EUSP: An Easy-to-Use Semantic Parsing PlatForm

Bo An, Bo Chen, Xianpei Han, Le Sun
State Key Laboratory of Computer Science
Institute of Software, Chinese Academy of Sciences, Beijing, China
{anbo, chenbo, xianpei, sunle}@iscas.ac.cn

Abstract

Semantic parsing aims to map natural language utterances into structured meaning representations. We present a modular platform, EUSP (Easy-to-Use Semantic Parsing PlatForm), that facilitates developers to build semantic parser from scratch. Instead of requiring a large amount of training data or complex grammar knowledge, in our platform developers can build grammar-based semantic parser or neural-based semantic parser through configure files which specify the modules and components that compose semantic parsing system. A high quality grammar-based semantic parsing system only requires domain lexicons rather than costly training data for a semantic parser. Furthermore, we provide a browser-based method to generate the semantic parsing system to minimize the difficulty of development. Experimental results show that the neural-based semantic parser system achieves competitive performance on semantic parsing task, and grammar-based semantic parsers significantly improve the performance of a business search engine.

1 Introduction

Artificially intelligent applications have been emerging in various forms, such as intelligent retrieval, personal assistants, intelligent customer service robot, etc. Most of the existing intelligent applications are capable of understanding user natural language utterance and returning accurate information. One of the core components of these systems is the semantic parser, which maps natural language utterances into formal meaning representations that facilitate the computer to process. Therefore, it is critically desirable to design an easy-to-use platform that facilitates developers to quickly build a high quality semantic parsing system for various domains and applications.

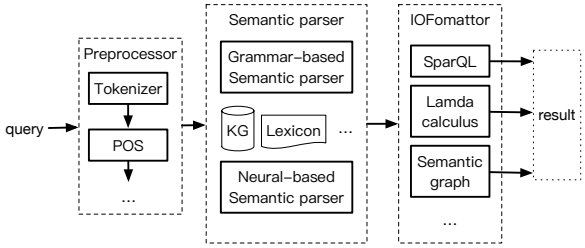

Figure 1: The framework of EUSP platform.

There are mainly two lines of work for semantic parsing: grammar-based semantic parsing and neural-based semantic parsing. Grammar-based semantic parsers employ a set of grammars and lexicons to generate meaning representations for a given utterance. The grammar is a set of expert defined rules to compose the semantic units into candidate meaning representations, which is based on the principle of compositionality (Pelletier, 1994). However, to implement grammar-based semantic parsing system the developers have to understand the complex grammar. What's worse, these parsers require an amount of training dataset that is hard to annotate and only work in a specific domain. Neural semantic parsers convert the utterance directly to meaning representations, like lambda-calculus (Dong and Lapata, 2016) and semantic graph (Chen et al., 2018). One of the major advantages of neural semantic parsing is that the model is trained in an end-to-end way without requiring the developers to understand the complex theory. Unfortunately, neural semantic parser requires a large amount of training data to achieve competitive performance. Thus, it is significantly and crucially desirable to develop a platform for helping developers build semantic parsing systems without requiring complex grammar or costly training data.

To address the above challenges, we present an easy-to-use semantic parsing platform (EUSP), which aims to help developers to build a seman-

Proceedings of the 2019 EMNLP and the 9th IJCNLP (System Demonstrations), pages 67–72
Hong Kong, China, November 3 – 7, 2019.

tic parser from scratch quickly. EUSP provides two kinds of complementary models: grammar-based semantic parser and neural-based semantic parser. The grammar-based semantic parser achieves competitive performance without training data, while neural-based semantic parsing is more generalizable. The advantages of our platform are trifold:

- Flexibility: it provides two kinds of semantic parsing methods (grammar-based and neural-based).
- Cold-Start and Continuous Optimizable: the grammar-based parser only needs domain lexicons, and both of grammar-based and neural-based semantic parsers can be optimized with training data.
- Plug and play: the generated semantic parser is an independent module and can be plugged in the original system without much modification. And it could produce various formats of outputs, like lambda-calculus and SparQL (Sirin and Parsia, 2007), etc.

2 Related Work

Semantic parsing can benefit to many intelligent applications, like intelligent retrieval, personal assistant, etc. There exists a range of semantic parsing toolkits, such as SEMPRE [1] (Berant et al., 2013) and RASA_NLU [2]. Unfortunately, most of these toolkits require both linguistic expertise and a large amount of annotated data. CRUISE (Shen et al., 2018) provide an utterance generation system to reduce the human workload of data annotation. However, CRUISE focuses on spoken language understanding. In this paper, we present a platform for building semantic parsing system quickly and easily.

