Semantic Parsing Using Hierarchical Concept Base

Yi Gao, Caifu Hong, and Xihong Wu(✉)

Speech and Hearing Research Center, Key Laboratory of Machine Perception and Intelligence(Ministry of Education), School of Electronics Engineering and Computer Science, Peking University, Beijing, China
{gaoyi,hongcf,wxh}@cis.pku.edu.cn

Abstract. Compositional question answering first maps natural language sentences into meaning representations, then a meaning interpreter is used to evaluate the corresponding answers against a database. A novel approach is proposed in this paper which involves a concept base with rich hierarchical information. A new meaning representation form is introduced correspondingly to match the hierarchical concept base. A set of constructions which encode the correspondence of concept sequences and their meaning representations are used for parsing. The experimental results show that the proposed semantic parser performs favorably in terms of both accuracy and generalization performance compared to existing semantic parsers.

Keywords: Semantic parsing · Concept base · Hierarchical information

1 Introduction

Compositional question answering is important in natural language understanding. Previous works involve semantic parsing which maps a natural language(NL) sentence into its meaning representation(MR). Semantic parsers based on syntax first derive syntactic trees from the NL sentences, then the syntactic trees are converted to the corresponding MRs [1–3]. Semantic parsers based on machine translation technologies use synchronous grammars, which match NL string patterns and construct MRs synchronously [4–7]. Semantic parsers using dependency-based compositional semantics(DCS) derive all the possible MRs from NL sentences, then a probabilistic model is used to find the answers [8,9]. Recent years, semantic parsers using knowledge bases [10–13] such as Freebase [14], or using grounded information [15,16] are developed to handle domain-independent, large-scale corpus.

Overall, two kinds of information are used to improve the generalization performance of a semantic parser. One is the syntactic information. Words or phrases are generalized into syntactic non-terminals, which capture the unseen

Xihong Wu—IEEE Senior member.

X. He et al. (Eds.): IScIDE 2015, Part II, LNCS 9243, pp. 567–575, 2015.
DOI: 10.1007/978-3-319-23862-3_56

phrases in the training corpus. Another is the semantic information such as knowledge bases, which capture the unseen relations in the training corpus. However, traditional methods using syntactic information always tend to overgeneralize the NL words, and the methods using semantic information do not capture the hierarchical relations which would further improve the generalization of semantic parsers.

A novel approach for semantic parsing is proposed in this paper. A concept base is introduced which contains hierarchical relations between concepts. A new form of MR for the semantic parser is proposed to match the concept base. It shares the same structure with the concept base. Compared to the widely used Montague semantics based on lambda calculus, the proposed MR is easily to combine with a concept base. Compared to DCS which constrains the concept relations to several kinds, the proposed MR is free to contain all relations.

To integrate the concept base into the semantic parser, a set of constructions are introduced which map concept sequences into their MRs. The constructions have some resemblances to the rules in synchronous grammars. They both capture the syntactic information in a specific way, which avoids the overgeneralization of NL words. But constructions are based on the concept base. They can also captures the semantic information between concepts.

The experiments in GeoQuery [17], a benchmark dataset, have shown that the proposed system outperforms all existing systems both in accuracy and generalization performance.

2 Concept Base

The basic elements in the concept base are concepts. A string with the first letter capitalized denotes the corresponding concept if no ambiguities exist. The formal definition of the concept base is $K =< E, A, R, E_i, A_i, R_i, R_h >$, in which:

E represents entity, examples include $City$, $State$, $Person$, etc. E_i represents the instances of entities. The extension of an entity is the set of the instances of that entity. For example, $Austin$ is an instance of $City$, so it's an element in the extension of $City$.

A represents attribute, examples include $Height$, $Length$, $Area$, etc. A_i represents the instances of attributes. The extension of an attribute is the set of the instances of that attribute.

R represents relations. A relation relates a set of concepts which are relative elements of that relation. R_i are the instances of relations. The extension of a relation is the set of instances of that relation. For example, a relation $Loc(City, State)$ means a $City$ is located in a $State$, while $Loc(Austin, Texas)$ represents an instance of the relation $Loc(City, State)$, where $City$ and $State$ are instantiated as $Austin$ and $Texas$.

