Learning Structured Natural Language Representations for Semantic Parsing

Jianpeng Cheng[†] **Siva Reddy**[†] **Vijay Saraswat**[‡] and **Mirella Lapata**[†]
[†]School of Informatics, University of Edinburgh
[‡]IBM T.J. Watson Research
{jianpeng.cheng,siva.reddy}@ed.ac.uk, vsaraswa@us.ibm.com,
mlap@inf.ed.ac.uk

Abstract

We introduce a neural semantic parser which is interpretable and scalable. Our model converts natural language utterances to intermediate, domain-general natural language representations in the form of predicate-argument structures, which are induced with a transition system and subsequently mapped to target domains. The semantic parser is trained end-to-end using annotated logical forms or their denotations. We achieve the state of the art on SPADES and GRAPHQUESTIONS and obtain competitive results on GEOQUERY and WEBQUESTIONS. The induced predicate-argument structures shed light on the types of representations useful for semantic parsing and how these are different from linguistically motivated ones.[1]

1 Introduction

Semantic parsing is the task of mapping natural language utterances to machine interpretable meaning representations. Despite differences in the choice of meaning representation and model structure, most existing work conceptualizes semantic parsing following two main approaches. Under the first approach, an utterance is parsed and grounded to a meaning representation *directly* via learning a task-specific grammar (Zelle and Mooney, 1996; Zettlemoyer and Collins, 2005; Wong and Mooney, 2006; Kwiatkowksi et al., 2010; Liang et al., 2011; Berant et al., 2013; Flanigan et al., 2014; Pasupat and Liang, 2015; Groschwitz et al., 2015). Under the second approach, the utterance is first parsed to an *intermediate* task-independent representation tied to a syntactic parser and then mapped to a grounded representation (Kwiatkowski et al., 2013; Reddy et al., 2016, 2014; Krishnamurthy and Mitchell, 2015; Gardner and Krishnamurthy, 2017). A merit of the two-stage approach is that it creates reusable intermediate interpretations, which potentially enables the handling of unseen words and knowledge transfer across domains (Bender et al., 2015).

The successful application of encoder-decoder models (Bahdanau et al., 2015; Sutskever et al., 2014) to a variety of NLP tasks has provided strong impetus to treat semantic parsing as a sequence transduction problem where an utterance is mapped to a target meaning representation in string format (Dong and Lapata, 2016; Jia and Liang, 2016; Kočiský et al., 2016). Such models still fall under the first approach, however, in contrast to previous work (Zelle and Mooney, 1996; Zettlemoyer and Collins, 2005; Liang et al., 2011) they reduce the need for domain-specific assumptions, grammar learning, and more generally extensive feature engineering. But this modeling flexibility comes at a cost since it is no longer possible to interpret how meaning composition is performed. Such knowledge plays a critical role in understand modeling limitations so as to build better semantic parsers. Moreover, without any task-specific prior knowledge, the learning problem is fairly unconstrained, both in terms of the possible derivations to consider and in terms of the target output which can be ill-formed (e.g., with extra or missing brackets).

In this work, we propose a neural semantic parser that alleviates the aforementioned problems. Our model falls under the second class of approaches where utterances are first mapped to an intermediate representation containing natural language predicates. However, rather than using an external parser (Reddy et al., 2014, 2016) or manually specified CCG grammars (Kwiatkowski et al., 2013), we induce intermediate representations in the form of predicate-argument structures

[1] Our code is available at `https://github.com/cheng6076/scanner`.

from data. This is achieved with a transition-based approach which by design yields recursive semantic structures, avoiding the problem of generating ill-formed meaning representations. Compared to most existing semantic parsers which employ a CKY style bottom-up parsing strategy (Krishnamurthy and Mitchell, 2012; Cai and Yates, 2013; Berant et al., 2013; Berant and Liang, 2014), the transition-based approach we proposed does not require feature decomposition over structures and thereby enables the exploration of rich, non-local features. The output of the transition system is then grounded (e.g., to a knowledge base) with a neural mapping model under the assumption that grounded and ungrounded structures are isomorphic.[2] As a result, we obtain a neural model that jointly learns to parse natural language semantics and induce a lexicon that helps grounding.