3 EUSP Platform Overview

EUSP [3], a modular platform, consists of various modules, such as tokenizer, syntax parsing, semantic parsing. Data is passed between different modules in Json format. And developers can use configure files to create different semantic parser by customizing different modules.

[1] https://nlp.stanford.edu/software/sempre/
[2] https://github.com/RasaHQ/rasa_nlu
[3] http://39.98.248.207:8000/nluweb

3.1 EUSP Workflow

The framework is shown in Figure 1, EUSP has three main components: preprocessor, semantic parser and QueryIO formatter, we will detailed describe these components in the following sections.

(i) Preprocessor component consists of various modules (tokenizer, name entity recognizer (NER), syntax parser, etc) to generate useful information for each utterance, including tokens, candidate entities, POS and syntax tree.

(ii) Semantic parser component includes two kinds of semantic parsers: grammar-based and neural-based. The grammar-based parser consists of modules of lexicon recognizer, grammar composer, scorer, reranker and trainer. The neural-based semantic parser is implemented based on our Seq2Action model (Chen et al., 2018), which generates the semantic graph directly from the utterance (with entity recognized).

(iii) IOformatter component generates different formats of output based on the result of semantic parser component, such lambda-calculus.

3.2 Preprocessor component

To extract useful information for semantic parsing, EUSP firstly employs Stanford Tokenizer (Manning et al., 2014) to divide an utterance into a sequence of tokens. Then, a string-based entity linking algorithm (Blanco et al., 2015) is utilized to link the tokens with the domain lexicons, which generate the candidate name entities for semantic parsing. Finally, we use Stanford CoreNLP (Manning et al., 2014) to generate the part-of-speech tags (POS) and constituency parse tree for the given sequence of tokens.

3.3 Semantic parser component

The semantic parser component consists of two kinds of semantic parsing methods, we will illustrate them in this section.

3.3.1 Grammar-based Semantic Parser

The framework of grammar-based semantic parser is illustrated in Figure 2, it consists of four modules: lexicon, grammar, scorer and reranker.

Lexicon module consists of lexicon entries which map the tokens to pre-defined types or functions. For example, the word "texas" triggers the function of {"EntityAttribute": "State"} and the word "states" triggers the function of {"EntityType": "State"}. The lexicon is defined as Figure 3, where 'Token' represents the token

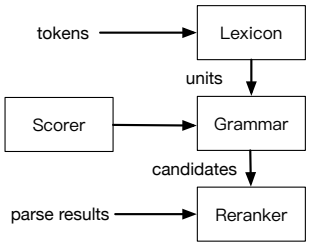

Figure 2: The framework of grammar-based semantic parser component.

which triggers the lexicon entry, and one token could trigger several lexicon entries; 'TokenType' is the type of the entity, includes *EntityClass, Attribute, AttributeValue, Value, Comparator, Combination, Aggregator and NULLQuery*; 'QueryValue' is the triggered value of the token, such as *AttributePredicate, EntityFunction*; 'NormalizedTokenValue' refers normalization of some tokens; 'DataType' refers to the types of values, include *String, Int, Double, Boolean*; 'Score' refers to the score of the lexicon entry.

The domain lexicons are the key resource for building semantic parser, in most case, EUSP only needs coarse-grained domain lexicons, such as the cell lexicon from Sogou [4].

Grammar module decides whether two kinds of tokens can be composed and the result of composition. For example, the grammar rule *EntityFunction* $\wedge$ *AttributePredicate* $\rightarrow$ *AttributeValueFunction* means 'AttributeValueFunction' can be composed by 'EntityFunction' and 'AttributePredicate'. Most of the grammar rules are pre-defined, and developers don't need to modify them. However, if the pre-defined grammar rules contradict with the constraints of KG, the grammar rules should comply with the constraints of knowledge graph (KG). Thus we define two constraints of of the grammar rules: (i) **Attribute constraint**: a 'EntityFunction' and a 'AtttributePredict' can be composed only if the entity type has the specific attribute. (ii) **Value constraint**: the type value of 'AtttributePredict' must be the same as 'AttributeValue'.

