A Supervised Semantic Parsing with Lexical Extension and Syntactic Constraint

Zhihua Liao
Foreign Studies College
Hunan Normal University
Changsha, China
liao.zhihua61@gmail.com

Qixian Zeng
Foreign Studies College
Hunan Normal University
Changsha, China
wanyouto@qq.com

Qiyun Wang
Foreign Studies College
Hunan Normal University
Changsha, China
qywang@qq.com

Abstract

Existing semantic parsing research has steadily improved accuracy on a few domains and their corresponding meaning representations. In this paper, we present a novel supervised semantic parsing algorithm, which includes the lexicon extension and the syntactic supervision. This algorithm adopts a large-scale knowledge base from the open-domain Freebase to construct efficient, rich Combinatory Categorial Grammar (CCG) lexicon in order to supplement the inadequacy of its manually-annotated training dataset in the small closed-domain while allows for the syntactic supervision from the dependency-parsed sentences to penalize the ungrammatical semantic parses. Evaluations on both benchmark closed-domain datasets demonstrate that this approach learns highly accurate parser, whose parsing performance benefits greatly from the open-domain CCG lexicon and syntactic constraint.

1 Introduction

Semantic parsers convert natural language sentences to logical forms through a meaning representation language. Recent research has focused on learning such parsers directly from corpora made up of sentences paired with logical meaning representations (Artzi and Zettlemoyer, 2011; Lu et al., 2008; Lu and Tou, 2011; Liao and Zhang, 2013; Kwiatkowski et al., 2010; Kwiatkowski et al., 2011; Kwiatkowski et al., 2012; Zettlemoyer and Collins, 2005; Zettlemoyer and Collins, 2007; Zettlemoyer and Collins, 2009; Zettlemoyer and Collins, 2012). And its goal is to learn a grammar that can map new, unseen sentences onto their corresponding meanings, or logical expressions.

For decades there have been many algorithms that learn probabilistic CCG grammars. These grammars are well suited to the semantic parsing because of the close linking with syntactic and semantic information. Thus, they are used to model a wide range of complex linguistic phenomena and are strongly lexicalized, which store all language-specific grammatical information directly with the words and the CCG lexicon. This CCG lexicon is useful for learning parser. However, it often suffers from the sparsity and the diversity in the training and testing datasets. Consequently, we hold that a large-scale knowledge base should play a key role in the semantic parsing. That is, it might be quite favorable in training such parser and resolving these syntactic ambiguities. Using the knowledge base which contains rich semantic information from the open-domain such as Freebase, can improve efficiently the parser's ability to solve complex syntactic parsing problem and benefit the accuracy. Besides, many previous approaches do not involve the syntactic constraint to penalize the ungrammatical parses when semantic parsing.

This paper presents a supervised approach to learn semantic parsing task using a large-scale open-domain knowledge base and syntactic constraint. The semantic parser is trained to learn parsing via a large-scale open-domain CCG lexicon while simultaneously producing parses that syntactically agree with their dependency parses. Combining these two elements allows us to train a more accurate semantic parser. In particular, it also contains a factored CCG lexicon from the closed-domain GeoQuery and ATIS. Therefore, our approach not only includes two traditional CCG lexicons from the closed-domain GeoQuery and ATIS, and from the open-domain Freebase, but also includes the factored lexicon from the closed-domain GeoQuery and ATIS. This joint of such different lexicons does well in dealing with

Proceedings of Recent Advances in Natural Language Processing, pages 362–370,
Hissar, Bulgaria, Sep 7–9 2015.

the sparsity and the diversity of the dataset where some words or phrases have been never appeared during the training and testing procedures.

This paper is structured as follows. We first provide some background information about Freebase dataset, Combinatory Categorial Grammar, probabilistic CCG (PCCG) and syntactic constraint function in Section 2. Section 3 describes how we use FUBL algorithm to construct a semantic parser FUBLLESC, and Section 4 presents our experiments and reports the results. Section 5 describes the related work. Finally, we make the conclusion and give the future work in Section 6.

2 Background

2.1 Freebase Dataset

Freebase is a free, online, user-contributed, relational database covering many different domains of knowledge (Cai and Yates, 2013; Cai and Yates, 2014; Reddy et al., 2014). The full schema and contents are available for download[1]. One main motivation we adopt Freebase is that it provides a much rich knowledge base to build a large-scale CCG lexicon for semantic parsing than traditional benchmark database like GeoQuery. The GeoQuery database contains only a single geography domain, 7 relations, and 698 total instances. However, the "Freebase Commons" subset of Freebase consists of 86 domains, an average of 25 relations per domain (total of 2134 relations), and 615000 known instances per domain (53 million instances total). The total dataset can be divided into 11 different subsets in terms of the domain types.

