rand	13.8	21.1	15.6	28.2	21.9	27.0	31.1	8.2	20.9
Mature Corpus	EMBED BERT	**45.9**	**52.5**	**52.7**	**58.5**	**61.9**	**52.1**	**69.8**	**33.6**	**53.4**
	Glove	44.1	51.5	48.5	56.4	58.8	50.2	68.9	32.0	51.3
	Rand	35.1	43.2	36.5	44.7	46.9	46.5	65.0	25.6	42.9
	w/o Denoising	32.8	45.0	40.1	46.8	52.5	45.6	63.1	26.6	44.1
	w/o Constraint	29.0	39.7	35.3	37.8	41.9	42.8	64.7	23.4	39.3

Iterative Learning. It is obviously that the model promotion and the data refining can reinforce each other: better parsers can generate data of higher quality, and higher quality data can be used to train stronger models. Based on this intuition, we propose to iteratively refine model and data by leveraging the duality between them.

Specifically, in each data-model refining iteration, we: 1) first synthesize the utterances $\mathbf{X}'$ of the sampled MRs $\mathbf{Y}'$ using the current lexicon and the denoising model; 2) train a new semantic parser using the synthesized data; 3) parse the unlabeled utterances via knowledge-constrained decoding; 4) induce a new lexicon using both the highly confident automatically labeled data and the synthesized data.

We gradually increase the proportion of parsing data at each iteration. In the k-th iteration, we select the top $\delta \times (k+1)$ confident parsing pairs for lexicon learning. The confidence scores are calculated as the normalized likelihood:

$$Score(x, y) = \frac{1}{N_y} \log P(y|x) \tag{5}$$

4 Experiments

4.1 Experimental Settings

Datasets. We conduct experiments on three standard datasets: GEO, and ATIS, OVERNIGHT, which use different meaning representations and contain different domains.

GEO. This is a semantic parsing benchmark about U.S. geography [49]. The variable-free semantic representation FunQL [23] is used in this dataset. We follow the standard 600/280 train/test instance splits.

ATIS. This is a large dataset, which contains 5,410 queries to a flight booking system. Each question is annotated with a lambda calculus query. Following Zettlemoyer and Collins [51], we use the standard 4,473/448 train/test instance splits in our experiments.

OVERNIGHT. OVERNIGHT contains natural language paraphrases paired with lambda DCS logical forms across eight domains. We evaluate on the standard train/test splits as Jia and Liang [40].

In all our experiments, we only use the unlabeled sentences in each dataset. The standard accuracy is used to evaluate different systems, which is obtained as the same as Jia and Liang [21].

Synthesized Training Corpus. We generate training instances proportional to the original dataset sizes (1500 for GEO, 5000 for ATIS, and 1500 for each domain in OVERNIGHT). For OVERNIGHT, we use its original defined grammar and lexicon.

Denoising Language Model. We train an individual denoising language model for each dataset (each domain for OVERNIGHT). For each utterance in unlabeled queries, we sample 5 noisy sentences to construct the training pairs by dropping words randomly or slightly shuffling the utterance as Lample et al. [26]. The pretrained language model GPT2.0 is adapted on paraphrase generation dataset, then fine-tuned on denoising sentences with 15 epochs and the learning rate of 1e-5.

System Settings. We train all our models with 5 data-model refining iterations. In each iteration, the neural semantic parser is trained 15 epochs, with the initial learning rate of 0.001. We use Adam algorithm [24] to update parameters, with batch size is 20. Our model uses 200-dimensional hidden units and 200-dimensional word vectors for sentence encoding. We initialize all parameters by uniformly sampling within $[-0.1, 0.1]$. $\text{BERT}_{\text{LARGE}}$ [11] is used to get word representations. The beam size K during decoding is 5. The hyper-parameter δ is 0.1. Following Dong and Lapata [12], we handle entities with a Replacing mechanism, which replaces identified entities with their types and IDs.

