Semantic Parsing Freebase: Towards Open-domain Semantic Parsing

Qingqing Cai
Temple University
Computer and Information Sciences
qingqing.cai@temple.edu

Alexander Yates
Temple University
Computer and Information Sciences
yates@temple.edu

Abstract

Existing semantic parsing research has steadily improved accuracy on a few domains and their corresponding databases. This paper introduces FreeParser, a system that trains on one domain and one set of predicate and constant symbols, and then can parse sentences for any new domain, including sentences that refer to symbols never seen during training. FreeParser uses a domain-independent architecture to automatically identify sentences relevant to each new database symbol, which it uses to supplement its manually-annotated training data from the training domain. In cross-domain experiments involving 23 domains, FreeParser can parse sentences for which it has seen comparable unannotated sentences with an F1 of 0.71.

1 Introduction

Semantic parsing is the task of converting a sentence into a representation of its meaning, usually in a logical form grounded in the symbols of some fixed ontology or relational database (Zelle and Mooney, 1996; Zettlemoyer and Collins, 2005; Kate and Mooney, 2006). A growing body of research on semantic parsing has yielded consistent improvements in parsing accuracy. Yet existing semantic parsers have always been limited by the need for significant amounts of manually-annotated training data for each domain of discourse, or for each new database. As a result, current semantic parsers have been constrained to small domains, like answering geography questions.

In an effort to break out of these narrowly-constrained domains, we investigate semantic parsers for Freebase, an online database of user-contributed facts divided into 86 domains, including everything from architecture to zoos. Freebase is much larger than standard benchmark databases for semantic parsing; for example, it contains 300 times as many relations, and 75,000 times as many instances, as the GeoQuery database. On average, the benchmark GeoQuery dataset has 125 training sentences per relation. An equivalent dataset for Freebase would require labeling close to 40,000 training sentences, an expensive undertaking.

The size and diversity of data in Freebase forces us to consider a new task of *open-domain* semantic parsing. We introduce FreeParser, which trains on labeled examples from a select group of initial domains. It also uses the information in Freebase to automatically find unlabeled training sentences from Wikipedia for every Freebase relation. Using a self-supervised architecture, FreeParser automatically labels these sentences, and then trains a semantic parser for all of Freebase. The current restriction to Wikipedia has a downside: 44% of the test questions in our dataset contained a word that never appeared in our set of automatically-collected sentences, suggesting that significant further gains could be had by scaling to a larger corpus. However, FreeParser is able to find correct parses for 70% of the questions from new domains where it could find relevant sentences in Wikipedia, at a precision of 72%.

The next section provides background on semantic parsing for Freebase, and discusses related work. Section 3 describes the main modules of the FreeParser system. Section 4 analyzes the performance of FreeParser on an open-domain semantic parsing task. Section 5 concludes.

*Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task*, pages 328–338, Atlanta, Georgia, June 13-14, 2013.

domain	num. queries	% of total
film	49	12
business	46	11
tv	34	8
location	32	8
award	32	8
people	30	7
medicine	25	6
organization	24	6
finance	21	5
book	21	5
et al.	89	22
total	403	100

Table 1: **Breakdown of our Freebase data set into domains.** Several questions used symbols from multiple Freebase domains, in which cases human judges selected the best domain they could for that question's category.

Examples

1. What are the neighborhoods in New York City?
   $\lambda x \,.\, \texttt{neighborhoods}(\texttt{new\_york}, x)$
2. How many countries use the rupee?
   $\texttt{count}(x) \,.\, \texttt{countries\_used}(\texttt{rupee}, x)$
3. How many Peabody Award winners are there?
   $\texttt{count}(x) \,.\, \exists y \,.\, \texttt{award\_honor}(y) \wedge$
   $\texttt{award\_winner}(y, x) \wedge$
   $\texttt{award}(y, \texttt{peabody\_award})$

Figure 1: **Example questions with their logical forms.**

2 Background and Previous Work

2.1 Freebase Dataset

Freebase is a free, online, user-contributed, relational database (www.freebase.com) covering many different domains of knowledge. The full schema and contents are available for download.

Freebase has a number of advantages for building an open-domain semantic parser. Most obviously, it provides a much tougher test for semantic parsing than traditional benchmark databases like GeoQuery. It also provides a testbed for semantic parsing across domains. As a reference point, the GeoQuery database contains a single domain (geography), 8 relations, and 698 total instances. The "Freebase Commons" subset of Freebase, which is our focus, consists of 86 domains, an average of 25 relations per domain (total of 2134 relations), and 615,000 known instances per domain (53 million instances total). By dividing Freebase into different sub-databases according to domain, we can readily test the portability of our parser across domains, and its ability to handle relations and symbols that never occur in manually-labeled training data.