2.2 Combinatory Categorial Grammar

CCG is a linguistic formalism that tightly couples syntax and semantic (Steedman, 1996; Steedman, 2000). It can be used to model a wide range of language phenomena. A traditional CCG grammar includes a lexicon Λ with entries like the following:

$flights \vdash N : \lambda x.flight(x)$

$to \vdash (N\backslash N)/NP : \lambda y.\lambda f.\lambda x.f(x) \wedge to(x, y)$

$Boston \vdash NP : bos$

where each lexical item $w \vdash X : h$ has words w, a syntactic category X, and a logical form h. For the first example, these are flights, N, and $\lambda x.flight(x)$. Furthermore, we also introduce the factored lexicon as *(lexeme,template)* pairs, as described in Subsection 3.3.

CCG syntactic categories may be atomic (such as S or NP) or complex (such as $(N\backslash N)/NP$) where the slash combinators encode word order information. CCG uses a small set of combinatory rules to build syntactic parses and semantic representations concurrently. It includes forward ($>$) and backward ($<$) application rules, and forward ($>$**B**) and backward ($<$**B**) composition rules as well as coordination rule. Except for the standard forward and backward slashes of CCG we also include a vertical slash for which the direction of application is underspecified.

2.3 Probabilistic CCG

Due to the ambiguity in both the CCG lexicon and the order in which combinators are applied, there will be many parses for each sentence. We discriminate between competing parses using a log-linear model which has a syntactic constraint function Φ that will be described in the next Subsection 2.4, a feature vector ϕ, and a parameter vector θ. The probability of a parse y that returns logical form $z_i, i = 1 \ldots n$, given a sentence $x_i, i = 1 \ldots n$ and a weak supervision variable μ is defined as:

$$P(y, z_i, \mu | x_i; \theta, \Lambda) = \frac{\Phi(x_i, y, \mu) e^{\theta \cdot \phi(x_i, y, z_i, \mu)}}{\Sigma_{y', z', \mu'} \Phi(x_i, y', \mu') e^{\theta \cdot \phi(x_i, y', z', \mu')}} \quad (1)$$

Subsection 4.3 fully defines the set of features used in the system presented. The most important of these control the generation of lexical items from (lexeme,template)pairs. Each (lexeme, template) pair used in a parse fires three lexical features as we will see in more details in Subsection 4.3.

The parsing or inference problem done at the testing step requires us to find the most likely logical form z given a sentence x_i and a weak supervision variable μ to encourage the agreement between the semantic parses and syntactic-based dependency ones, assuming that the parameters θ and lexicon Λ are known:

$$f(x_i) = \arg\max_z p(z | x_i; \theta, \Lambda) \quad (2)$$

where the probability of the logical form is found by summing over all parses that produce it:

$$p(z | x_i; \theta, \Lambda) = \Sigma_{y \in Yst.\mu=1} p(y, z, \mu | x_i; \theta, \Lambda) \quad (3)$$

[1] http://www.freebase.com

In this approach the distribution over parse trees y is modeled as a hidden variable. Thereby, the parse tree y must agree with a dependency parse of the same sentence x_i. That is, it must guarantee the weak supervision variable μ value to be **1**. For each sentence x_i, we perform a beam search to produce all possible semantic parse y, then check the value of the syntactic constraint function Φ for each generated parse and eliminate parses which are not consistent with their dependency parses. The sum over parses can be calculated efficiently using the inside-outside algorithm with a CKY-style parsing algorithm.

To estimate the parameters themselves, we use stochastic gradient updates. Given a set of n sentence-meaning pairs $(x_i, z_i) : i = 1 \dots n$, we update the parameters θ iteratively, for each example i, by following the local gradient of the conditional log-likelihood objective $O_i = \log P(z_i|x_i; \theta, \Lambda)$. The local gradient of the individual parameter θ_j associated with feature ϕ_j and training instance (x_i, z_i) is given by:

$$\frac{\partial O_i}{\partial \theta_j} = E_{p(y,\mu|x_i,z_i;\theta,\Lambda)}[\phi_j(x_i, y, z_i, \mu)] - E_{p(y,z,\mu|x_i;\theta,\Lambda)}[\phi_j(x_i, y, z, \mu)] \quad (4)$$

All of the expectations in above equation are calculated through the use of the inside-outside algorithm on a pruned parse chart. For a sentence of length m, each parse chart span is pruned using a beam width proportional to $m^{\frac{2}{3}}$, to allow larger beams for shorter sentences.

2.4 Syntactic Constraint Function Φ

A main problem within the above semantic parsing is that it admits a large number of ungrammatical parses. This may result in the waste of time for searching the parse space. Our motivation using the syntactic constraint is that it can shrink the space of searching parse tree and reduce the time of finding the correct parse. Thus, it will enhance the efficiency of semantic parsing. The syntactic constraint function penalizes ungrammatical parses by encouraging the semantic parser to produce parse trees that agree with a dependency parse of the same sentence (Krishnamurthy and Mitchell, 2012; Krishnamurthy and Mitchell, 2013; Krishnamurthy and Mitchell, 2015). Specifically, the syntactic constraint requires the predicate-argument structure of the CCG parse to agree with the predicate-argument structure of the dependency parse.