R_h represents the hierarchical relations. They are a kind of relations. If a concept C_1 is the hypernym of another concept C_2, then there exists a hierarchical relation between C_1 and C_2. Examples include the relation between $City$ and $Capital$, the relation between $Attribute$ and $Area$, etc.

The hypernym of a concept C is denoted as C_h^C. The extension of a concept C is denoted as $Ext(C)$. Note that words in NL are also regarded as concepts, they are instances of the entity $Word$.

Since there are no appropriate hierarchical concept bases for GeoQuery currently, a manually annotated one is used to conduct the experiments. Theoretically the hierarchical concept base is domain-independent, and can be used in any other systems.

3 Meaning Representation

3.1 Semantic Tree

The MRs of sentences also consist of concepts. A basic assumption is made that all the MRs are trees. A semantic tree is represented as $t =< Root\ (C_1)\ (C_2)\ \ldots\ (C_m)\ >$, where $Root$ is the root of the tree, and $C_1, C_2, \ldots, C_m$ are the child trees of $Root$. Figure 1 shows some examples. Each concept in the tree is a hyponym of some concept in the concept base, because they have different extensions. In Fig. 1(a), the extension of $State'$ should be the states bordering Texas, while its hypernym, $State$, has all the known states in its extension. This specific form of MR shares the same structure with the concept base, which allows convenient computation of the semantic tree.

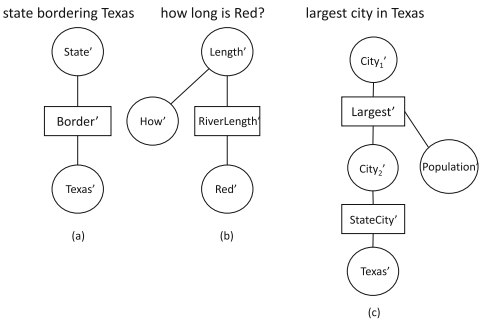

Fig. 1. Some examples of semantic trees in our system

3.2 Computation

The computation of a semantic tree is defined as the procedure of finding the extension of the focus concept in the semantic tree. Normally, the focus concept is the root of the tree. If there exists interrogatives in the tree, such as $When$, $Where$, $What$, How, etc., then focus concept is the concept connected to that interrogative. There are two typical cases:

(1) The focus concept is a relation R, and all the concepts connected to it are its relative elements, denoted as $C_1, C_2, \ldots, C_m$. The instances of the focus

concept should be included in the extension of its hypernym C_h^R. Denote an instance in the extension of R_h^R as R_i. Its relative elements are $C_{1i}, C_{2i}, \ldots, C_{mi}$, which are all instances. If for every $j \in 1, 2, \ldots, m$, C_{ji} is an instance of C_j, then R_i is an instance of R. Formally:

$$\begin{aligned} Ext(R) = \{R_i(C_{1i}, C_{2i}, \ldots, C_{mi}) \in Ext(C_h^R)| \\ C_{1i} \in Ext(C_1) \wedge C_{2i} \in Ext(C_2) \wedge \cdots \wedge C_{mi} \in Ext(C_m)\} \end{aligned} \tag{1}$$

This is called the "match" method. Since not all relation instances are known in advance, several computing methods are needed in GeoQuery:

Count. This computing method is used when counting the number of instances of a concept. Formally:

$$Ext(R) = \{R_i(C_{1i}, C_2) \in Ext(C_h^R)|C_{1i} \in Ext(C_1) \wedge C_{1i} = |Ext(C_2)|\} \tag{2}$$

Quantification. Examples of Quantification include *Total*, *Average*, etc. For example, the computing method of *Average* is:

$$\begin{aligned} Ext(R) = \{R_i(C_{1i}, C_2) \in Ext(C_h^R)|C_{1i} \in Ext(C_1) \wedge \\ C_{1i} = \frac{1}{N} \sum_{j=1}^{|Ext(C_2)|} C_{2j}, C_{2j} \in Ext(C_2)\} \end{aligned} \tag{3}$$