Our dataset contains 403 questions and a meaning representation for each question, written in a variant of lambda calculus[1]. We believe the dataset in itself is an important contribution to the field, as it provides a testbed for semantic parsing across multiple domains. Several examples are listed in Fig. 1, and Table 1 provides a breakdown of the domains in our data. The questions were provided by two native English speakers, one high school student and one computer science undergraduate student. Each contributor was introduced to the Freebase website, and asked to come up with English questions that they would like to have answered. No restrictions were placed on the type of questions they should produce, except that they should produce questions for multiple domains. 23 domains are represented in the data set. Inspection of the dataset indicates that most questions have relatively simple and regular syntax, compared with the more complex constructions observed in datasets like GeoQuery. Collecting more complex questions for open-domain tests is an ongoing project, but the existing dataset is already a significant challenge for current semantic parsing learning algorithms.

2.2 Challenges for a Freebase Semantic Parser

To provide a benchmark for comparison, we applied the PCCG-based semantic parser called UBL, developed by Kwiatkowski *et al.* (2010). Source code for UBL is freely available. Its authors found that it achieves results competitive with the state-of-the-art on a variety of standard semantic parsing data sets, including Geo250 English (0.85 F1). Using a fixed CCG grammar and a procedure based on unification in second-order logic, UBL learns a lexicon Λ from the training data which includes entries like:

[1] The data is available from the second author's webpage.

Example Lexical Entries

New York City $\vdash NP : \texttt{new\_york}$
neighborhoods in $\vdash$
$S\backslash NP/NP : \lambda x \lambda y.\texttt{neighborhoods}(x, y)$

Example CCG Grammar Rules

$X/Y : f \quad Y : g \Rightarrow X : f(g)$
$Y : g \quad X\backslash Y : f \Rightarrow X : f(g)$

Using Λ, UBL selects a logical form z for a sentence S by selecting the z with the most likely parse derivations y: $h(S) = \arg\max_z \sum_y p(y, z|x; \theta, \Lambda)$. The probabilistic model is a log-linear model with features for lexical entries used in the parse, as well as indicator features for relation-argument pairs in the logical form, to capture selectional preferences. Inference (parsing) and parameter estimation are driven by standard dynamic programming algorithms (Clark and Curran, 2007; Wilks et al., 1990), using a context-free, combinatory categorial grammar that includes rules for forward application and composition.

In a standard experimental setup on our dataset, UBL provides an F1 of 0.35. We took a random split of 70% of the data for training, 30% for test. An F1 of 0.35 is significantly worse than UBL's performance on GeoQuery data (F1 of 0.85) but within the bounds of reason, given that our data has over 200 relation symbols that need to be learned using less than 300 training sentences, compared with the 8 relations and 250 sentences that make up the Geo250 English dataset.

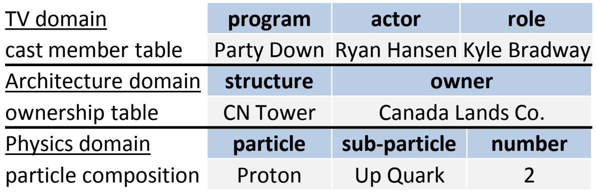
However, UBL is not designed for open-domain semantic parsing, and after training on the training set above, it is not be able to handle questions for any of the remaining 63 domains in Freebase. In open-domain tests, it achieves an F1 of 0.0, and for most sentences, it cannot produce a parse. As one example, we created a test set from the `business` and `finance` domains, and separated the remaining domains for training. Every test example has a predicate symbol that has never been observed before in training. The F1 of 0.0 on this dataset is not a fault of UBL, but rather it shows the difficulty of the task. Porting a system across domains often results in substantial loss of accuracy for many natural language processing tasks (Huang et al., 2011), but usually the drop in accuracy is no more than 10-20%. Open-domain semantic parsing is an even starker challenge; it involves not just new natural language words in the new domains, but also new database symbols, which existing technology cannot handle.

2.3 Previous Work

Krishnamurthy and Mitchell (2012) also create a semantic parser for Freebase, covering 77 of Freebase's over 2000 relations. Like our work, their technique uses distant supervision to drive training over a collection of sentences gathered from the Web, and they do not require any manually-labeled training data. However, their technique does require manual specification of rules that construct CCG lexical entries from dependency parses. In comparison, we fully automate the process of constructing CCG lexical entries for the semantic parser by making it a learning task. We test our results on a dataset of over 400 questions covering over 200 Freebase relations, a more extensive test than the 50 questions used by Krishnamurthy and Mitchell.