Therefore, the agreement can be defined as a function of each CCG rule application in y. In the parse tree y, each rule application combines two subtrees, y_h and y_c, into a single tree spanning a larger portion of the sentence x_i. A rule application AGREE(y,t) is consistent with a dependency parse t if the head words of y_h and y_c have a dependency edge between them in t. Here, the weak supervision variable μ is defined as AGREE(y,t). Therefore, the syntactic Constraint function $\Phi(\mu, y, x_i)$ is true if and only if every rule application AGREE(y,t) in y is consistent with t.

$$\Phi(\mu, y, x_i) = \begin{cases} 1 & \text{if } \mu = \text{AGREE}(y, \text{DEPPARSE}(x_i)) \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

3 Learning Factored PCCGs with Lexicon Extension and Syntactic Constraint

Our factored unification based learning method with lexicon extension and syntactic constraint (FUBLLESC) extends the factored unification based learning (FUBL) algorithm (Kwiatkowski et al., 2011) to induce an open-domain lexicon, while also simultaneously adding dependency-based syntactic constraint to permit semantic parsing. In this section, we first define knowledge base K - Freebase and construct the open-domain CCG lexicon Λ_O, then provide the factored lexicon Λ_F from the closed-domain GeoGuery and ATIS, and finally present our FUBLLESC algorithm.

3.1 Knowledge Base K - Freebase

The main input in our system is a propositional knowledge base $K = (E, \Re, C, \Delta)$ (Hoffmann et al., 2011). It contains entities E, categories C, relations $\Re$, and relation instances Δ. The categories and relations are predicates which operate on the entities and return truth values; the categories $c \in C$ are one-place predicates and the relations $r \in \Re$ are two-place predicates. The entity $e \in E$ represents a real-world entity and has a set of known text names. Examples of such knowledge base come from the open-domain Freebase.

This knowledge base influences the semantic parser by two ways. Firstly, CCG logical forms are constructed by combining the categories, relations and entities from the knowledge base with

logical connectives; hence, the predicates in the knowledge base determine the expressivity of the parser's semantic representation. Secondly, the known relation instances $r(e_1, e_2) \in \Delta$ are used to train the semantic parser.

3.2 Construct the Open-domain CCG Lexicon Λ_O

The first step in constructing the semantic parser is to define a open-domain CCG lexicon Λ_O. We construct Λ_O by applying simple dependency-parse-based heuristics to sentences in the training corpus (i.e., NYT-Freebase[2]). Here we adopt MALTPARSER (Nivre et al., 2006) as the dependency-parser. The resulting lexicon Λ_0 captures a variety of linguistic phenomena, including verbs, common nouns, noun compounds and prepositional modifiers. Next, we use the mention identification procedure to identify all mentions of entities in the sentence set $x_i, i = 1 \dots n$. Here we adopt sentential relation extractor MULTIR (Hoffmann et al., 2011), which is a state-of-the-art weakly supervised relation extractor for multi-instance learning with overlapping relation that combines a sentence-level extraction model with a simple, corpus-level component for aggregating the individual facts. This process results in (e_1, e_2, x_i) triple, consisting of sentences with two entity mentions. The dependency path between e_1 and e_2 in x_i is then matched against the dependency parse patterns in Table 1. Each matched pattern adds one or more lexical entries to Λ_O.

Each pattern in Table 1 has a corresponding lexical category template, which is a CCG lexical category containing parameters e, c and r that are chosen at initialization time. Given the triple (e_1, e_2, x_i), the relations r are chosen such that $r(e_1, e_2) \in \Delta$, and the categories c are chosen such that $c(e_1) \in \Delta$ or $c(e_2) \in \Delta$. The template is then instantiated with every combination of these e, c and r values.

3.3 Factored Lexicon Λ_F

A factored lexicon includes a set L of lexemes and a set T of lexical templates (Kwiatkowski et al., 2011). A lexeme $(w, \vec{c})$ pairs a word sequence with an ordered list of logical constants $\vec{c} = [c_1 \dots c_m]$. For example, lexemes can contain a single lexeme (flight, [flight]). It also can contain multiple constants, for example (cheapest, [$\arg\max$,cost]). A lexical template takes a lexeme and produces a lexical items. Templates have the general form $\lambda(\omega, \vec{v}).[\omega \vdash X : h_{\vec{v}}]$, where $h_{\vec{v}}$ is a logical expression that contains variables from the list $\vec{v}$. Applying this template to input lexeme $(w, \vec{c})$ gives the full lexical item $w \vdash X : h$ where the variable ω has been replaced with the wordspan w and the logical form h has been created by replacing each of the variables in $\vec{v}$ with the counterpart constants from $\vec{c}$. Then the lexical items are constructed from the specific lexemes and templates.