Scorer module computes the confidence of each parsed results of the given utterance. Due to the ambiguity of the language, an utterance may produce many different results. To resolve the above issue, we calculate the confidences of the parse results based on the composition features of different grammar rules as Formula (1).

$$score(R) = F(R) \cdot W_{parser} \quad (1)$$

where $F(R)$ is the feature vector of the parse result R; W_{parser} refers to the trainable weight vector of features.

Reranker module further improves the parse results by incorporating the global features of the parse results, includes the size of tokens, the layers of the composition, the coverage of the grammar, etc. And the final confidence of a produced parse result is calculated as Formula (2).

$$score_{Reranker}(R) = F_{global}(R) \cdot W_{rank} \quad (2)$$

where F_{global} refers to the global feature vector of the result and W_{rank} refers to the weight vector of global features, which is also trainable. And the final parse result is the one with maximum $score_{Reranker}(R)$.

3.3.2 Neural-based Semantic Parser

In this paper, we implement the neural-based semantic parser based on Seq2Action (Chen et al., 2018), which directly convert the utterance to semantic graph, and both structural constraints and semantic constraints are applied to ensure the parse result confirms with domain-specific schema. The framework of Seq2Action is illustrated in Figure 4. It is worth mentioning that we implement various neural-based semantic parsing algorithms (such as Seq2Seq, Seq2Tree, Coarse2Fine), developers can specify any one of them through the configure file.

3.4 IOformatter component

To support more natural language applications, EUSP platform implements an IOformatter component, which can generate various kinds of semantic representations for different usage, such as SQL, lambda-calculus, frame semantic, etc.

4 Training Semantic Parser

Grammar-based Semantic Parser

It is worth mentioning that the grammar-based semantic parser in our platform can generate high quality results for many application (such as intelligent retrieval, recommendation) without training data. If the developers have enough training data, the grammar-based semantic parser can be further improved by optimizing the score of lexicon entries, the weight vector W_{parser} and W_{rank}. The

[4] https://pinyin.sogou.com/dict/

```
Token TokenType #Q:QueryValue #N:NormalizedTokenValue #D:DataType #S:Score
```

Figure 3: The format of lexicons.

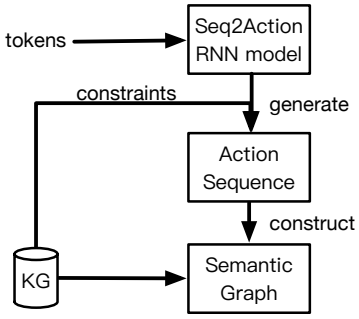

Figure 4: The framework of Seq2Action.

system uses a mini-batch gradient-based discriminant online learning algorithm (Collins, 2002). The training process is as Formula (3).

$$
\begin{aligned}
&g(S) = F(R) - F(T) \\
&g(S_1, ..., S_k) = (g(S_1) + ... + g(S_k))/k \\
&W_{t+1} = W_t - \alpha * g(S_1, ..., S_k)
\end{aligned} \tag{3}
$$

where $F(T)$ is the annotated result of an utterance S; k is the number of instances in a mini-batch; W_t is the parameters at t-th iteration; α is the learning rate and g is the gradient which is calculated based on SGD.

Neural-based Semantic Parser

The parameters of Seq2Action in our model include RNN parameters W^s, W^a, U_w, word embeddings ϕ^x and action embeddings ϕ^y. The parameters are estimated based on training data. Given an utterance X and action sequence Y (the components of semantic graph), we maximize the likelihood of the generated sequence of actions. And the objective function is defined as Formula (4). We employ standard stochastic gradient descent algorithm to update the parameters.

$$
L = \sum_{i=1}^{n} logP(Y_i|X_i) \tag{4}
$$

4.1 Building Semantic Parser

To facilitate the developers to build the semantic parser system for their applications, our platform provides two methods to generate the semantic parsers: the toolkit-based building method and the browser-based building method.

The proposed toolkit consists of all the components and modules needed for building a domain specific semantic parsing system. The developers

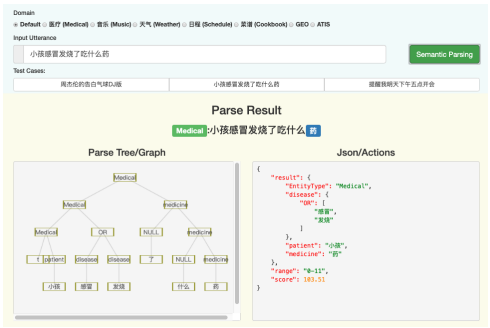

Figure 5: The UI of grammar-based semantic parser.