Comparative and Superlative. This computing method is used when comparing entities C_1 and C_2, on a specific attribute A_3. Formally:

$$\begin{aligned} Ext(R) = \{R_i(C_{1i}, C_2, A_3)|C_{1i} \in Ext(C_1), \forall C_{2i} \in Ext(C_2), \\ R_i(C_{1i}, C_{2i}, A_3) \in Ext(C_h^R)\} \end{aligned} \tag{4}$$

Negation. If the relation is negative, this computing method is used. Formally:

$$\begin{aligned} Ext(R) = \{R_i(C_{1i}, C_{2i}, \ldots, C_{mi})|C_{1i} \in Ext(C_1) \wedge C_{2i} \in Ext(C_2) \wedge \\ \cdots \wedge C_{mi} \in Ext(C_m) \wedge R_i(C_{1i}, C_{2i}, \ldots, C_{mi}) \notin Ext(C_h^R)\} \end{aligned} \tag{5}$$

(2) The concepts connected to the focus concept are relations, and each relation takes the focus concept as one of its relative elements. Denote the tree as $t =< C(R_1)(R_2)\ldots(R_m) >$. The extension of C is first set to be the extension of its hypernym C_h^C. Then the extensions of those relations are computed using the computing methods introduced above. For each instance C_i in the extension of C, if C_i is one relative element of some instance of R_1, as well as $R_2, \ldots, R_m$, then C_i is an instance of C. Usually, there exists a computing order. Absolute

relations, such as relations computed using "match", are computed before relative relations, such as ones computed using "comparative and superlative".

$$S = \bigcap_{j=1}^{m} \{x \in Ext(C_h^R) | \exists C_{1i}^j, C_{2i}^j, \ldots C_{mi}^j, R_j(C_{1i}^j, C_{2i}^j, \ldots, x, \ldots, C_{mi}^j) \in Ext(R^j), R^j \textit{ is an absolute relation}\} \tag{6}$$

$$Ext(R) = \bigcap_{j=1}^{m} \{x \in S | \exists C_{1i}^j, C_{2i}^j, \ldots C_{mi}^j, R_j(C_{1i}^j, C_{2i}^j, \ldots, x, \ldots, C_{mi}^j) \in Ext(R^j), R^j \textit{ is a relative relation}\} \tag{7}$$

For semantic trees containing the both cases, the result should be the intersection of the results obtained in them. Using the computing methods above mentioned, a semantic tree can be recursively computed.

4 Construction

Construction Representation. A construction encodes the correspondence of a concept sequence and its MR. Note that words in NL are regarded as concepts, so constructions can also be used to denote the correspondence of words and their MRs. A basic assumption for constructions is that except for the constructions of words, the MR of a construction should have one and only one relation concept, and all the other concepts in the MR are the relative elements of that relation. Formally, $cons =< P; F; T; H; R >$. Here P represents the concept sequence. F represents morphological and semantic features(MSFs), such as number for entities, participle for relations, affirmative or negative for relations, etc. They are used to restrict the concept usage to appropriate syntax and semantic context. They are universal and domain-independent. T is the corresponding semantic tree which consists of the concepts in P, and H is the root of the semantic tree. R represents the only relation in T.

Construction Annotation. The sentences in training corpus are manually annotated into constructions. For a sentence S, the procedure is as follows:

(1)For every word W in S, the corresponding MSF is extracted as F_w. Assume that W corresponds to only one MR in this context, denoted as C. The construction is annotated as: $< W; F_w; < C >; C; \epsilon >$.

(2)For every phrase $P_{nl} =< W_1, W_2, \ldots, W_m >$ in S, first map every word in the phrase to corresponding MR, $P =< C_1, C_2, \ldots, C_m >$, with the extracted MSF sequence as F. Then for every concepts in P, find its highest level of hypernym in the hierarchical concept base. This forms a new concept sequence, denoted as P_h. Annotate the corresponding semantic tree T with head H for P_h. Note that there should be only one relation R in T, otherwise the phrase should be split into smaller phrases. Thus the construction is $cons =< P_h; F; T; H; R >$.