[2]http://iesl.cs.umass.edu/riedel/data-univSchema/

3.4 The FUBLLESC Algorithm

Figure 1 shows the FUBLLESC learning algorithm. We assume training data $\{(x_i, z_i) : i = 1 \dots n\}$ where each example is a sentence x_i paired with a logical form z_i. The algorithm induces a factored PCCG with lexicon extension and syntactic constraint, including traditional CCG lexicon Λ_T from the closed-domain GeoQuery and ATIS, the CCG lexicon Λ_O from the open-domain Freebase, the lexeme L, templates T, the factored lexicon Λ_F from the closed-domain GeoQuery and ATIS, and parameter θ.

This algorithm is online, repeatedly performing both lexical expansion (**Step 1**) and parameter update (**Step 2**) procedures for each training example. The overall approach is closely related to the FUBL algorithm (Kwiatkowski et al., 2011), but includes a large-scale CCG lexicon from the open-domain Freebase knowledge base and the syntactic constraint function from the dependency parser.

Inputs: Training set$\{(x_i, z_i) : i = 1 \cdots n\}$ where each example is a sentence x_i paired with a logical form z_i. Set of entity name lexemes L_e. Number of iteration J. Learning rate parameter α_0 and cooling rate parameter c. Set of entity name lexemes L_e. Empty lexeme set L. Empty template set T. Set of NP lexical items l_F from the factored lexicon Λ_F. Set of NP lexical items l_T from the closed-domain CCG lexicon Λ_T. Set of NP lexical items l_O from the open-domain CCG lexicon Λ_O.

Definitions: NEW-LEX(y) returns a set of new lexical items from a parse y. MAX-FAC(l) generates a (lexeme, template) pair from a lexical item $l \in l_F \cup l_T \cup l_O$. PART-FAC($y$) generates a set of templates from parse y. The distributions $p(y, \mu|x, z; \theta, \Lambda_F)$ and $p(y, z, \mu|x; \theta, \Lambda_F)$ are defined by the log-linear model.

Initialization: Let

- For $i = 1 \cdots n$.
  - $(\Psi, \pi) =$ MAX-FAC $(x_i \vdash S : z_i)$
  - $L = L \cup \Psi, T = T \cup \pi$
- set $L = L \cup L_e$.

Part of Speech	Dependency Parse Pattern	Lexical Category Template
Proper Noun	(name of entity e) Sacramento	$w := N : \lambda x.x = e$ $Sacramento := N : \lambda x.x = Sacramento$
Common Noun	$e_1 \overset{SBJ}{\Longrightarrow} [is, are, was, \ldots] \overset{OBJ}{\Longleftarrow} w$ Sacramento is the capital	$w := N : \lambda x.c(x)$ $capital := N : \lambda x.City(x)$
Noun Modifier	$e_1 \overset{NMOD}{\Longleftarrow} e_2$ Sacramento, California	Type change $N : \lambda x.c(x)$ to $N\|N : \lambda f.\lambda x.\exists y.c(x) \wedge f(y) \wedge r(x,y)$ $N : \lambda x.City(x)$ to $N\|N : \lambda f.\lambda x.\exists y.City(x) \wedge f(y) \wedge LocatedIn(x,y)$
Preposition	$e_1 \overset{NMOD}{\Longleftarrow} w \overset{PMOD}{\Longleftarrow} e_2$ Sacremento in California $e_1 \overset{SBJ}{\Longrightarrow} VB^* \overset{ADV}{\Longleftarrow} w \overset{PMOD}{\Longleftarrow} e_2$ Sacramento is located in California	$w := (N\backslash N)/N : \lambda f.\lambda g.\lambda x.\exists y.f(y) \wedge g(x) \wedge r(x,y)$ $in := (N\backslash N)/N : \lambda f.\lambda g.\lambda x.\exists y.f(y) \wedge g(x) \wedge LocatedIn(x,y)$ $w := PP/N : \lambda f.\lambda x.f(x)$ $in := PP/N : \lambda f.\lambda x.f(x)$
Verb	$e_1 \overset{SBJ}{\Longrightarrow} w^* \overset{OBJ}{\Longleftarrow} e_2$ Sacramento governs California $e_1 \overset{SBJ}{\Longrightarrow} w^* \overset{ADV}{\Longleftarrow} [IN, TO] \overset{OBJ}{\Longleftarrow} e_2$ Sacramento is located in California $e_1 \overset{NMOD}{\Longrightarrow} w^* \overset{ADV}{\Longleftarrow} [IN, TO] \overset{OBJ}{\Longleftarrow} e_2$ Sacramento located in California	$w^* := (S\backslash N)/N : \lambda f.\lambda g.\exists x,y.f(y) \wedge g(x) \wedge r(x,y)$ $governs := (S\backslash N)/N : \lambda f.\lambda g.\exists x,y.f(y) \wedge g(x) \wedge LocatedIn(x,y)$ $w^* := (S\backslash N)/PP : \lambda f.\lambda g.\exists x,y.f(y) \wedge g(x) \wedge r(x,y)$ $islocated := (S\backslash N)/PP : \lambda f.\lambda g.\exists x,y.f(y) \wedge g(x) \wedge LocatedIn(x,y)$ $w^* := (S\backslash N)/PP : \lambda f.\lambda g.\lambda y.f(y) \wedge g(x) \wedge r(x,y)$ $located := (S\backslash N)/PP : \lambda f.\lambda g.\lambda y.f(y) \wedge g(x) \wedge LocatedIn(x,y)$
Forms of "to be"	(none)	$w^* := (S\backslash N)/N : \lambda f.\lambda g.\exists x.g(x) \wedge f(x)$

