Scaling Dependency-Based Compositional Semantics to a Wide Domain

Amelia Harrison
Department of Computer Science
University of Texas at Austin
ameliaj@cs.utexas.edu

Subhashini Venugopalan
Department of Computer Science
University of Texas at Austin
vsubhashini@cs.utexas.edu

Abstract

Recent work in semantic parsing for compositional question answering has shown that this task can be successfully approached using question-answer pairs rather than questions annotated with logical forms for training. However, existing datasets for this task are relatively small. It is unclear that any existing approaches to compositional question answering can scale to larger datasets or datasets with wide coverage, even without the burden of producing logical forms for training. In this paper, we explore the possibility of scaling the approach presented in (Liang et al., 2011) to a larger dataset with wider coverage.

1 Introduction

The semantic parsing task addressed in this paper is that of answering compositional natural language questions given a database which is assumed to contain sufficient information to derive an answer. The GEO dataset has historically been used for this task. State-of-the-art systems can answer the compositional questions given in the GEO dataset with fairly high accuracy.

In recent work, Liang et al. present a novel approach to the semantic parsing task. Most prior approaches have focused on training parsing models using natural language utterances annotated with logical forms.[1] In this work, however, the logical forms are treated as latent variables, and a semantic parsing model is trained using natural language questions and their corresponding answers. This sort of "weak supervision" is advantageous because obtaining logical forms for natural language utterances requires expert annotators and is thus quite costly. This work is notable because it takes this weakly supervised approach, and also because it introduces an new formalism called dependency-based compositional semantics (DCS) trees for representing the meaning of a natural language question.

In this paper, we explore the possibility of applying the DCS framework in a wide-domain setting with a much larger underlying database. We present a new wide-domain dataset for compositional question answering called INFOBOX. We identify obvious limitations that prevent the unmodified DCS framework from scaling to this dataset. We then show that those limitations can be lifted, incurring some performance hit, for the GEO task. Finally, we present the results of the applying the modified DCS framework to INFOBOX.

2 A Wide-Domain Dataset for Compositional Question Answering

The dataset we present, INFOBOX, is very similar in spirit to GEO, but covers a much broader domain. It consists of a database and a set of English questions. Each question is annotated with an answer consistent with the facts in the database. The database itself is a subset of the publicly available DBPEDIA.[2] The whole of DBPEDIA is very large, consisting of about 25 million facts. Manipulating so many facts can introduce memory management issues, and so we use a subset of DBPEDIA for our database. We chose facts with information about the 30,000 most commonly occurring entities to serve as the INFOBOX database. This yielded a database containing about 672,000 facts, 775 distinct *relational predicates* and 171 distinct *type*

[1] (Clarke et al., 2010), (Goldwasser et al., 2011), and (Poon and Domingos, 2009) are notable exceptions.

[2] http://wiki.dbpedia.org/Downloads38

predicates. We use the term relational predicates to refer to binary predicates that express a relation between entities. Type predicates are unary predicates that express ontological information about an entity. All of the predicates in the INFOBOX databse are either binary or unary. (For comparison, the GEO database contains about Z facts and Z distinct predicates.) We then manually produced a set of 150 question-answer pairs. Many of the questions are compositional in the sense that they require aggregating information from multiple facts.

Table 1 shows examples of the questions in INFOBOX. It is worth noting, that we originally hypothesized that using the DCS framework would remove the burden of producing annotated logical forms and thereby make producing a large body of questions much easier. When we began to construct INFOBOX, however, we realized that it is essential that the question-answer pairs be consistent with the given database in order to learn to answer questions using DCS. Without directly consulting the database while drafting answers, it is easy to produce answers that are not consistent with the database. One might provide answers that differ in spelling or phrasing, or alternatively provide an answer altogether different from the answer the database would provide. Accordingly, in order to ensure that the answers we provided were correct, we still had to produce a database query corresponding to each question.

3 Background and Limitations of the DCS Framework

There is one obvious limitation in the DCS framework that might hinder attempts to bring this work to scale on a wide-domain dataset. Here, we provide a brief overview of DCS and explain its major limitation. We then explain our approach to addressing this limitation.

3.1 Background

Liang et al. propose a novel logical form, called DCS trees, for representing the meaning of a natural language question. A DCS tree is designed to closely mirror the syntactic dependency parse of a natural language utterance. Nodes in the tree correspond to predicates in the database, and each edge corresponds to one of a fixed set of "relations". Although the structure of a tree reflects that of the dependency parse of the utterance it represents, the semantics of the relations associated with the edges are powerful enough to allow for divergence between syntactic and semantic scope. This makes DCS a more expressive logical language than FUNQL (Kate et al., 2005). Like an SQL or FUNQL query, a DCS tree can be evaluated with respect to a database to produce an answer.