The whole network is trained end-to-end on natural language utterances paired with annotated logical forms or their denotations. We conduct experiments on four datasets, including GEOQUERY (which has logical forms; Zelle and Mooney 1996), SPADES (Bisk et al., 2016), WEBQUESTIONS (Berant et al., 2013), and GRAPHQUESTIONS (Su et al., 2016) (which have denotations). Our semantic parser achieves the state of the art on SPADES and GRAPHQUESTIONS, while obtaining competitive results on GEOQUERY and WEBQUESTIONS. A side-product of our modeling framework is that the induced intermediate representations can contribute to rationalizing neural predictions (Lei et al., 2016). Specifically, they can shed light on the kinds of representations (especially predicates) useful for semantic parsing. Evaluation of the induced predicate-argument relations against syntax-based ones reveals that they are interpretable and meaningful compared to heuristic baselines, but they sometimes deviate from linguistic conventions.

2 Preliminaries

Problem Formulation Let $\mathcal{K}$ denote a knowledge base or more generally a reasoning system, and x an utterance paired with a grounded meaning representation G or its denotation y. Our problem is to learn a semantic parser that maps x to G via an intermediate ungrounded representation U. When G is executed against $\mathcal{K}$, it outputs denotation y.

Predicate	Usage	Sub-categories
answer	denotation wrapper	—
type	entity type checking	*stateid*, *cityid*, *riverid*, etc.
all	querying for an entire set of entities	—
aggregation	one-argument meta predicates for sets	*count*, *largest*, *smallest*, etc.
logical connectors	two-argument meta predicates for sets	*intersect*, *union*, *exclude*

Table 1: List of domain-general predicates.

Grounded Meaning Representation We represent grounded meaning representations in FunQL (Kate et al., 2005) amongst many other alternatives such as lambda calculus (Zettlemoyer and Collins, 2005), λ-DCS (Liang, 2013) or graph queries (Holzschuher and Peinl, 2013; Harris et al., 2013). FunQL is a variable-free query language, where each predicate is treated as a function symbol that modifies an argument list. For example, the FunQL representation for the utterance *which states do not border texas* is:

$$answer(exclude(state(all), next\_to(texas)))$$

where *next_to* is a domain-specific binary predicate that takes one argument (i.e., the entity *texas*) and returns a *set* of entities (e.g., the states bordering Texas) as its denotation. *all* is a special predicate that returns a collection of entities. *exclude* is a predicate that returns the difference between two input sets.

An advantage of FunQL is that the resulting *s*-expression encodes semantic compositionality and derivation of the logical forms. This property makes FunQL logical forms convenient to be predicted with recurrent neural networks (Vinyals et al., 2015; Choe and Charniak, 2016; Dyer et al., 2016). However, FunQL is less expressive than lambda calculus, partially due to the elimination of variables. A more compact logical formulation which our method also applies to is λ-DCS (Liang, 2013). In the absence of anaphora and composite binary predicates, conversion algorithms exist between FunQL and λ-DCS. However, we leave this to future work.

Ungrounded Meaning Representation We also use FunQL to express ungrounded meaning representations. The latter consist primarily of natural language predicates and domain-general predicates. Assuming for simplicity that domain-general predicates share the same vocabulary

[2] We discuss the merits and limitations of this assumption in Section 5

in ungrounded and grounded representations, the ungrounded representation for the example utterance is:

$$\mathit{answer}(\mathit{exclude}(\mathit{states}(\mathit{all}), \mathit{border}(\mathit{texas})))$$

where *states* and *border* are natural language predicates. In this work we consider five types of domain-general predicates illustrated in Table 1. Notice that domain-general predicates are often implicit, or represent extra-sentential knowledge. For example, the predicate *all* in the above utterance represents all states in the domain which are not mentioned in the utterance but are critical for working out the utterance denotation. Finally, note that for certain domain-general predicates, it also makes sense to extract natural language rationales (e.g., *not* is indicative for *exclude*). But we do not find this helpful in experiments.

In this work we constrain ungrounded representations to be structurally isomorphic to grounded ones. In order to derive the target logical forms, all we have to do is replacing predicates in the ungrounded representations with symbols in the knowledge base.

3 Modeling

In this section, we discuss our neural model which maps utterances to target logical forms. The semantic parsing task is decomposed in two stages: we first explain how an utterance is converted to an intermediate representation (Section 3.1), and then describe how it is grounded to a knowledge base (Section 3.2).