Table 1: Dependency parse pattern used to instantiate lexical categories for the semantic parser lexicon Λ_O. Each pattern is followed by an example phrase that instantiates it. An $*$ indicates a position that may be filled by multiple consecutive words in the sentence. e_1 and e_2 are the entities identified in the sentence, r represents a relation where $r(e_1, e_2)$, and c represents a category where $c(e_1)$. Each template may be instantiated with multiple values for the variables e, c, r.

- set $\Lambda_F = (L, T)$.
- set $\Lambda_F = \Lambda_F \cup \Lambda_T \cup \Lambda_O$.
- Initialize θ using coocurrence statistics.

Algorithm: For $t = 1 \cdots J$, $i = 1 \cdots n$:

Step 1: Add Lexemes and Templates

- Let $y^* = \arg\max_{y,\mu_i} p(y, \mu_i | x_i, z_i; \theta, \Lambda_F)$
- For $l \in$ NEW-LEX(y^*)
  - $(\Psi, \pi) =$ MAX-FAC(l)
  - $L = L \cup \Psi, T = T \cup \pi, \Lambda_F = \Lambda_F \cup (\Psi, \pi)$
- $\Pi =$ PART-FAC $(y^*), T = T \cup \Pi$

Step 2: Update Parameters with Syntactic Constraint

- Let $\gamma = \frac{\alpha_0}{1+c\times k}$ where $k = i + t \times n$.
- Let $\mu_i =$ AGREE$(y,$ DEPPARSE$(x_i))$.
- Let

$$\begin{aligned}\Delta &= E_{p(y,\mu_i|x_i,z_i;\theta,\Lambda_F)}[\phi(x_i, y, z_i, \mu_i)] \\ &- E_{p(y,z,\mu_i|x_i;\theta,\Lambda_F)}[\phi(x_i, y, z, \mu_i)]\end{aligned}$$

- Set $\theta = \theta + \gamma\Delta$

Output: Lexeme L, template T, factored lexicon Λ_F, and parameters θ.

Figure 1: The FUBLLESC algorithm.

Initialization This model is initialized with two traditional CCG lexicons and a factored lexicon as follow. Firstly, a traditional CCG lexicon Λ_T is built from the closed-domain GeoQuery and ATIS whereas another CCG lexicon Λ_O is constructed from the open-domain Freebase. Secondly, we start to build the factored lexicon Λ_F from the closed-domain GeoQuery and ATIS. MAX-FAC is a function that takes a lexical item l and returns the maximal factoring of it, that is the unique, maximal (lexeme,template) pair that can be combined to construct l. We apply MAX-FAC to each of the training examples (x_i, z_i), creating a single way of producing the desired meaning z from a lexeme containing all of the words in x_i. The lexemes and templates created in this way provide the initial factored lexicon Λ_F. Finally, we combine the initial factored lexicon Λ_F with these two traditional CCG lexicons Λ_T and Λ_O to create a new larger factored lexicon Λ_F.

Step 1: The first step of the learning algorithm adds lexemes and templates to the factored model given by performing manipulations on the highest scoring correct parse y^* of the current training example (x_i, z_i). NEW-LEX function generates lexical items by splitting and merging nodes in the best parse tree of each training example. The splitting procedure is a three-step process that first splits the logical form h, then splits the CCG syntactic category X and finally splits the string w. The merging procedure is to recreate the original parse tree $X : h$ spanning w by recombining two new lexical items with CCG combinators (application or composition). First, the NEW-LEX procedure is run on y^* to generate new lexical items. We then use the function MAX-FAC to create the maximal factoring of each of these new lexical items and these are added to the factored representation of the lexicon Λ_F. New templates can also be introduced through partial factoring of in-

ternal parse nodes. These templates are generated by using the function PART-FAC to abstract over the wordspan and a subset of the constants contained in the internal parse nodes of y^*. This step allows for templates that introduce new semantic content to model elliptical language.

Step 2: The second step does a stochastic gradient descent update on the parameter θ used in the parsing model. In particular, this update first computes the weak supervision variable μ_i value for each parse tree y through the syntactic constraint function Φ and then judges whether the punishment need to be done. More details about this update are described in Subsection 2.3.