Probabilistic Construction. In general, a word or a phrase may correspond to multiple MRs. These ambiguities are the main motivation for extending constructions into probabilistic constructions. Given a word or a phrase, the probability of a MR derived from this phrase is $p(T, H, R|P, F)$. It can be obtained from the corpus:

$$P(T, H, R|P, F) = \frac{count(< P; F; T; H; R >)}{\sum_{(T,H,R)} count(< P; F; T; H; R >)} \tag{8}$$

5 Semantic Parsing

A construction can be split into two parts, a production with H as its left hand side (LHS) and P as its right hand side (RHS), and a semantic tree T. Denote a random subsequence of the input concept sequence as $P_s = (C_{s1}, C_{s2}, \ldots, C_{sm})$, whose MSF sequence is F_s. For a construction $cons =< C_{c1}, C_{c2}, \ldots, C_{cm}; F_c; T; H; R >$, and for every $j \in 1, 2, \ldots, m$, check the following propositions: (1) C_{sj} is C_{cj}; (2) C_{sj} is one hyponym of C_{cj}; (3) C_{sj} is an instance of C_{cj}. If one of them is true, then replace C_{cj} in T as C_{sj}. This forms a new semantic tree T_s, which is part of the meaning representation of the input concept sequence. Denote the root of T_s as H_s. H_s can be further used to combine with other concepts in the input concept sequence. The parsing task has some resemblance with the probabilistic context-free grammar (PCFG) parsing, and the Earley's context-free parsing algorithm [18] is used. Unlike Earley's algorithm which is operated on words and syntactic non-terminals, here the parser operates on concepts. The probability of a semantic tree is obtained by multiplying the probabilities of all the constructions used in that semantic tree. Finally, the most probable one is selected from the semantic tree set T_{set}:

$$\begin{aligned} T_{best} &= argmax_{T \in T_{set}}(P(T, H, R|S, F) \\ &= argmax_{T \in T_{set}}(\prod_{T_j \in T} P(T_j, H_j, R_j|P_j, F_j))) \end{aligned} \tag{9}$$

6 Experiments and Results

The experiments on GeoQuery are conducted. The dataset contains about 800 facts asserting relational information about U.S. geography, and 880 questions annotated with the corresponding MRs. The average length of a sentence is 7.48 words. The proposed system is compared to: (1)WASP [4], which is based on machine translation techniques; (2) λ-WASP [5], an extension of WASP for handling MRs; (3) SYNSEM [1], which combines syntactic information and semantic information together, here we choose its result based on the gold-standard syntactic parses; (4) L2013 [9], which uses DCS as MRs; (5) W2014 [3], which performs best in current CCG-based parsers.

In the first experiment, the accuracy of the proposed system is tested. The corpus is split as 600 questions for training and 280 questions for testing.

Table 1. The Accuracy on GeoQuery

System	Accuracy(%)
WASP	74.8
λ-WASP	86.6
SYNSEM	88.2
L2013	91.4
W2014	90.4
New System	93.4

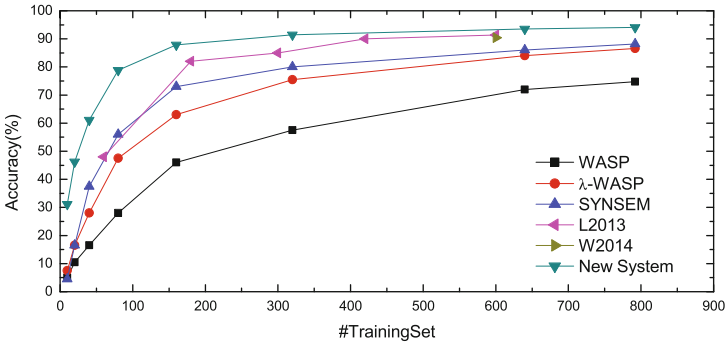

Fig. 2. Learning curves for various parsing algorithms on the GeoQuery corpus

Table 1 shows the results. A few observations can be made: (1)The new system outperforms all existing systems; (2)Though the new system needs more annotation, compared to SYNSEM which uses gold-standard syntactic parses, it still performs better.

The second experiment is about the generalization performance. The standard 10-fold cross validation is used. Figure 2 shows the learning curves of difference systems. It can be observed that, the new system outperforms other systems by wide margins, matching their best final accuracy with only 50 % of the total training examples. This can greatly alleviate the burden of annotation.