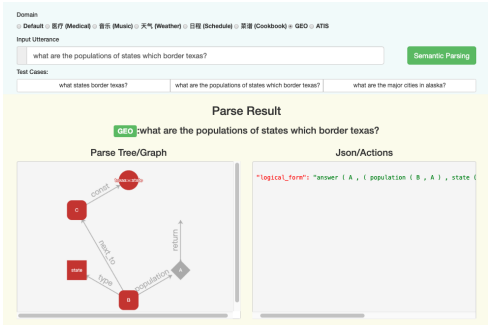

Figure 6: The UI of neural-based semantic parser.

could quickly build a domain semantic parser accord to the instructions of the toolkit with the domain dependent lexicon.

Furthermore, our platform provides a more easy-to-use way to build their semantic parser. We provide a web page that the developers can upload their domain lexicons and specify some key information of the semantic parser (or default values), and our platform will generate a compressed file with the lexicon entries, grammar rules and parsing engine. And developers can deploy the semantic parser with necessary environment, like JDK and Python development environment.

And we offer user friendly interfaces for developers to build and test their semantic parsers like Figure 5 and Figure 6.

5 Experimental Evaluation

In this section, we implement neural semantic parser and grammar semantic parser based on EUSP platform for English and Chinese respectively. We compare our method with several state-of-the-art neural semantic parsers as well as the baselines without leveraging constraints. Furthermore, we deploy our grammar-based semantic parser in a Chinese business search engine to ver-

ify its value.

5.1 Neural-based Semantic Parser Results

We assess the performance of our method and compare it with previous methods. We conduct experiments on two datasets: GEO and ATIS.

GEO contains natural language questions about 494 US geography paired with corresponding Prolog database queries. Following (Zettlemoyer and Collins, 2005), we use the standard 600/280 instance splits for training/test.

ATIS contains natural language questions of a flight database, with each question is annotated with a lambda calculus query. Following (Zettlemoyer and Collins, 2007), we use the standard 4473/448 instance splits for training/test.

We use 200 hidden units and 100 dimensional word vectors for sentence encoding. And we initialize all parameters by uniformly sampling within the interval $[-0.1, 0.1]$. We train our model for a total of 30 epochs with an initial learning rate of 0.1, and halve the learning rate every 5 epochs after epoch 15. We replace word vectors for words occurring only once with a universal word vector. We evaluate different systems using the standard accuracy metric, and the accuracies on different datasets are obtained.

Results

We compare our method with state-of-the-art neural based systems on both datasets. Because all systems using the same train/test splits, we directly use the reported best performances from their original papers for fair comparison.

For our method, we train our model with three settings: the first one is the basic sequence-to-action model without constraints Seq2Act; the second one adds structure constraints in decoding Seq2Act (+C1); the third one is the full model which adds both structure and semantic constraints Seq2Act (+C1+C2). The overall results are shown in Table 1.

From Table 1 we can see that: 1) Our method achieved comparative performances on both datasets. 2) By leveraging knowledge base schema during decoding, semantic constraints are effective for semantic parsing. Compared to Seq2Act and Seq2Act (+C1), the Seq2Act (+C1+C2) achieves the best performance on both datasets. This is because semantic constraints can further filter semantic illegal actions using selectional preference and consistency between types.

Model	GEO	ATIS
Seq2Seq Models		
(Jia and Liang, 2016)	85.0	76.3
(Jia and Liang, 2016)* (+data)	89.3	83.3
(Dong and Lapata, 2016) 2Seq	84.6	84.2
(Dong and Lapata, 2016)2Tree	87.1	84.6
(Dong and Lapata, 2018)	88.2	87.7
(Dong and Lapata, 2018)+oracle sketch	93.9	95.1
Seq2Action Models		
Seq2Act	87.5	84.6
Seq2Act (+C1)	88.2	85.0
Seq2Act (+C1+C2)	88.9	85.5

Table 1: Test accuracies on GEO and ATIS datasets, where * indicates systems with extra resources are used

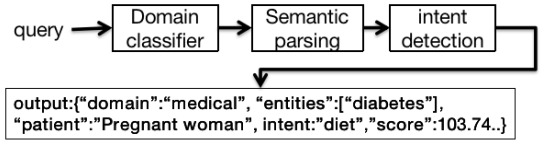

Figure 7: The illustrate of semantic parser for search engine.