Yahya *et al.* (2012) report on a system for translating natural language queries to SPARQL queries over the Yago2 (Hoffart et al., 2013) database. Yago2 consists of information extracted from Wikipedia, WordNet, and other resources using manually-defined extraction patterns. The manual extraction patterns pre-define a link between natural language terms and Yago2 relations. Our techniques automate the process of identifying matches between textual phrases and database relation symbols, in order to scale up to databases with more relations, like Freebase. A more minor difference between Yahya *et al.*'s work and ours is that their system handles SPARQL queries, which do not handle aggregation queries like `argmax` and `count`. We rely on an existing semantic parsing technology to learn the language that will translate into such aggregation queries. On the other hand, their test questions involve more conjunctions and complex semantics than ours. Developing a dataset with more complicated semantics in the queries is part of our ongoing efforts.

Goldwasser *et al.*'s self-supervised, grounded semantic parser (2011) relies on co-training between two different semantic parsing models, one being a simple machine-translation model and the other a more complex structured-prediction model. They achieve an impressive F1 of 0.66 on the benchmark GeoQuery 250 (English) dataset, compared with state-of-the-art supervised models that achieve



TV domain	program	actor	role
cast member table	Party Down	Ryan Hansen	Kyle Bradway
Architecture domain	structure	owner	
ownership table	CN Tower	Canada Lands Co.	
Physics domain	particle	sub-particle	number
particle composition	Proton	Up Quark	2

Figure 2: **Example Freebase relations (tables) and instances for three domains.**

accuracies around 0.85. Unlike semantic parsers for Freebase, Goldwasser *et al.*'s work assumes that a dataset of unlabeled geography questions already exists, for use in unsupervised training. FreeParser answers orthogonal questions: how can we automatically acquire a dataset containing the right keywords and phrases, given only the database itself, and how can we ensure that the acquired sentences are relevant to the relations in the database, without manual supervision? Also, unlike Goldwasser *et al.*'s experiments, FreeParser is tested in a significantly more challenging setting, with far more domains, relations, and entities to be learned.

Many supervised learning frameworks have been applied, including inductive logic programming (Zelle and Mooney, 1996; Thompson and Mooney, 1999; Thompson and Mooney, 2003), support vector machine-based kernel approaches (Kate et al., 2005; Kate and Mooney, 2006; Kate and Mooney, 2007), machine translation-style synchronous grammars (Wong and Mooney, 2007), and context-free grammar-based approaches like probabilistic Combinatory Categorial Grammar (Zettlemoyer and Collins, 2005; Zettlemoyer and Collins, 2007; Zettlemoyer and Collins, 2009; Kwiatkowski et al., 2010; Kwiatkowski et al., 2011; Lu et al., 2008) and discriminative reranking (Ge and Mooney, 2006; Ge and Mooney, 2009). These approaches have yielded steady improvements on standard test sets like GeoQuery, but are difficult to apply to Freebase because of their built-in assumption that relation symbols will be observed during training.

There has been a recent push towards developing techniques which reduce the annotation cost or the data complexity of the models. Models have been developed which can handle some ambiguity in terms of which logical form is the correct label for each training sentence (Chen et al., 2010; Liang et al., 2009). Another set of approaches has investigated the case where no logical forms are provided, but instead some form of feedback or response from the world is used as evidence for what the correct logical form must have been (Clarke et al., 2010; Liang et al., 2011; Artzi and Zettlemoyer, 2011). While such techniques are important, they can only reduce the annotation cost *per domain*, and annotation efforts would still be required for each new domain that contains new database symbols. The goal of the Freebase semantic parser, in contrast, is to port to all domains automatically, without any new manually-labeled data per domain.

3 FreeParser

We introduce FreeParser, an automated system for converting natural language sentences into representations of their meaning, where the relation and constant symbols for the meaning representations are taken from Freebase. FreeParser's modules are described below.

Sentence Retrieval Engine: This module constructs keyword queries for sentences that are likely to express the same relationships as the ones observed in Freebase. It uses an index over a large corpus, currently a snapshot of English Wikipedia, to identify sentences that match the query. Each sentence, along with the Freebase relation r and query q that generated it, is then fed to the Auto-Labeler.

Auto-Labeler: The Auto-Labeler uses knowledge of the relation and query for a sentence to automatically generate a simple logical form for the sentence. The automatically-labeled sentences are then sent to the Assessor.

Assessor: Using a set of domain-independent features, the Assessor filters out sentences that are unsuitable for training the semantic parser. These include sentences that are too long or complex, and sentences where the label from the Auto-Labeler appears to be incorrect. The sentences that pass this filter are added to the training data for FreeParser's semantic parser.

Open-domain Regularizer: FreeParser relies on an existing semantic parser, but with a novel regularizer that helps it learn more appropriate lexical entries for domain-independent function words.