4 Experimental Setup

This section describes our experimental setup and comparisons of the result. We follow the setup of Zettlemoyer and Collins (2007; 2009) and Kwiatkowski et al. (2010; 2011), including datasets, features, evaluation metrics, and initialization as well as systems, as reviewed below. Finally, we report the experimental results.

4.1 Datasets

We evaluate on two benchmark closed-domain datasets. GeoQuery is made up of natural language queries to a database of geographical information, while ATIS contains natural language queries to a flight booking system (Zettlemoyer and Collins, 2007; Zettlemoyer and Collins, 2009; Zettlemoyer and Collins, 2012; Kwiatkowski et al., 2010; Kwiatkowski et al., 2011). The Geo880 dataset has 880(English sentence, logical form) pairs split into a training set of 600 pairs and a test set of 280 ones. The Geo250 dataset is a subset of the Geo880, and is used 10-fold cross validation experiments with the same splits of this subset. The ATIS dataset contains 5410 (English sentence, logical form) pairs split into a 5000 example development set and a 450 example test set.

4.2 Evaluation Metrics

We report exact math *Recall*, *Precision* and *F1*. *Recall* is the percentage of sentences for which the correct logical form was returned, *Precision* is the percentage of returned logical forms that are correct, and *F1* is the harmonic mean of *Precision* and *Recall*. For ATIS we also report partial match *Recall*, *Precision* and *F1*. Partial match *Recall* is the percentage of correct literals returned. Partial match *Precision* is the percentage of returned literals that are correct.

4.3 Features

We introduce two types of features to discriminate among parses: lexical features and logical-form features. First, for each lexical item $L \in \Lambda_T \cup \Lambda_O$ from the closed-domain CCG lexicon Λ_T and the open-domain CCG lexicon Λ_O, we include a feature ϕ_L that fires when L was used. Second, For each (lexeme, template) pair used to create another lexical item $(l, t) \in \Lambda_F$ about the factored lexicon Λ_F we have indicator features ϕ_l for the lexeme used, ϕ_t for the template used, and $\phi_{l,t}$ for the pair that was used. Thereby, the lexical feature includes ϕ_L and $\phi_{l,t}$. We assign the features on the lexical templates a weight of 0.1 to prevent them from swamping the far less frequent but equally informative lexeme features. For each logical-form feature, it is computed on the lambda-calculus expression z returned at the root of the parse. Each time a predicate p in the output logical expression z takes a argument a with type $T(a)$ in position i, it triggers two binary indicator features: $\phi_{(p,a,i)}$ for the predicate-argument relation and $\phi_{(p,T(a),i)}$ for the predicate argument-type relation.

4.4 Initialization

The weights for lexeme features are initialized according to coocurrance statistics between words and logical constants. They are estimated with the GIZA++ implementation of IBM Model 1 (Och and Ney, 2003; Och and Ney, 2004). The weights of the seed lexical entries from the closed-domain CCG lexicon Λ_T and the open-domain CCG lexicon Λ_O are set to 10 that can be equivalent to the highest possible coocurrence score. The initial weights for templates are set by adding -0.1 for each slash in the syntactic category and -2 if the template contains logical constants. Features on (lexeme, template) pairs and all parse features are initialized to zero. We use the learning rate $\alpha_0 = 1.0$ and cooling rate $c = 10^{-5}$ in all training, and run the algorithm for $J = 20$ iterations.

4.5 Systems

We compare this performance to those recently-published and directly-comparable results. For GeoQuery, they include the ZC07 (Zettlemoyer and Collins, 2007), λ-WASP (Wong and Mooney, 2007), UBL (Kwiatkowski et al., 2010) and

system	Rec.	Pre.	F1
ZC07	74.4	87.3	80.4
UBL	65.6	67.1	66.3
FUBL	81.9	82.1	82.0
FUBLLESC	85.2	**92.8**	88.8

Table 3: Performance of Exact Match on the ATIS development set.

FUBL (Kwiatkowski et al., 2011). For ATIS, we report results from ZC07 (Zettlemoyer and Collins, 2007), UBL (Kwiatkowski et al., 2010) and FUBL (Kwiatkowski et al., 2011).

4.6 Results

Tables 2-4 present all the results on the GeoGuery and ATIS domains. In all cases, FUBLLESC achieves at state-of-the-art recall and precision when compared to directly comparable systems and it significantly outperforms FUBL and ZC07. Most importantly, it is obvious that on precision our FUBLLESC remarkably exceeds other systems because of the joint effect about the addition of an open-domain CCG lexicon and the usage of syntactic constraint. As shown in Table 2, on Geo250 FUBLLESC achieves the highest recall 86.2% and precision 92.0%, whereas on Geo880 the only higher recall and precision (90.8% and 95.6%) are also achieved by FUBLLESC. On the ATIS development set, FUBLLESC outperforms FUBL by 3.3% of recall and by 10.7% of precision, which is shown in Table 3. Table 4 indicates that on the ATIS test set FUBLLESC significantly outperforms FBUL by 10% of precision on Exact Match and 5% of precision on Partial Match, respectively.