3.1 Generating Ungrounded Representations

At this stage, utterances are mapped to intermediate representations with a transition-based algorithm. In general, the transition system generates the representation by following a derivation tree (which contains a set of applied rules) and some canonical generation order (e.g., depth-first). For FunQL, a simple solution exists since the representation itself encodes the derivation. Consider again *answer*(*exclude*(*states(all)*, *border*(*texas*))) which is tree structured. Each predicate (e.g., *border*) can be visualized as a non-terminal node of the tree and each entity (e.g., *texas*) as a terminal. The predicate *all* is a special case which acts as a terminal directly. We can generate the tree with a top-down, depth first transition system reminiscent of recurrent neural network grammars (RNNGs; Dyer et al. 2016). Similar to RNNG, our algorithm uses a buffer to store input tokens in the utterance and a stack to store partially completed trees. A major difference in our semantic parsing scenario is that tokens in the buffer are not fetched in a sequential order or removed from the buffer. This is because the lexical alignment between an utterance and its semantic representation is hidden. Moreover, some predicates cannot be clearly anchored to a token span. Therefore, we allow the generation algorithm to pick tokens and combine logical forms in arbitrary orders, conditioning on the entire set of sentential features. Alternative solutions in the traditional semantic parsing literature include a floating chart parser (Pasupat and Liang, 2015) which allows to construct logical predicates out of thin air.

Our transition system defines three actions, namely NT, TER, and RED, explained below.

NT(X) generates a Non-Terminal predicate. This predicate is either a natural language expression such as *border*, or one of the domain-general predicates exemplified in Table 1 (e.g., *exclude*). The type of predicate is determined by the placeholder X and once generated, it is pushed onto the stack and represented as a non-terminal followed by an open bracket (e.g., *'border('*). The open bracket will be closed by a reduce operation.

TER(X) generates a TERminal entity or the special predicate *all*. Note that the terminal choice does not include variable (e.g., \$0, \$1), since FunQL is a variable-free language which sufficiently captures the semantics of the datasets we work with. The framework could be extended to generate directly acyclic graphs by incorporating variables with additional transition actions for handling variable mentions and co-reference.

RED stands for REDuce and is used for subtree completion. It recursively pops elements from the stack until an open non-terminal node is encountered. The non-terminal is popped as well, after which a composite term representing the entire subtree, e.g., *border(texas)*, is pushed back to the stack. If a RED action results in having no more open non-terminals left on the stack, the transition system terminates. Table 2 shows the transition actions used to generate our running example.

The model generates the ungrounded representation U conditioned on utterance x by recursively calling one of the above three actions. Note that U is defined by a sequence of actions (denoted

Sentence: *which states do not border texas*
Non-terminal symbols in buffer: *which*, *states*, *do*, *not*, *border*
Terminal symbols in buffer: *texas*

Stack	**Action**	NT choice	TER choice
	NT	*answer*	
answer (	NT	*exclude*	
answer (*exclude* (	NT	*states*	
answer (*exclude* (*states* (	TER		*all*
answer (*exclude* (*states* (*all*	RED		
answer (*exclude* (*states* (*all*)	NT	*border*	
answer (*exclude* (*states* (*all*) , *border* (	TER		*texas*
answer (*exclude* (*states* (*all*) , *border* (*texas*	RED		
answer (*exclude* (*states* (*all*) , *border* (*texas*)	RED		
answer (*exclude* (*states* (*all*) , *border* (*texas*))	RED		
answer (*exclude* (*states* (*all*) , *border* (*texas*)))			

Table 2: Actions taken by the transition system for generating the ungrounded meaning representation of the example utterance. Symbols in red indicate domain-general predicates.

by a) and a sequence of term choices (denoted by u) as shown in Table 2. The conditional probability $p(U|x)$ is factorized over time steps as:

$$\begin{aligned} p(U|x) &= p(a, u|x) \quad (1) \\ &= \prod_{t=1}^{T} p(a_t|a_{<t}, x) p(u_t|a_{<t}, x)^{\mathbb{I}(a_t \neq \text{RED})} \end{aligned}$$

where $\mathbb{I}$ is an indicator function.