3.1 Sentence Retrieval Engine

The Sentence Retrieval Engine is FreeParser's open-domain technique for retrieving sentences from a

```
**Input**: Freebase relation $r$, unlabeled corpus $C$
**Output**: $Sent$, a set of sentences relevant to $r$
1. Initialize $Sent \leftarrow \emptyset$
2. $E \leftarrow M$ random instances from $r$,
   each projected onto two random attributes
3. $E' \leftarrow N$ pairs $(e_1, e_2) \in E$ with smallest
   $relation\text{-}count(e_1, e_2)$
4. **For each** $(e_1, e_2) \in E'$:
   $S2 \leftarrow \{$sentences in $C$ containing $e_1$ and $e_2\}$
   $S1 \leftarrow \{$sents containing $e_2$, in docs with $e_1\}$
   $Sent \leftarrow Sent \cup S1 \cup S2$
5. **Return** $Sent$
```

Figure 3: **The Sentence Retrieval Engine algorithm**

corpus that are relevant to a particular relation in the database.

Definition *relevance*: We say that a sentence s is relevant to a relation r in Freebase if there exist database symbols $a_1, \dots, a_k$ such that $r(a_1, \dots, a_k)$ appears in Freebase, and $r(a_1, \dots, a_k)$ forms part of the meaning of s, if the meaning were written in a logical form.

For example, for the `cast member` relation in the Freebase sample shown in Figure 2, the sentence

> Hansen also played Kyle Bradway on the Starz show Party Down.

would be relevant, since the sentence expresses a known instance of `cast member`.

Of course, the corpus given as input to the Sentence Retrieval Engine contains only sentences, not the logical forms required to determine relevance according to our definition. FreeParser's strategy is to generate keyword queries that list several named entities that belong to a particular relation. For instance, one query that the Sentence Retrieval Engine might generate for the `cast member` relation is "Ryan Hansen Kyle Bradway," and another might be "Kyle Bradway Party Down."

Figure 3 shows the algorithm for the Sentence Retrieval Engine. In our experiments, we create $M = 1000$ candidate entity pairs, and we select $N = 50$ for queries. We use the open-source Apache Lucene software for constructing an index over the Wikipedia corpus and retrieving relevant sentences.

We have found that selecting good queries is in fact quite tricky, and our experiments in Section 4.4 indicate how badly things can go wrong if it is not done carefully. Two important lessons stand out: First, for reasonable recall, we limit queries to just one or two names. Queries with two names (we call these 2-entity queries) are very often highly relevant, but there are not enough sentences in Wikipedia that match such queries for all relations. We therefore also include queries (which we refer to as 1-entity queries) that first identify Wikipedia articles for one name from a relation, such as articles that mention "Ryan Hansen", and then within this resulting document set, we select sentences that match a second name, such as "Party Down". The resulting sentences therefore always contain one name from the relation, and appear near (within the same document as) a second name. These sentences are noisier than sentences selected with two names, but there are far more matches of such sentences within Wikipedia.

The second lesson for sentence retrieval is that we need to select queries that are not ambiguous. For instance, "James Cameron Avatar" retrieves many sentences for the relation `directed by`. Unfortunately, this same query also produces many sentences for the relations `written by` and `won award for`. The two entities are not enough to unambiguously identify the relationship between them. To combat this problem, FreeParser scores candidate queries according to $relation\text{-}count$, the number of relations in Freebase that hold between the names in the query, and keeps the top-N least ambiguous queries, breaking ties randomly.

3.2 Auto-Labeler

The Auto-Labeler automatically generates a logical form label for every sentence in our data set. It provides a form of "distant supervision" (Bunescu and Mooney, 2007). As an example, if the sentence above were generated from the `cast member` relation using the query "Hansen Bradway", and "Hansen" and "Bradway" are names for the database symbols `Ryan Hansen` and `Kyle Bradway`, then the Auto-Labeler produces $\exists_p$`cast member`(p,`Ryan Hansen`,`Kyle Bradway`) as a label for the sentence. The existentially quantified p variable is necessary to supply enough arguments for the `cast member` relation.

For the general case, let s be a sentence generated from a relation r of arity n via queries involving the entities $\mathbf{e} = (e_1, \dots, e_m)$, and let $\mathbf{a} = (a_1, \dots, a_m)$ be the sequence of attribute indices of r such that

```
Input: auto-labeled sentences $S$ for relation $r$
Output: $S' \subset S$, a high-quality training dataset

1. For each $(s, l) \in S$:
     $C[s, l] \leftarrow complexity\text{-}score(s, l)$
2. Sort $S$ in descending order according to $C$
3. $T \leftarrow$ top 100 examples from $S$
4. $CW \leftarrow critical\text{-}words(T)$
5. $result \leftarrow \emptyset$
6. For each $cw \in CW$:
     $S_{cw} \leftarrow$ top two $(s, l) \in (S - result)$
                 such that $s$ contains $cw$
     $result \leftarrow result \cup S_{cw}$
7. Return $result$
```