4.2 Experimental Results

Overall Results. We compare our model with different settings:

1) **Naive Corpus** – the semantic parser is trained from the naive corpus, which is generated by meaning representation sampling and utterance synthesizing;

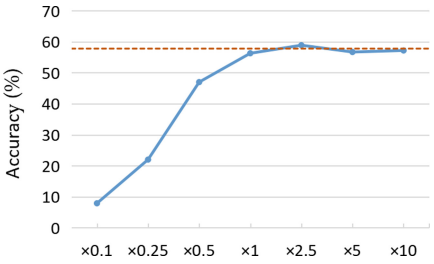

Fig. 4. Test accuracies on GEO with different size of synthesized data. The number of sampled meaning representations has increased from 0.1 times the amount of original data to 10 times. The dash line shows the accuracy of Golden MRs

2) **Mature Corpus** – the corpus is improved by iterative data-model refining;
3) **Supervised** – the model is trained using the original training corpus with the same settings.

For Overnight, we further compare with the Cross-domain Zero Shot [18] which is trained on other source domains and then generalized to new domains and GEN-OVERNIGHT [40] in which all the canonical utterances are also generated without manual annotation. With the nonparallel data: Two-stage [7] employs the cycle learning framework. WmdSamples [7] labels each input sentences with the most possible outputs in the unparallel corpus and deals with these faked samples in a supervised way. Our Mature Corpus + Samples method follows WmdSamples, using the parser built on the refined data to label each input.

The results are shown in Table 2 and Table 3. We can see that:

1) **Our learning framework is promising for resolving the training data bottleneck problem of semantic parsing.** In all datasets, our method outperforms other baselines in the same unsupervised settings. On OVERNIGHT, our method also surpasses the previous approaches in unsupervised data settings. These results verify that data synthesis and iterative data-model refinement is a promising method for semantic parsing without annotated logical forms.
2) **The iterative data-model refining is effective to bootstrap semantic parsers.** Compared with Naive Corpus, after corpus refinement our Mature Corpus gains 27.9 accuracy improvement in ATIS. This verifies the effectiveness of the data-model refining. We believe it results from: i) denoising language model can improve the quality of generated utterances and knowledge-constrained decoding can filter out invalid meaning representations; ii) the bootstrapping can leverage the duality between data and model for iterative refining.

Detailed Analysis

Effects of Utterance Denoising and Constrained Decoding. Table 2 and 3 show the accuracies by removing denoising language model (–Denoising) and by removing

Table 3. Accuracies on GEO and ATIS. The previous methods with superscript $*$ means they use different unsupervised settings. Confidence-driven and Two-stage both use the nonparallel data.

	GEO	ATIS
Supervised		
SEQ2SEQ	88.2	84.2
Dong and Lapata [12]	87.1	84.6
Jia and Liang [21]	89.3	83.3
Susanto and Lu [39]	90.0	-
Xu et al. [45]	88.1	85.9
Chen, Sun, and Han [9]	88.9	85.5
Jie and Lu [22]	89.3	-
Guo et al. [17]	87.1	83.1
Unsupervised		
Confidence-driven*	66.4	-
Two-stage*	63.7	-
Naive Corpus	29.3	25.0
Mature Corpus		
EMBED BERT	**58.2**	**52.9**
GloVe	55.0	52.5
Rand	44.6	43.3
w/o Denoising	45.0	39.5
w/o Constraint	38.9	37.1

Table 4. Evaluation Accuracies on GEO and ATIS with the increase of iterations.

	GEO	ATIS
Iterative updating		
Iter.1	41.4	37.7
Iter.2	49.3	44.6
Iter.3	57.1	48.0
Iter.4	58.9	52.5
Iter.5	58.2	52.9

knowledge constraints during decoding (–Constraint). We can see that: 1) Both utterance denoising and constrained decoding are effective. In average on all three datasets, removing denoising results in 12.0 accuracy drop and removing constrained decoding results in 16.4 accuracy drop. 2) Constrained decoding is more helpful than denoising. We believe this is because the grammar and the knowledge-base can effectively improve the quality of automatically generated parallel data, from which a new lexicon is built and is further used to synthesize new parallel data.