7 Conclusion

Hierarchical information can greatly improve the performance of semantic parsers. It includes a concept base which encodes the hierarchical relations between concepts, and a set of constructions which encodes the correspondences of concept sequences and their MRs. By using Earley's algorithm in the semantic parser, the accuracy and generalization performance on the standard semantic parsing dataset, GeoQuery, is clearly improved.

However, the constructions in the system were manually annotated. Learning these constructions automatically will be the future work. This could alleviate the burden of annotation, and also reduce the errors in the annotated corpus.

Acknowledgments. The research was partially supported by the National Basic Research Program of China (973 Program) under grant 2013CB329304, the Major Project of National Social Science Foundation of China under grant 12&ZD119, and the National Natural Science Foundation of China under grant 61121002.

References

1. Ge, R.: Learning for semantic parsing using statistical syntactic parsing techniques. Ph.D. thesis, University of Texas at Austin (2010)
2. Kwiatkowski, T., Zettlemoyer, L., Goldwater, S., Steedman, M.: Lexical generalization in CCG grammar induction for semantic parsing. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp. 1512–1523 (2011)
3. Wang, A., Kwiatkowski, T., Zettlemoyer, L.: Morpho-syntactic lexical generalization for CCG semantic parsing. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP) (2014)
4. Wong, Y. W., and Mooney, R. J.: Learning for semantic parsing with statistical machine translation. In: Proceedings of HLT/NAACL 2006, pp. 439–446, New York City (2006)
5. Wong, Y.W., Mooney R.J.: Learning synchronous grammars for semantic parsing with lambda calculus. In: Association for Computational Linguistics (ACL), pp. 960–967, Prague, Czech Republic (2007)
6. Jones, B. K., Johnson, M., Goldwater, S.: Semantic parsing with bayesian tree transducers. In: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pp. 488–496 (2012)
7. Andreas, J., Vlachos, A., and Clark, S.: Semantic parsing as machine translation. In: Proceedings of the Conference of the Association for Computational Linguistics (ACL), pp. 47–52 (2013)
8. Liang, P., Jordan, M. I., and Klein, D.: Learning Dependency-Based Compositional Semantics. In: Association for Computational Linguistics (ACL), pp. 590–599, Portland (2011)
9. Liang, P., Jordan, M.I., Klein, D.: Learning dependency-based compositional semantics. Comput. Linguist. **39**(2), 389–446 (2013)
10. Krishnamurthy, J. and Mitchell, T.: Weakly Supervised Training of Semantic Parsers. In: Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pp. 754–765 (2012)
11. Cai, Q. and Yates, A.: Semantic parsing freebase: towards open-domain semantic parsing. In: Proceedings of the Joint Conference on Lexical and Computational Semantics (2013)
12. Berant, J., Chou, A., Frostig, R., and Liang, P.: Semantic parsing on freebase from question-answer Pairs. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing(EMNLP), pp. 1533–1544 (2013)

13. Kwiatkowski, T., Choi, E., Artzi, Y., and Zettlemoyer, L.: Scaling Semantic Parsers with On-the-Fly Ontology Matching. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing. Seattle, Washington (2013)
14. Bollacker, K., Evans, C., Paritosh, P., Sturge, T., Taylor, J.: Freebase: a collaboratively created graph database for structuring human knowledge. In: Proceedings of the International Conference on Management of Data (SIGMOD), pp. 1247–1250 (2008)
15. Chen, D., Mooney, R.: Learning to interpret natural language navigation instructions from observations. In: Proceedings of the National Conference on Artificial Intelligence(AAAI), vol. 2, pp. 1–2 (2011)
16. Poon, H.: Grounded unsupervised semantic parsing. Assoc. Comput. Linguist. (ACL) **1**, 933–943 (2013)
17. Zelle, M., Mooney, R.J.: Learning to parse database queries using inductive logic proramming. In: Association for the Advancement of Artificial Intelligence (AAAI). MIT Press, Cambridge (1996)
18. Earley, J.: An efficient context-free parsing algorithm. Commun. Assoc. Comput. Mach. **6**(8), 451–455 (1970)