5.2 Grammar-based Semantic Parser Results

In this section, we implement grammar-based semantic parser for improving a Chinese intelligent retrieval performance.

Evaluation definition

In this paper, our system parses the user query and outputs useful information for the search engine, such as the entities, user intents and confidences. And the search engine can directly link to the entities and relations in the knowledge graph, instead of just depending on the word features, and present more relevant information or pages to the user.

Evaluation We use the following evaluation metrics: (1) **pageviews coverage (PV)** : the coverage rate of pageviews of one domain; (2) **entity recall (Recall)**: the recall of the entity in the queries; (3) **classification accuracy (Acc)**: the intent classification accuracy of the queries; (4) **DCG**: discounted cumulative gain.

Experimental Settings

Due to the fact that most of the lexicon entries are domain specific and most of user queries and income refer to some top domains, such as medical, entertainment, thus we conduct experiments on three domains: medical, entertainment and novel. And we implement three semantic parsers based on the lexicons of these domains. The overall framework of this task is illustrated in Figure 7.

As showed in Figure 7, there are three components in our system: (i) We first use a domain classifier to filter out the queries that don't belong

the domain, we implement the classifier based on SVM with word tfidf features and word embeddings. (ii) Then, the semantic parser component parses the queries and produces the entities, attributes, confidences, etc. (iii) Finally, the intent detection component identifies the intent of the queries based on the information generated above.

Results

We deploy our system in the pipeline of the business search engine and give the parse results to the search engine instead of plain queries. We compare our system with the baseline search engine without leveraging the parse results. To evaluate the benefit of our system for the search engine, we randomly select 2000 real user queries as input, and manually evaluate the results from the search engine. Table 2 presents our results. Overall, we observe that by incorporating our semantic parsing system all of the metrics of the search engine improved in all of the domains by a large margin. Most importantly, all of three domain semantic parsers are built only based on domain lexicons without training data for semantic parsers.

Domain	PV	Recall	Acc	DCG
Medical	1.5%	22.5%	85%	1.53
Medical + SP	**3.0%**	**39.1%**	**98%**	**1.63**
Entertainment	2.4%	26.1%	87%	1.49
Entertainment + SP	**4.9%**	**40.2%**	**99%**	**1.61**
Novel	1.6%	36.0%	90%	1.58
Novel + SP	**2.3%**	**46.1%**	**99%**	**1.67**

Table 2: The overall results of the search engine, where +SP indicates systems with leveraging results from semantic parser.

6 Conclusion

We have presented an easy-to-use platform for building domain semantic parsers from scratch, without requiring developers to understand the complex theory of semantic parsing. To reduce the requirement of training data for semantic parser, the grammar-based semantic parser can be generated only based on domain dependent lexicons without requiring training data. Although we only validate our model in search engines, our platform is universal and can be easily embedded in applications such as question answering and dialogue.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grants no. 61433015, 61572477 and 61772505.

References

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544.

Roi Blanco, Giuseppe Ottaviano, and Edgar Meij. 2015. Fast and space-efficient entity linking for queries. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, pages 179–188. ACM.

Bo Chen, Le Sun, and Xianpei Han. 2018. Sequence-to-action: End-to-end semantic graph generation for semantic parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*.

Michael Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of Empirical methods in natural language processing*, pages 1–8.

Li Dong and Mirella Lapata. 2016. Language to logical form with neural attention. *arXiv preprint arXiv:1601.01280*.

Li Dong and Mirella Lapata. 2018. Coarse-to-fine decoding for neural semantic parsing. *arXiv preprint arXiv:1805.04793*.

R. Jia and P. Liang. 2016. Data recombination for neural semantic parsing. In *Association for Computational Linguistics (ACL)*.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of the 52th Annual Meeting of the Association for Computational Linguistics: system demonstrations*, pages 55–60.

Francis Jeffry Pelletier. 1994. The principle of semantic compositionality. *Topoi*, 13(1):11–24.

Yilin Shen, Avik Ray, Abhishek Patel, and Hongxia Jin. 2018. Cruise: Cold-start new skill development via iterative utterance generation. In *Proceedings of ACL 2018, System Demonstrations*, pages 105–110.

Evren Sirin and Bijan Parsia. 2007. Sparql-dl: Sparql query for owl-dl. In *OWLED*, volume 258. Citeseer.

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*.

Luke S. Zettlemoyer and Michael Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *Conference on Uncertainty in Artificial Intelligence*.