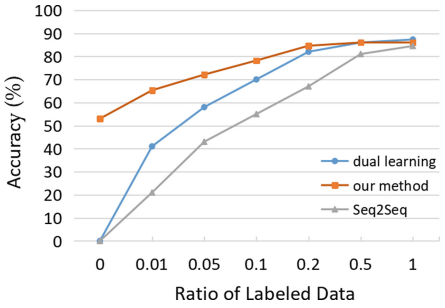

Fig. 5. Test accuracies on ATIS with different amounts of labeled data.

Effects of Word Embeddings. To analyze the effects of word embeddings settings, we compare our method with different settings of word embeddings: BERT – word representations are from the pretrained $\text{BERT}_{\text{LARGE}}$ [11]; GloVe – word embeddings are initialized by GloVe [31]; Rand – the word embeddings are initialized by uniformly sampling within the interval [-0.2, 0.2], and the unseen words are all presented as UNK token. We can see that the pretrained word embeddings can effectively improve the model. We believe this is because it empowers the model with better representation and helps the model generalize to similar words.

Effect of Data Synthesis. To analyze the effectiveness of synthesized data, we: 1) compare our models with Golden MRs – in which all utterances are synthesized from the manually labeled meaning representations in original corpus; 2) increase the amount of sampled meaning representations from $\times 0.1$ to $\times 10$ size of the original labeled data. The results on GEO are shown on Fig. 4.

We can see that: 1) the graph sampling algorithm can effectively sample meaning representations – compared with Golden-MRs, our method can achieve nearly the same performance with $\times 1$ dataset. 2) The data synthesis is useful, when the size of data increases from $\times 0.1$ to $\times 1$, the performance gradually increases. We also noticed that when the data size exceeds the original data, the performance of the model does not improve much. We believe that this is because too much data generated with a certain amount of noise can no longer provide useful supervision information.

Effect of Iterative Bootstrapping. Table 4 shows the accuracies by increasing the number of iterations. We can see that: 1) the iterative data-model refining is effective: when we conduct more refining iterations, the performance gradually increases and stabilizes at a reasonable level – from 41.4 accuracy in Iter 1 to 58.9 in Iter 4 in GEO; 2) The bootstrapping process can reach its equilibrium within few iterations: for GEO in 5 iterations and for ATIS in 4 iterations.

Semi-supervised Learning. To investigate the effectiveness of our method given some additional labeled instances, we vary the amount of labeled data from 0 to all labeled data. Our model can use the labeled data to train semantic parser and induce lexicon in each iteration. Seq2Seq can only use the labeled data. Dual learning [6] forms a closed

loop to learn unlabeled data in reinforcement learning. In Fig. 5, We can see that our model enhances semantic parsing over most settings. Especially, our model has obvious advantages when there is a small amount of labeled data.

5 Related Work

Neural Semantic Parsers. In recent years, neural semantic parsers have achieved significant progress. Neural parsers model semantic parsing as a sentence to logical form translation task [20–22,44], And many constrained decoding algorithms are also proposed [9,20,25,27];

Data Scarcity in Semantic Parsing. Witnessed the labeled data bottleneck problem, many techniques have been proposed to reduce the demand for labeled logical forms. Many weakly supervised learning are proposed [1,2,4,35], such as denotation-base learning [14,30], iterative searching [10]. Semi-supervised semantic parsing is also proposed, such as variational auto-encoding [48], dual learning [6], dual information maximization [46], and back-translation [38]. Constrained language models are also proposed to resolve few-shot semantic parsing [36,43].