5 Related Work

Semantic parsers have been thought of mapping sentences to logical representations of their underlying meanings. There has been significant work on supervised learning for inducing semantic parsers. Various techniques were applied to this problem including machine translation (Wong and Mooney, 2006; Wong and Mooney, 2007), using CCG to building meaning representations (Zettlemoyer and Collins, 2005; Zettlemoyer and Collins, 2007; Zettlemoyer and Collins, 2009; Zettlemoyer and Collins, 2012), higher-order unification (Kwiatkowski et al., 2010; Kwiatkowski et al., 2011), modeling child language acquisition (Kwiatkowski et al., 2012),generative model (Ruifang and Mooney, 2006; Lu et al., 2008), inductive logic programming (Zelle and Mooney, 1996; Thompson and Mooney, 2003; Tang and Mooney, 2000), probabilistic forest to string model for language generation (Lu and Tou, 2011), and the extension from English to Chinese (Liao and Zhang, 2013). The algorithm we develop in this paper builds on some previous work on the supervised learning CCG parsers (Kwiatkowski et al., 2010; Kwiatkowski et al., 2011), as described in Section 3.4.

Recent research in this field has focused on learning for various forms of relatively weak but easily gathered supervision. This includes unannotated text (Poon and Domingos, 2009; Poon and Domingos, 2010), learning from question-answer pairs (Liang et al., 2011; Berant et al., 2013), via paraphrase model (Berant and Liang, 2014), from conversational logs (Artzi and Zettlemoyer, 2011), with distant supervision (Krishnamurthy and Mitchell, 2012; Krishnamurthy and Mitchell, 2013; Krishnamurthy and Mitchell, 2015; Cai and Yates, 2013; Cai and Yates, 2014), and from sentences paired with system behaviors (Artzi and Zettlemoyer, 2013) as well as via semantic graphs (Reddy et al., 2014).

Our approach builds on a number of existing algorithm ideas which include adopting PCCG to building the meaning representation (Kwiatkowski et al., 2010; Kwiatkowski et al., 2011), using the weakly supervised parameter leaning with the syntactic constraint (Krishnamurthy and Mitchell, 2012; Krishnamurthy and Mitchell, 2013), and employing the open-domain Freebase to semantic parsing (Cai and Yates, 2013).

6 Conclusion and Future Work

This paper presents a novel supervised method for semantic parsing which induces PCCG from sentences paired with logical forms. This approach contains an open-domain Freebase lexicon and syntactic constraint which employs dependency parser to penalize uncorrect CCG parsing tree. The experiments on both benchmark datasets (i.e., GeoQuery and ATIS) show that our method achieves higher performances.

In the future work, we are interested in exploring morphological model and containing more open-domain lexicons as well as more syntactic

(a) The Geo250 test set

system	Rec.	Pre.	F1
λ-WASP	75.6	91.8	82.9
UBL	81.8	83.5	82.6
FUBL	83.7	83.7	83.7
FUBLLESC	86.2	**92.0**	89.0

(b) The Geo880 test set

system	Rec.	Pre.	F1
ZC07	86.1	91.6	88.8
UBL	87.9	88.5	88.2
FUBL	88.6	88.6	88.6
FUBLLESC	90.8	**95.6**	93.1

Table 2: Performance of Exact Match between the different GeoQuery test sets.

(a) Exact Match

system	Rec.	Pre.	F1
ZC07	84.6	85.8	85.2
UBL	71.4	72.1	71.7
FUBL	82.8	82.8	82.8
FUBLLESC	86.4	**92.8**	89.5

(b) Partial Match

system	Rec.	Pre.	F1
ZC07	96.7	95.1	95.9
UBL	78.2	98.2	87.1
FUBL	95.2	93.6	94.6
FUBLLESC	97.2	**98.6**	97.9

Table 4: Performance of Exact and Partial Matches on the ATIS test set.

information. Besides, it will also be important to better model some variations within the existing lexemes.

Acknowledgments

We are grateful to the anonymous reviewers for their valuable feedback on an earlier version of this paper. This research was supported in part by Undergraduate Innovative Research Training project of Hunan Normal University (grant no.201401021) and the Social Science Foundation (grant no.14YBA260) in Hunan Province, China.

References

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

Franz Joseph Och and Hermann Ney. 2003. *A Systematic Comparison of Various Statistical Alignment Models*. *Computational Linguistics*, 29(1):19–51.

Franz Joseph Och and Hermann Ney. 2004. *The Alignment Template Approach to Statistical Machine Translation*. *Computational Linguistics*, 30:417–449.

Ruifang Ge and Raymond J. Mooney. 2006. *Discriminative Reranking for Semantic Parsing*. *In Proceedings of the Conference of the Association for Computational Linguistics (ACL)*.