In the DCS framework, each natural language question is associated with a number of permissible DCS trees – that is DCS trees which may correspond to the actual meaning of the utterance, and in turn evaluate to the correct answer. The set of permissible trees is determined, in part, by a fixed set of *lexical triggers*. A lexical trigger associates a word or sequence of words with a predicate in the database. Roughly speaking, the set of permissible trees is then formed by identifying all syntactically correct ways of combining the predicates triggered by the words in an utterance, using the relations available. Clearly, the success of this model depends crucially on the set of lexical triggers. In (Liang et al., 2011), two sets of lexical triggers are evaluated, L and L^+. In L, each predicate has an associated set of POS-tags, and each word in a question is POS-tagged. A given word then triggers all predicates whose associated POS-tags include that assigned to the word. In L^+, many predicates are manually associated with a "prototype" word. The prototype words then trigger only the predicates with which they have been manually associated and do not trigger any predicates based on their POS-tags. Words that are not prototype words still trigger predicates based on their POS-tags. Both L and L^+ also include a small set of general manually specified triggers for function words and a set of "trace predicates". Trace predicates are those that can be inserted without any overt lexical trigger.

Learning in the DCS framework uses a discriminative model which places a probability distribution over all of the permissible DCS trees for a given natural language utterance. The model features are indicators of various relationships between the utterance and the candidate tree. For example, one type of feature indicates that a given

Q: In which city was the author of *T*arzan born?	A: Chicago.
Q: What is the currency of the country with the capital Kampala?	A: Ugandan shilling.
Q: How many instruments does the San Francisco born Metallica band member play?	A: 4.
Q: What state was Upton Sinclair born in?	A: Maryland.

Table 1: Questions from INFOBOX

word triggers a given predicate. The model is trained using an EM-like algorithm which attempts to find the feature weights that maximize the number of questions whose top-weighted DCS tree evaluates to the specified answer.

Evaluating on the GEO and JOBS datasets, the model presented in (Liang et al., 2011) using the lexical trigger set L^+ outperforms all previous systems. Even using the more rudimentary lexical trigger set L, the model achieves recall comparable to the best previous system (that of Kwiatkowski et al. (Kwiatkowski et al., 2010)), which utilizes logical form annotations.

3.2 Limitations

There is one notable aspect of the DCS framework which requires nontrivial supervision: identifying the set of lexical triggers. Even the base set of lexical triggers L, used in (Liang et al., 2011) required annotating the predicates with the POS-tags that should trigger them. This is not such a daunting task for the GEO dataset which includes only about 30 predicates. However, if we try to extend the system to INFOBOX, the time investment required for specifying lexical triggers suddenly becomes quite significant. Using POS-tagged based triggers for INFOBOX is also unrealistic because of the number of distinct predicates in the database. Lexical triggers play the role of reducing the number of possible DCS trees that a question can generate. However, if each word is associated with a very large number of predicates the permissible set of DCS trees will be correspondingly large.

4 Automatically Generating Lexical Triggers

We alleviate the burden of hand-crafting a set of lexical triggers by taking advantage of predicate names and the values associated with predicates.

4.1 Lexical Triggers for GEO

Our approach for the GEO dataset depends entirely on predicate names. Admittedly, the predicates in a database may not be informatively named. Furthermore, this approach eliminates the possibility of a language independent system. We assume that predicates have informative English names. For the GEO dataset, this assumption for the most part holds. We explore a method for automatically generating lexical triggers using WaCkypedia. [3] We construct a list of words most commonly occurring in the same sentence as words in the predicate names. For a given predicate, we then use the words in WaCkypedia with which its constituent words most commonly co-occur, removing stop words.

4.2 Lexical Triggers for INFOBOX

For the INFOBOX dataset we also use lexical triggers extracted from WaCkypedia. However, with this dataset we attempted to make our lexical trigger generation approach not depend on the assumption that predicate names are in English and informative.