Unsupervised Semantic Parsers. There are also some unsupervised semantic parsers, such as USP [32] proposes the first unsupervised semantic parse, and GUSP [33] builds semantic parser by annotating the dependency-tree nodes and edges. Wang et al. [15] select high confidence pairs for unsupervised learning. Two-stage [7] train unsupervised paraphrasing model with non-parallel data for semantic parsing.

6 Conclusions

We propose a data synthesis and iterative data-model refining algorithm for neural semantic parsing, which can build semantic parsers without labeled data. In our method, the naive corpus is generated from scratch by grammar-based method and knowledge base schemas, and the corpus is improved on bootstrapping to refine model and data with the knowledge of language models and knowledge bases constraints. Experimental results show our approach can achieve promising performance in unsupervised settings.

References

1. Agrawal, P., Dalmia, A., Jain, P., Bansal, A., Mittal, A.R., Sankaranarayanan, K.: Unified semantic parsing with weak supervision. In: ACL (2019)
2. Artzi, Y., Zettlemoyer, L.: Weakly supervised learning of semantic parsers for mapping instructions to actions. TACL **1**, 49–62 (2013)
3. Berant, J., Liang, P.: Semantic parsing via paraphrasing. In: ACL (2014)
4. Berant, J., Chou, A., Frostig, R., Liang, P.: Semantic parsing on freebase from question-answer pairs. In: EMNLP (2013)
5. Bogin, B., Berant, J., Gardner, M.: Representing schema structure with graph neural networks for text-to-SQL parsing. In: ACL (2019)

6. Cao, R., Zhu, S., Liu, C., Li, J., Yu, K.: Semantic parsing with dual learning. In: ACL (2019)
7. Cao, R., et al.: Unsupervised dual paraphrasing for two-stage semantic parsing. In: ACL (2020)
8. Chang, S., Liu, P., Tang, Y., Huang, J., He, X., Zhou, B.: Zero-shot text-to-SQL learning with auxiliary task. In: AAAI (2020)
9. Chen, B., Sun, L., Han, X.: Sequence-to-action, end-to-end semantic graph generation for semantic parsing. In: ACL (2018)
10. Dasigi, P., Gardner, M., Murty, S., Zettlemoyer, L., Hovy, E.H.: Iterative search for weakly supervised semantic parsing. In: NAACL-HLT (2019)
11. Devlin, J., Chang, M.-W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: NAACL-HLT (2019)
12. Dong, L., Lapata, M.: Language to logical form with neural attention. In: ACL (2016)
13. Dong, L., Lapata, M.: Coarse-to-fine decoding for neural semantic parsing. In: ACL (2018)
14. Goldman, O., Latcinnik, V., Nave, E., Globerson, A., Berant, J.: Weakly supervised semantic parsing with abstract examples. In: ACL (2018)
15. Goldwasser, D., Reichart, R., Clarke, J., Roth, D.: Confidence driven unsupervised semantic parsing. In: ACL (2011)
16. Guo, J., et al.: Towards complex text-to-SQL in cross-domain database with intermediate representation. In: ACL (2019)
17. Guo, J., et al.: Benchmarking meaning representations in neural semantic parsing. In: EMNLP (2020)
18. Herzig, J., Berant, J.: Decoupling structure and lexicon for zero-shot semantic parsing. In: EMNLP (2018)
19. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**, 1735–1780 (1997)
20. Iyyer, M., Yih, W., Chang, M.-W.: Search-based neural structured learning for sequential question answering. In: ACL (2017)
21. Jia, R., Liang, P.: Data recombination for neural semantic parsing. In: ACL (2016)
22. Jie, Z., Lu, W.: Dependency-based hybrid trees for semantic parsing. In: EMNLP (2018)
23. Kate, R.J., Wong, Y.W., Mooney, R.J.: Learning to transform natural to formal languages. In: IAAI (2005)
24. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: ICLR (2015)
25. Krishnamurthy, J., Dasigi, P., Gardner, M.: Neural semantic parsing with type constraints for semi-structured tables. In: EMNLP (2017)
26. Lample, G., Ott, M., Conneau, A., Denoyer, L., Ranzato, M.: Phrase-based & neural unsupervised machine translation. In: EMNLP (2018)
27. Liang, C., Berant, J., Le, Q., Forbus, K.D., Lao, N.: Neural symbolic machines: learning semantic parsers on freebase with weak supervision. In: ACL (2017)
28. Lu, W., Ng, H.T., Lee, W.S., Zettlemoyer, L.S.: A generative model for parsing natural language to meaning representations. In: EMNLP (2008)
29. Misra, D., Chang, M.-W., He, X., Yih, W.: Policy shaping and generalized update equations for semantic parsing from denotations. In: EMNLP (2018)
30. Pasupat, P., Liang, P.: Inferring logical forms from denotations. In: ACL (2016)
31. Pennington, J., Socher, R., Manning, C.D.: Glove: global vectors for word representation. In: EMNLP (2014)
32. Poon, H., Domingos, P.M.: Unsupervised semantic parsing. In: EMNLP (2009)
33. Poon, H.: Grounded unsupervised semantic parsing. In: ACL (2013)
34. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language models are unsupervised multitask learners (2019)
35. Reddy, S., Lapata, M., Steedman, M.: Large-scale semantic parsing without question-answer pairs. Trans. Assoc. Comput. Linguist. **2**, 377–392 (2014)