Figure 4: **The Assessor algorithm**

e_i is a value for r's attribute a_i. The Auto-Labeler produces the following logical form for s:

$$\exists v_1, \ldots v_n \text{ s.t. } r(v_1, \ldots, v_n) \wedge \\ v_{a_1} = e_1 \wedge \ldots \wedge v_{a_m} = e_m$$

3.3 Assessor

Automatically retrieving training sentences from an unlabeled corpus is a noisy process. In order to improve its precision, FreeParser automatically assesses whether each sentence from the Sentence Retrieval Engine is relevant and useful for training. Its goal is to select, for each relation r, a set of sentences that are all structurally simple; that include a variety of ways of expressing r in English; and that do not include any sentences about other relations r'. The full Assessor algorithm is given in Figure 4.

The Assessor uses two sources of evidence. The first is the complexity of the sentence. After experimenting with numerous features for measuring complexity, we have found that a few types of word counts are the most helpful. Specifically, the most helpful features include: the number of words between two named entities (for two-entity queries), the number of words before the named entity that was part of the query (for one-entity queries), and the total number of non-named-entity, non-stopword words in the sentence. Our implementation uses a list of 200 common stopwords. We trained a maximum-entropy classifier over the complexity features to predict the probability that a sentence is simple enough for training. We manually labeled a small sample of 50 sentences, which were retrieved for relations not found in any of our test sentences. Sentences that truly expressed the relations in the logical form and no other relation were labeled as positive, and all others were labeled negative. The Assessor uses the probability from this classifier to rank all sentences for a relation, and selects the top 100 sentences for further processing.

Complexity statistics alone are not sufficient for selecting good training sentences. For instance, "'Being Spiderman is a dream come true,' says Andrew Garfield" is a short sentence mentioning two entities that participate in the `acted in` relation. However, none of the words in the sentence are particularly indicative of `acted in`, and if FreeParser were to use this as a training sentence, it would most likely learn a wrong lexical entry.

The Assessor additionally weeds out sentences which do not include words strongly associated with a database relation. Previous work has used statistical machine translation models like IBM Model 1 (Brown et al., 1993) as a method for initially determining which words should be associated with which database symbols. After experimenting with this and other models, as implemented in GIZA++ (Och and Ney, 2003), we have found that a simpler procedure is more effective for finding the words which are most indicative of a database relation. Taking the set T of top 100 sentences for r from the complexity ranker, we preprocess the sentences by discarding stopwords and applying stemming. We then count all the remaining word types $v \in V$ appearing in T, and rank them by frequency. We select the top K as word stems that are highly indicative of relation r; we call these word stems the *critical words* for r. For example, for the relation `date founded`, this technique produces the critical words "found," "establish," and "settl," among others. Sentences which do not contain some variant of one of these critical words are unlikely to be good training examples. To obtain a set of diverse but relevant sentences, the Assessor selects at most two sentences for each of the K critical words, taking care not to select any sentence twice. In practice we found that using more than 2 sentences per critical word has no effect on parsing accuracy, but slows the parser training procedure significantly. We tuned K on development data, and set it to $K = 7$.

Example lexical entries for “is” learned by UBL
$S\|NP : \lambda x\ .\ \texttt{religion}(x)$
$S\|NP\|NP : \lambda x \lambda y\ .\ \texttt{person}(x) \wedge$ $\texttt{appearance\_type}(x, \texttt{newscaster})$
$S\|NP\|NP : \lambda x \lambda y\ .\ \texttt{brand}(x, y) \wedge$ $\texttt{company\_brand\_relationship}(x)$

Table 2: **Overly-specific lexical entries for the function word “is,” as learned by a state-of-the-art PCCG semantic parser on our Freebase data set.** All entries shown have significant positive weight in the learned lexicon.

3.4 Initializing the Lexicon for Learning a Semantic Parser

Existing semantic parsing technology requires some initial knowledge in order to learn a full parser. Typically, this knowledge includes lexical entries for named entities and the database symbols to which they correspond, a small number of additional entries for important function words, and a procedure for initializing the weights for learned lexical entries. For instance, UBL uses GIZA++ (Och and Ney, 2003) to initialize the weight of learned lexical entries.

FreeParser includes initial lexical entries for all named entities in our dataset, as well as 29 hand-crafted lexical entries for the words “who,” “what,” “when,” and “where.” These helped to combat the problem of learning a semantic parser from small numbers of questions and large numbers of automatically-retrieved sentences that were almost all declarative statements rather than questions. Following Kwiatkowski *et al.*, these hand-crafted lexical entries are assigned a fixed positive initial weight of 10. We found the following procedure more effective than GIZA++ for initializing the lexicon weights for learned lexical entries in practice: for each critical word and relation pair (v, r) in the sentences from the Assessor, we found a maximum likelihood estimate of $P(v|r)$, the probability of observing the critical word v, given that a sentence expresses the relation r. We then created initial learned lexical entries that pair v and r, with a weight equal to $P(v|r)$.