Rather than using the words that most commonly co-occur with the words in a predicate name, we use the words that most commonly co-occur with the co-occur with the set of values associated with a given predicate. For example, the relational predicate **mascot** occurs in the facts **mascot**(*Arizona State University*, *Sparky*) and **mascot**(*California Institute of Technology*, *Beaver*), among others. To construct the lexical trigger set for **mascot** we take the words that most commonly co-occur with both *Arizona State University* and *Sparky* or any other value pairs for the predicate **mascot**. Again, we remove stop words. The top results for the **mascot** predicate are "university",

[3] http://wacky.sslmit.unibo.it/doku.php?id=corpora

"school", and "mascot."

This predicate-name-independent approach was possible for the INFOBOX dataset because of the very regular structure of the database. (Recall that the INFOBOX database contains only binary relational predicates and unary type predicates.) Such an approach was not applicable for the GEO database on the other hand because the predicates in that database have widely varying arities.

5 Evaluation and Results

We perform two sets of experiments: one to evaluate the effectiveness of our automatically generated lexical trigger sets for the GEO dataset, and one to test our approach on the INFOBOX dataset.

5.1 Experiments for the GEO dataset

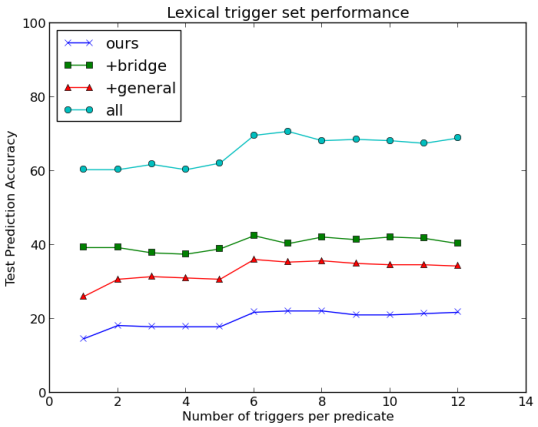

Figure 1: Graph showing the effect of varying the number of lexical triggers per predicate on GEO.

The code that implements (Liang et al., 2011) is publicly available.[4] We use this implementation, and replace the trigger sets L and L^+, with our own automatically generated trigger sets. We do experiments with and without the general functional triggers and trace predicates from (Liang et al., 2011), and varying the number of lexical triggers included for each predicate. All of our experiments use the GEO dataset with 880 questions and an 80/20 train/test split.

Figure 5.1 shows the test prediction accuracy (i.e. the percentage of test questions answered correctly) for each trigger set. The line labeled "ours" shows the results using only the automatically generated triggers. The line labeled "+bridge" shows the results using the generated triggers and the trace predicates from (Liang et al., 2011). The line labeled "+general" represents generated triggers and the general functional predicates from (Liang et al., 2011). The line labeled "all" represents results when all three sets are included.

[4] http://cs.stanford.edu/ pliang/papers/

5.2 Experiments for the INFOBOX dataset

We run similar experiments for the INFOBOX dataset. Again we use an 80/20 train/test split, and vary the number of automatically generated lexical triggers used. The test prediction accuracy for this dataset was identically zero. (See Section **??** for discussion.) For this reason, we report the training prediction accuracy as well as the test and training oracle accuracy for this dataset.

The training prediction accuracy refers to the percentage of training questions for which the model produces a correct answer. For any question input the DCS framework uses the lexical triggers to generate a set of DCS trees corresponding to possible logical forms for the given question. The oracle accuracy refers to the percentage of questions for which a DCS tree corresponding to a correct answer was among the trees generated by the model. Figure 5.2 shows the results.

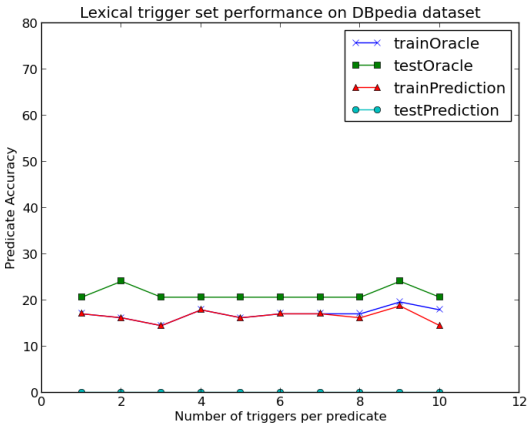

Figure 2: Performance of automatically generated lexical triggers on the wide-domain INFOBOX dataset.

5.3 Discussion

The automatically generated trigger sets perform relatively well on the GEO dataset. When the trace predicates and general functional triggers are included, the test prediction accuracy approaches .70. It is worth noting that including the trace predicates and general functional triggers is reasonable because these are independent of the size of the database. It makes sense then, that we could manually develop a comparable set of trac predicates and functional triggers for the INFOBOX dataset.