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

Jayant Krishnamurthy and Tom M. Mitchell. 2013. *Joint Syntactic and Semantic Parsing with Combinatory Categorial Grammar*. *In Proceedings of the 52th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Jayant Krishnamurthy and Tom M. Mitchell. 2015. *Learning a Compositional Semantics for Freebase with an Open Predicate Vocabulary*. *Transactions of the Association for Computational Linguistics (TACL)*.

Joakim Nivre, Johan Hall, and Jens Nilsson. 2006. *Maltparser: A Data-driven Parser-denerator for Dependency Parsing*. *In Proceedings of the 21st International Conference on Computatonal Lianguistics and 44th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. *Semantic Parsing on Freebase from Question-Answer Pairs*. *In Proceedings of EMNLP*.

Jonathan Berant and Percy Liang. 2014. *Semantic Parsing via Paraphrasing*. *In Proceedings of ACL*.

John M. Zelle and Raymond J. Mooney. 1996. *Learning to Parse Database Queries using Inductive Logic Programming*. *In Proceedings of the National Conference on Artificial Intelligence (AAAI)*.

Lappoon R. Tang and Raymond J. Mooney. 2000. *Automated Construction of Database Interfaces: Integrating Statistical and Relational Learning for Semantic Parsing*. *In Proceedings of the Joint Conference of Empirical Methods in Natural Language Procesing and Very Large Corpora (EMNLP)*.

Luke S. Zettlemoyer and Michael Collins. 2005. *Learning to Map Sentences to Logical Form: Structured Classification with Probabilistic Categorial Grammars*. *In Proceedings of UAI*, pages 658–666.

Luke S. Zettlemoyer and Michael Collins. 2007. *Online Learning of Relaxed CCG Grammars for Parsing to Logical Form*. *In Proceedings of EMNLP-CoNLL*, pages 678–687.

Luke S. Zettlemoyer and Michael Collins. 2009. *Learning Context-Dependent Mappings from Sentences to Logical Form*. *In Proceedings of ACL-IJCNLP*, pages 976–984.

Luke S. Zettlemoyer and Michael Collins. 2012. *Learning to Map Sentences to Logical Form: Structured Classification with Probabilistic Categorial Grammars*. *CoRR abs*.

Percy Liang, Michael I. Jordan, and Dan Klein. 2011. *Learning Dependency-based Compositional Semantics*. *In Proceedings of the Conference of the Association for Computational Linguistics (ACL)*.

Poon Hoifung and Pedro Domingos. 2009. *Unsupervised Semantic Parsing*. *In Proceedings of the Conference on Empiricial Methods in Natural Language Processing (EMNLP)*.

Poon Hoifung and Pedro Domingos. 2010. *Unsupervised Ontology Induction from Text*. *In Proceedings of the Conference of the Association for Computational Linguistics (ACL)*.

Qingqing Cai and Alexander Yates. 2013. *Semantic Parsing Freebase: Towards Open-domain Semantic Parsing*. *In Proceedings of the Second Joint Conference on Lexical and Computational Semantics (SEM)*.

Qingqing Cai and Alexander Yates. 2014. *Large-scale Semantic Parsing via Schema Matching and Lexicon Extension*. *In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke S. Zettlemoyer, and Daniel S. Weld. 2011. *Knowledge-based Weak Supervision for Information Extraction of Overlapping Relations*. *In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL)*.

Siva Reddy, Mirella Lapata, Mark Steedman. 2014. *Large-scale Semantic Parsing without Question-Answer Pairs*. *Transactions of the Association for Computational Linguistics (TACL)*.

Steedman Mark. 1996. *Surface Structure and Interpretation*. *The MIT Press*.

Steedman Mark. 2000. *The Syntactic Process*. *The MIT Press*.

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)*, Cambridge, MA.

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)*, Edinburgh, UK.

Tom Kwiatkowski, Luke Zettlemoyer, Sharon Goldwater, and Mark Steedman. 2012. *A Probabilistic Model of Syntactic and Semantic Acquisition from Child-Directed Utterances and their Meanings*. *In Proceedings of the European Chapter of the Association for Computational Linguistics (EACL)*, Avignon, France.

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)*, pages 783–792.

Wei Lu and Hwee Tou Ng. 2011. *A Probabilistic Forest-to-String Model for Language Generation from Typed Lambda Calculus Expressions*. *In Proceedings of The Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1611–1622.

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

Yoav Artzi and Luke Zettlemoyer. 2013. *Weakly Supervised Learning of Semantic Parsers for Mapping Instructions to Actions*. *Transactions of the Association for Computational Linguistics (TACL)*.

Yuk Wah Wong and Raymond J. Mooney. 2006. *Learning for Semantic Parsing with Statistical Machine Translation*. *In Proceedings of the Human Language Technology Conference of the North American Association for Computational Linguistics (NAACL)*.

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

Zhihua Liao and Zili Zhang. 2013. *Learning to Map Chinese Sentences to Logical Forms*. *In Proceedings of the 7th International Conference on Knowledge Science, Engineering and Management (KSEM)*, pages 463–472.