3.5 New Learning Component for Semantic Parsing: An Open-Domain Regularizer

Training FreeParser’s semantic parsing component on the automatically-labeled sentences, as the system has been described thus far, results in disappointing performance. This is in large part because the UBL semantic parser learns highly domain-specific meanings for function words. Table 2 shows example lexical entries learned for the word “is”. These types of learned meanings are the rule, not the exception, in the existing semantic parser. For single-domain tests, they pose no particular difficulties, even though intuitively they are bad representations of the meaning of a function word. For open-domain semantic parsing, however, it becomes nearly impossible to parse sentences correctly on a new domain, if the only meanings for function words include relations from training domains.

To overcome this problem, we devised a novel regularization technique to encourage the parser to learn domain-independent meanings for function words. Unlike most of FreeParser, this technique is specific to the log-linear CCG semantic parsing technique used by Kwiatkowski *et al.* However, similar mechanisms could potentially be devised for other semantic parsing frameworks. The Kwiatkowski *et al.* model includes a feature function $f_{w,l}$ for every lexical entry mapping a word w to a logical form l. Our novel regularizer $R(\cdot)$ over the parameters θ, which we call an *open-domain regularizer*, penalizes parameters for lexical entries mapping function words to any domain-specific lambda calculus expression. Formally, let F be a set of function words, and P a set of domain-specific predicates from Freebase:

$$R(\theta) = \sum_{w,l} \begin{cases} \theta_{w,l}^2 & \text{if } w \in F \wedge \exists p \in P . p \in l \\ 0 & \text{otherwise.} \end{cases}$$

In our implementation, we added all relations in Freebase that are not part of its `common` domain to P, and collected a set of 282 common English function words for F.

4 Experiments

We now test FreeParser’s ability to provide semantic parses in domains where it has seen no manually-labeled training data. We also empirically analyze the design decisions for FreeParser.

4.1 Experimental Setup

All of our experiments are conducted on the Freebase dataset described in Section 2.1. To create

Q: *What is 'Big Daddy' rated?*
Movie ratings are stored as special codes in Freebase, and are rarely observed 'as is' in text.

Q: *Who is the CEO of Apple?*
Wikipedia regularly uses the full form 'Chief Executive Officer'; no retrieved sentence had 'CEO' together with the executive's name and company name.

Q: *When did Jack Albertson die?*
Many sentences contain "$person$ died on $date$", but no retrieved sentence contained the morphological variant "(did) die."

Table 3: Example infeasible questions, and why FreeParser had difficulty finding sentences in Wikipedia that contain the relevant keywords from the question.

manually-labeled training and test sets for domain adaptation, we divide the dataset into three groups of nearly-equal size by placing similar domains together in the same group. No domain has questions in more than one group. We then perform 3-fold cross-validation across these three groups. We run FreeParser's Sentence Retrieval Engine, Auto-Labeler, and Assessor for all relations that appear in our dataset, and we include the automatically-labeled data in the training data.

4.2 Testing the Sentence Retrieval Engine

179 of the 403 questions (44%) in our dataset included critical words which could not be found using the Sentence Retrieval Engine's queries over Wikipedia. Table 3 lists example infeasible questions. One obvious improvement is to open the retrieval engine to sentences from the Web, for greater recall; this is an important task for future work. For now, this is FreeParser's biggest source of errors. However, note that without this component, the semantic parser parses none of the test data correctly.

4.3 Open-domain semantic parsing tests

We now turn to an experiment that assesses the full FreeParser system on open-domain semantic parsing. For the current experiments, we concentrate on the 224 questions (56% of the full dataset) for which all of the words (except named entities) could be found in at least one of the auto-labeled sentences returned by the Sentence Retrieval Engine. We call these 224 questions the *feasible* questions. For the remaining infeasible questions, FreeParser almost never produces a correct logical form.

Figure 5 shows FreeParser's performance on feasible questions in all test domains, as well as for each of the seven most-common test domains. FreeParser performs quite well, achieving an overall F1 of 0.71, which represents a huge improvement over the F1 of 0.0 for the supervised UBL semantic parser in a domain adaptation setting. An unsupervised parser, which uses only the initial lexical entries from FreeParser and the auto-labeled training data, achieves an F1 of 0.43. Precision and recall differences between FreeParser and these two baselines are statistically significant ($p < 0.01$) using Fisher's exact test. Including both feasible and infeasible questions, FreeParser's F1 is 0.37 because of the low recall on infeasible questions, but as more unlabeled text becomes available to FreeParser, it should have fewer and fewer infeasible questions.