Although the results for our automatcally generated trigger sets are substantially lower than the best test accuracies reported in (Liang, 2011) (see Table 2) ,the results do lend some credence to the idea that the DCS framework might successfully scale to a wide domain.

However, the test prediction accuracies for the INFOBOX dataset literally could not be worse. We suspect that the INFOBOX is too small to produce meaningful results. Certainly, as it contains only 150 questions it does not make sense to try and compare these results to the results on the GEO dataset. With more time, we would run the experiments described in the section above usig only 150 of the GEO questions. This would allow us to directly compare the performance of automatically generated lexical triggers across the two datasets, and give a better idea of the effect of scaling the number of facts and predicates in the database on performance.

As Figure 2 shows, in training the oracle and prediction accuracies are almost uniformly identical. This indicates that the model is overfitting the small amount of training data. Although the test oracle accuracy is high, the learned model does not generalize to the test set. Accordingly, the test prediction accuracy results are abysmal.

Lexicon Trigger Set	**Test Prediction Accuracy**
ours	21.78
+bridge	42.5
+general	36.07
all	69.64
L	87.9
L^+	91.4

Table 2: Comparing performance of lexical trigger sets on GEO dataset (600 train and 280 test questions). The first four trigger sets listed each include 6 automatically generated triggers.

6 Related Work

The task of semantic parsing has been studied quite extensively. Early works (Zelle and Mooney, 1996) introduced a statistical approach to learn a semantic parser using inductive logic programming. This led to a line of machine learning approaches (Tang and Mooney, 2001);(Ge and Mooney, 2005);(Kate et al., 2005);(Zettlemoyer and Collins, 2007);(Wong and Mooney, 2006);(Kwiatkowski et al., 2010) that have continually improved the accuracy of semantic parsing. Although the statistical methods are robust and portable, a major drawback of the approach is that it relies crucially on having natural language utterances paired with their logical forms. This not only requires substantial human effort but also expects the annotators to have expertise in some formal language.

In order to alleviate the burden of annotation recent works (Goldwasser et al., 2011);(Clarke et al., 2010);(Poon and Domingos, 2009);(Liang et al., 2011);(Artzi and Zettlemoyer, 2011) are focusing on learning directly from question and answer pairs obviating the need for logical forms.

(Clarke et al., 2010);(Goldwasser et al., 2011) use external world context to supervise semantic interpretation. Similar in vein to (Chen and Mooney, 2008);(Liang et al., 2009), the (Clarke et al., 2010) model learns to map lexical elements to logical forms based on external sources and additionally creates the logical form query relying on the composition rules of the meaning representation language. However, their main contribution that obviates annotated logical forms is the feedback based approach that guides the learning algorithm in identifying hidden alignments between lexical and logical elements.

An important limitation of the (Clarke et al., 2010) system is that although it does require explicit logical form annotations, their model still relies on the framework of the meaning representations language which lacks in its expressive-

ness (in its ability to express) natural language queries. Researchers have, in the past, explored widely with respect to the formal language used for the logical forms: (Giordani and Moschitti, 2010) use SQL, (Zelle and Mooney, 1996);(Tang and Mooney, 2001) use prolog, (Kate et al., 2005) develop a simple functional query language called FUNQL, and (Zettlemoyer and Collins, 2005) use lambda calculus. The construction mechanisms that generate these logical forms in a compositional manner from the utterances have also been quite diverse. The logical forms and the construction mechanisms are in sort of orthogonal problems and hence the expressiveness of these models vary to a good degree.

(Liang et al., 2011) proposes the DCS trees logical form that pushes towards a deeper representation of language. We have discussed the DCS framework previously in [ref the background] As mentioned before, although the DCS framework appears to be a good candidate to tackle wide-domain semantic parsing, the model requires some non-trivial supervision to identify the set of lexical triggers. Our work addresses the task of automatically identifying the lexical triggers in the DCS framework and aims to reduce the burden of supervision further.