36. Shin, R., et al.: Constrained language models yield few-shot semantic parsers. In: Moens, M.-F., Huang, X., Specia, L., Yih, S.W. (eds.) Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event/Punta Cana, Dominican Republic, 7–11 November 2021, pp. 7699–7715. Association for Computational Linguistics (2021)
37. Su, Y., Yan, X.: Cross-domain semantic parsing via paraphrasing. In: EMNLP (2017)
38. Sun, Y., et al.: Neural semantic parsing in low-resource settings with back-translation and meta-learning. CoRR (2019)
39. Susanto, R.H., Lu, W.: Semantic parsing with neural hybrid trees. In: AAAI (2017)
40. Wang, Y., Berant, J., Liang, P.: Building a semantic parser overnight. In: ACL (2015)
41. Wong, Y.W., Mooney, R.J.: Learning for semantic parsing with statistical machine translation. In: NACL (2006)
42. Wong, Y.W., Mooney, R.J.: Learning synchronous grammars for semantic parsing with lambda calculus. In: ACL (2007)
43. Wu, S., et al.: From paraphrasing to semantic parsing: unsupervised semantic parsing via synchronous semantic decoding. In: Zong, C., Xia, F., Li, W., Navigli, R. (eds.) Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, 1–6 August 2021, pp. 5110–5121. Association for Computational Linguistics (2021)
44. Xiao, C., Dymetman, M., Gardent, C.: Sequence-based structured prediction for semantic parsing. In: ACL (2016)
45. Xu, K., Wu, L., Wang, Z., Yu, M., Chen, L., Sheinin, V.: Exploiting rich syntactic information for semantic parsing with graph-to-sequence model. In: EMNLP (2018)
46. Ye, H., Li, W., Wang, L.: Jointly learning semantic parser and natural language generator via dual information maximization. In: ACL (2019)
47. Yin, P., Neubig, G.: A syntactic neural model for general-purpose code generation. In: ACL (2017)
48. Yin, P., Zhou, C., He, J., Neubig, G.: StructVAE: tree-structured latent variable models for semi-supervised semantic parsing. In: ACL (2018)
49. Zelle, J.M., Mooney, R.J.: Learning to parse database queries using inductive logic programming. In: AAAI (1996)
50. Zettlemoyer, L.S., Collins, M.: Learning to map sentences to logical form: structured classification with probabilistic categorial grammars. In: UAI (2005)
51. Zettlemoyer, L.S., Collins, M.: Online learning of relaxed CCG grammars for parsing to logical form. In: EMNLP-CoNLL (2007)