4.4 Testing Critical Design Components

We tested FreeParser with different choices for key parts of the design, to measure their impact. Table 4 presents precision, recall, and F1 scores for four variations of FreeParser, where each variation is missing a critical component of the design. In the first variation, the Sentence Retrieval Engine only issues two-entity queries; it is missing the ability to issue the less-precise single-entity queries. In the second variation, the Assessor uses only the critical words to select sentences for training; it is missing the ability to rank sentences based on their complexity. In the third variation, the Assessor selects the top $2K$, or 14, sentences based on the complexity ranking; it ignores the critical words test. Finally, the last variation shows FreeParser's performance when UBL's training procedure has not been modified with the open-domain regularizer.

Deleting any one of these critical design elements substantially degrades FreeParser's performance, but the 1-entity queries appear to be the most critical design choice, followed by the critical words test and open-domain regularizer. Removing the 1-entity queries surprisingly hurts both precision and recall. The 2-entity queries do tend to retrieve better sentences on average than 1-entity queries, but because they retrieve so few, the Assessor has more difficulty selecting good critical words.

Error analysis showed that incorrect or missing

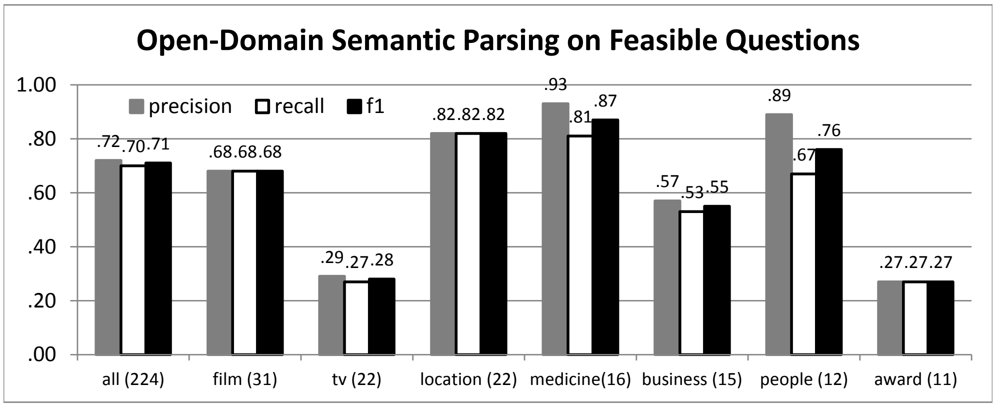


Figure 5: **FreeParser achieves an overall F1 of 0.71, in a test where every correct logical form has some element never seen in manually-labeled training data.** Results across different domains vary, but FreeParser performs well in a variety of domains. Numbers next to domain labels indicate the number of feasible test questions. Results for "all" domains are the micro-average across our three cross-validation folds.

Model	P	R	F1
−1-entity queries	.29	.29	.29
−complexity ranking	.59	.53	.56
−critical words test	.47	.41	.44
−open-domain regul.	.50	.45	.47
FreeParser	**.72**	**.70**	**.71**

Table 4: **FreeParser compared with variations that are missing critical design components.** All precision and recall differences between the full system and its variations are statistically significant ($p < 0.01$) using a two-tailed Fisher's exact test.

lexical entries for critical words were responsible for most (68%) of the 67 incorrect or missing parses for feasible test questions. Many of the incorrect entries mapped critical words like "directed" to related but incorrect predicates, like `written by`. Missing lexical entries were often because the Assessor incorrectly weeded out good auto-labeled examples. The remaining 32% of the errors were mostly due to complex syntax in the questions, or vague questions that require significant reasoning to come up with a valid interpretation.

5 Conclusion and Future Work

Most work on semantic parsing focuses on improving parser accuracy on a small number of relations in a single domain. FreeParser is an exploration of the possibility of automated semantic parsing for arbitrary domains. Among the lessons from our experience in designing FreeParser, these stand out: First, finding training sentences that cover all of the different ways a person may refer to a database element is difficult, and requires carefully constructed retrieval mechanisms for sufficient recall. Second, simple measures of sentence complexity and cooccurrence statistics are effective techniques for identifying good training sentences. And third, standard semantic parsing algorithms require modification for open-domain semantic parsing, to enforce that function words are not mapped to domain-specific logical forms. We report results that help in understanding FreeParser's current strengths and weaknesses, and that also serve as a baseline for future open-domain semantic parsers.

Significant work remains: ideally, a system would be able to incorporate relational data from multiple schemas, and could leverage much larger corpora for learning alignments. Also, FreeParser currently maps English words only to individual Freebase symbols; more sophisticated algorithms and representations are necessary for learning how to map to conjunctions, disjunctions, and more complex combinations of Freebase symbols.

Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant No. IIS-1218692. We wish to thank Sophia Kohlhaas and Ragine Williams for providing data for the project.