Apart from generating the lexical triggers automatically from the database, our work is amongst the first (to the best of our knowledge) to explore the idea of semantic parsing on a wide-domain dataset. The system BOXER developed by JB is a wide-coverage semantic parsing system. However, his system is incomparable to our – due to the lack of evaluation

7 Future Work

The most obvious avenue for future work is to extend the INFOBOX dataset to include many more questions. The dataset is currently too small to yield informative results. One approach to extending the dataset might be to use a grammar to generate questions-answer pairs. The drawback to this approach is that many of the questions generated would have very similar structure. For this reason, it would be preferable to have humans generate the questions. To that end, crowd-sourcing methods like Mechanical Turk might be used for question generation. However, a crowd-sourcing approach would be difficult because the answers for each question must be consistent with the underlying database, as previously noted.

Another direction for future work is to examine adding general functional lexical triggers and trace predicates for the INFOBOX dataset. Our experiments for INFOBOX include neither and experiments on GEO showed that these greatly improve test accuracy.

8 Conclusion

In this paper, we present methods for automatic generation of lexical trigger sets for the DCS framework. We present the results showing that automatically generated trigger sets can achieve reasonable test accuracy on the GEO dataset. We also provide a new (albeit small) dataset for evaluating compositional question answering in a wide-domain.

Acknowledgments

Thanks to Karl Pichotta and Ray Mooney for helpful discussions and healthy skepticism.

References

Yoav Artzi and Luke S. Zettlemoyer. 2011. Bootstrapping semantic parsers from conversations. In *EMNLP*, pages 421–432.

David L. Chen and Raymond J. Mooney. 2008. Learning to sportscast: a test of grounded language acquisition. In *ICML*, pages 128–135.

James Clarke, Dan Goldwasser, Ming-Wei Chang, and Dan Roth. 2010. Driving semantic parsing from the world's response. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, CoNLL '10, pages 18–27, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ruifang Ge and Raymond J. Mooney. 2005. A statistical semantic parser that integrates syntax and semantics. In *Proceedings of the Ninth Conference on Computational Natural Language Learning*, CONLL '05, pages 9–16, Stroudsburg, PA, USA. Association for Computational Linguistics.

Alessandra Giordani and Alessandro Moschitti. 2010. Semantic mapping between natural language questions and sql queries via syntactic pairing. In *Proceedings of the 14th international conference on Applications of Natural Language to Information Systems*, NLDB'09, pages 207–221, Berlin, Heidelberg. Springer-Verlag.

Dan Goldwasser, Roi Reichart, James Clarke, and Dan Roth. 2011. Confidence driven unsupervised semantic parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 1486–1495, Stroudsburg, PA, USA. Association for Computational Linguistics.

Rohit J. Kate, Yuk Wah Wong, and Raymond J. Mooney. 2005. Learning to transform natural to formal languages. In *Proceedings of the Twentieth National Conference on Artificial Intelligence (AAAI-05)*, pages 1062–1068, Pittsburgh, PA, July.

Tom Kwiatkowski, Luke Zettlemoyer, Sharon Goldwater, and Mark Steedman. 2010. Inducing probabilistic ccg grammars from logical form with higher-order unification. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 1223–1233, Stroudsburg, PA, USA. Association for Computational Linguistics.

Percy Liang, Michael I. Jordan, and Dan Klein. 2009. Learning semantic correspondences with less supervision. In *ACL/IJCNLP*, pages 91–99.

P. Liang, M. I. Jordan, and D. Klein. 2011. Learning dependency-based compositional semantics. In *Association for Computational Linguistics (ACL)*, pages 590–599.

Percy Shuo Liang. 2011. *Learning Dependency-Based Compositional Semantics*. Ph.D. thesis, University of California, Berkeley, Fall.

Hoifung Poon and Pedro Domingos. 2009. Unsupervised semantic parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, EMNLP '09, pages 1–10, Stroudsburg, PA, USA. Association for Computational Linguistics.

Lappoon R. Tang and Raymond J. Mooney. 2001. Using multiple clause constructors in inductive logic programming for semantic parsing. In *Proceedings of the 12th European Conference on Machine Learning*, EMCL '01, pages 466–477, London, UK, UK. Springer-Verlag.

Yuk Wah Wong and Raymond J. Mooney. 2006. Learning for semantic parsing with statistical machine translation. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, HLT-NAACL '06, pages 439–446, Stroudsburg, PA, USA. Association for Computational Linguistics.

John M. Zelle and Raymond J. Mooney. 1996. Learning to parse database queries using inductive logic programming. In *AAAI/IAAI, Vol. 2*, pages 1050–1055.

Luke S. Zettlemoyer and Michael Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *UAI 05, Proceedings of the 21st Conference in Uncertainty in Artificial Intelligence, July 26-29 2005, Edinburgh, Scotland*, pages 658–666. AUAI Press.

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