To predict the actions of the transition system, we encode the input buffer with a bidirectional LSTM (Hochreiter and Schmidhuber, 1997) and the output stack with a stack-LSTM (Dyer et al., 2015). At each time step, the model uses the representation of the transition system e_t to predict an action:

$$p(a_t|a_{<t}, x) \propto \exp(W_a \cdot e_t) \quad (2)$$

where e_t is the concatenation of the buffer representation b_t and the stack representation s_t. While the stack representation s_t is easy to retrieve as the top state of the stack-LSTM, obtaining the buffer representation b_t is more involved. This is because we do not have an explicit buffer representation due to the non-projectivity of semantic parsing. We therefore compute at each time step an adaptively weighted representation of b_t (Bahdanau et al., 2015) conditioned on the stack representation s_t. This buffer representation is then concatenated with the stack representation to form the system representation e_t.

When the predicted action is either NT or TER, an ungrounded term u_t (either a predicate or an entity) needs to be chosen from the candidate list depending on the specific placeholder X. To select a domain-general term, we use the same representation of the transition system e_t to compute a probability distribution over candidate terms:

$$p(u_t^{\text{GENERAL}}|a_{<t}, x) \propto \exp(W_p \cdot e_t) \quad (3)$$

To choose a natural language term, we directly compute a probability distribution of all natural language terms (in the buffer) conditioned on the stack representation s_t and select the most relevant term (Jia and Liang, 2016):

$$p(u_t^{\text{NL}}|a_{<t}, x) \propto \exp(s_t) \quad (4)$$

When the predicted action is RED, the completed subtree is composed into a single representation on the stack. For the choice of composition function, we use a single-layer neural network as in Dyer et al. (2015), which takes as input the concatenated representation of the predicate and argument of the subtree.

3.2 Generating Grounded Representations

Since we constrain the network to learn ungrounded structures that are isomorphic to the target meaning representation, converting ungrounded representations to grounded ones becomes a simple lexical mapping problem. For simplicity, hereafter we do not differentiate natural language and domain-general predicates.

To map an ungrounded term u_t to a grounded term g_t, we compute the conditional probability of g_t given u_t with a bi-linear neural network:

$$p(g_t|u_t) \propto \exp \vec{u_t} \cdot W_{ug} \cdot \vec{g_t}^\top \tag{5}$$

where $\vec{u_t}$ is the contextual representation of the ungrounded term given by the bidirectional LSTM, $\vec{g_t}$ is the grounded term embedding, and W_{ug} is the weight matrix.

The above grounding step can be interpreted as learning a lexicon: the model exclusively relies on the intermediate representation U to predict the target meaning representation G without taking into account any additional features based on the utterance. In practice, U may provide sufficient contextual background for closed domain semantic parsing where an ungrounded predicate often maps to a single grounded predicate, but is a relatively impoverished representation for parsing large open-domain knowledge bases like Freebase. In this case, we additionally rely on a discriminative reranker which ranks the grounded representations derived from ungrounded representations (see Section 3.4).

3.3 Training Objective

When the target meaning representation is available, we directly compare it against our predictions and back-propagate. When only denotations are available, we compare surrogate meaning representations against our predictions (Reddy et al., 2014). Surrogate representations are those with the correct denotations. When there exist multiple surrogate representations,[3] we select one randomly and back-propagate. The global effect of the above update rule is close to maximizing the marginal likelihood of denotations, which differs from recent work on weakly-supervised semantic parsing based on reinforcement learning (Neelakantan et al., 2017).

Consider utterance x with ungrounded meaning representation U, and grounded meaning representation G. Both U and G are defined with a sequence of transition actions (same for U and G) and a sequence of terms (different for U and G). Recall that $a = [a_1, \cdots, a_n]$ denotes the transition action sequence defining U and G; let $u = [u_1, \cdots, u_k]$ denote the ungrounded terms (e.g., predicates), and $g = [g_1, \cdots, g_k]$ the grounded terms. We aim to maximize the likelihood of the grounded meaning representation $p(G|x)$ over all training examples. This likelihood can be decomposed into the likelihood of the grounded action sequence $p(a|x)$ and the grounded term sequence $p(g|x)$, which we optimize separately.

For the grounded action sequence (which by design is the same as the ungrounded action sequence and therefore the output of the transition system), we can directly maximize the log likelihood $\log p(a|x)$ for all examples:

$$\mathcal{L}_a = \sum_{x \in \mathcal{T}} \log p(a|x) = \sum_{x \in \mathcal{T}} \sum_{t=1}^{n} \log p(a_t|x) \tag{6}$$

where $\mathcal{T}$ denotes examples in the training data.