References

Yoav Artzi and Luke Zettlemoyer. 2011. Bootstrapping Semantic Parsers from Conversations. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

P. F. Brown, S. D. Pietra, V. J. D. Pietra, and R. L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Razvan Bunescu and Raymond Mooney. 2007. Learning to extract relations from the web using minimal supervision. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL-07)*.

David L. Chen, Joohyun Kim, and Raymond J. Mooney. 2010. Training a Multilingual Sportscaster: Using Perceptual Context to Learn Language. *Journal of Artificial Intelligence Research*, 37:397–435.

Stephen Clark and James R. Curran. 2007. Wide-coverage efficient statistical parsing with ccg and log-linear models. *Computational Linguistics*, 33(4):493–552.

J. Clarke, D. Goldwasser, M. Chang, and D. Roth. 2010. Driving semantic parsing from the world's response. In *Computational Natural Language Learning (CoNLL)*.

Ruifang Ge and Raymond J. Mooney. 2006. Discriminative Reranking for Semantic Parsing. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING/ACL-06)*.

Ruifang Ge and Raymond J. Mooney. 2009. Learning a Compositional Semantic Parser using an Existing Syntactic Parser. In *Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing (ACL-IJCNLP 2009)*.

D. Goldwasser, R. Reichart, J. Clarke, and D. Roth. 2011. Confidence driven unsupervised semantic parsing. In *Association for Computational Linguistics (ACL)*.

Johannes Hoffart, Fabian M. Suchanek, Klaus Berberich, and Gerhard Weikum. 2013. YAGO2: A Spatially and Temporally Enhanced Knowledge Base from Wikipedia. *Artificial Intelligence*, 194:28–61, January.

Fei Huang, Alexander Yates, Arun Ahuja, and Doug Downey. 2011. Language Models as Representations for Weakly Supervised NLP Tasks. In *Conference on Computational Natural Language Learning (CoNLL)*.

Rohit J. Kate and Raymond J. Mooney. 2006. Using String-Kernels for Learning Semantic Parsers. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL*.

Rohit J. Kate and Raymond J. Mooney. 2007. Semi-Supervised Learning for Semantic Parsing using Support Vector Machines. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics, Short Papers (NAACL/HLT-2007)*.

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)*.

Jayant Krishnamurthy and Tom Mitchell. 2012. Weakly Supervised Training of Semantic Parsers. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

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 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Tom Kwiatkowski, Luke Zettlemoyer, Sharon Goldwater, and Mark Steedman. 2011. Lexical Generalization in CCG Grammar Induction for Semantic Parsing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

P. Liang, M. I. Jordan, and D. Klein. 2009. Learning semantic correspondences with less supervision. In *Association for Computational Linguistics and International Joint Conference on Natural Language Processing (ACL-IJCNLP)*.

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

Wei Lu, Hwee Tou Ng, Wee Sun Lee, and Luke S. Zettlemoyer. 2008. A Generative Model for Parsing Natural Language to Meaning Representations. In *Proceedings of The Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

C.A. Thompson and R.J. Mooney. 1999. Automatic construction of semantic lexicons for learning natural language interfaces. In *Proc. 16th National Conference on Artificial Intelligence (AAAI-99)*, pages 487–493.

Cynthia A. Thompson and Raymond J. Mooney. 2003. Acquiring Word-Meaning Mappings for Natural Language Interfaces. *Journal of Artificial Intelligence Research (JAIR)*, 18:1–44.

Y. Wilks, D. Fass, C. Guo, J. MacDonald, T. Plate, and B. Slator. 1990. *Providing Machine Tractable Dictionary Tools*. MIT Press.

Yuk Wah Wong and Raymond J. Mooney. 2007. Learning Synchronous Grammars for Semantic Parsing with Lambda Calculus. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL-2007)*.

Mohamed Yahya, Klaus Berberich, Shady Elbassuoni, Maya Ramanath, Volker Tresp, and Gerhard Weikum. 2012. Natural Language Questions for the Web of Data. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

John M. Zelle and Raymond J. Mooney. 1996. Learning to Parse Database Queries using Inductive Logic Programming. In *AAAI/IAAI*, pages 1050–1055.

Luke S. Zettlemoyer and Michael Collins. 2005. Learning to Map Sentences to Logical Form: Structured Classification with Probabilistic Categorial Grammars. In *Proceedings of the Twenty First Conference on Uncertainty in Artificial Intelligence (UAI)*.

Luke S. 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 (EMNLP-CoNLL)*.

Luke S. Zettlemoyer and Michael Collins. 2009. Learning Context-dependent Mappings from Sentences to Logical Form. In *Proceedings of the Joint Conference of the Association for Computational Linguistics and International Joint Conference on Natural Language Processing (ACL-IJCNLP)*.