For the grounded term sequence g, since the intermediate ungrounded terms are latent, we maximize the expected log likelihood of the grounded terms $\sum_u [p(u|x) \log p(g|u, x)]$ for all examples, which is a lower bound of the log likelihood $\log p(g|x)$:

$$\begin{aligned} \mathcal{L}_g &= \sum_{x \in \mathcal{T}} \sum_{u} [p(u|x) \log p(g|u, x)] \\ &= \sum_{x \in \mathcal{T}} \sum_{u} \left[p(u|x) \sum_{t=1}^{k} \log p(g_t|u_t) \right] \end{aligned} \tag{7}$$

The final objective is the combination of $\mathcal{L}_a$ and $\mathcal{L}_g$, denoted as $\mathcal{L}_G = \mathcal{L}_a + \mathcal{L}_g$. We optimize this objective with the method described in Lei et al. (2016).

3.4 Reranker

As discussed above, for open domain semantic parsing, solely relying on the ungrounded representation would result in an impoverished model lacking sentential context useful for disambiguation decisions. For all Freebase experiments, we followed previous work (Berant et al., 2013; Berant and Liang, 2014; Reddy et al., 2014) in additionally training a discriminative ranker to re-rank grounded representations globally.

The discriminative ranker is a maximum-entropy model (Berant et al., 2013). The objective is to maximize the log likelihood of the correct answer y given x by summing over all grounded candidates G with denotation y (i.e.,$[\![G]\!]_\mathcal{K} = y$):

$$\mathcal{L}_y = \sum_{(x,y) \in \mathcal{T}} \log \sum_{[\![G]\!]_\mathcal{K} = y} p(G|x) \tag{8}$$

$$p(G|x) \propto \exp\{f(G, x)\} \tag{9}$$

where $f(G, x)$ is a feature function that maps pair (G, x) into a feature vector. We give details on the features we used in Section 4.2.

[3] The average Freebase surrogate representations obtained with highest denotation match (F1) is 1.4.

4 Experiments

In this section, we verify empirically that our semantic parser derives useful meaning representations. We give details on the evaluation datasets and baselines used for comparison. We also describe implementation details and the features used in the discriminative ranker.

4.1 Datasets

We evaluated our model on the following datasets which cover different domains, and use different types of training data, i.e., pairs of natural language utterances and grounded meanings or question-answer pairs.

GEOQUERY (Zelle and Mooney, 1996) contains 880 questions and database queries about US geography. The utterances are compositional, but the language is simple and vocabulary size small. The majority of questions include at most one entity. SPADES (Bisk et al., 2016) contains 93,319 questions derived from CLUEWEB09 (Gabrilovich et al., 2013) sentences. Specifically, the questions were created by randomly removing an entity, thus producing sentence-denotation pairs (Reddy et al., 2014). The sentences include two or more entities and although they are not very compositional, they constitute a large-scale dataset for neural network training. WEBQUESTIONS (Berant et al., 2013) contains 5,810 question-answer pairs. Similar to SPADES, it is based on Freebase and the questions are not very compositional. However, they are real questions asked by people on the Web. Finally, GRAPHQUESTIONS (Su et al., 2016) contains 5,166 question-answer pairs which were created by showing 500 Freebase graph queries to Amazon Mechanical Turk workers and asking them to paraphrase them into natural language.

4.2 Implementation Details

Amongst the four datasets described above, GEOQUERY has annotated logical forms which we directly use for training. For the other three datasets, we treat surrogate meaning representations which lead to the correct answer as gold standard. The surrogates were selected from a subset of candidate Freebase graphs, which were obtained by entity linking. Entity mentions in SPADES have been automatically annotated with Freebase entities (Gabrilovich et al., 2013). For WEBQUESTIONS and GRAPHQUESTIONS, we follow the procedure described in Reddy et al. (2016). We identify potential entity spans using seven handcrafted part-of-speech patterns and associate them with Freebase entities obtained from the Freebase/KG API.[4] We use a structured perceptron trained on the entities found in WEBQUESTIONS and GRAPHQUESTIONS to select the top 10 non-overlapping entity disambiguation possibilities. We treat each possibility as a candidate input utterance, and use the perceptron score as a feature in the discriminative reranker, thus leaving the final disambiguation to the semantic parser.

Apart from the entity score, the discriminative ranker uses the following basic features. The first feature is the likelihood score of a grounded representation aggregating all intermediate representations. The second set of features include the embedding similarity between the relation and the utterance, as well as the similarity between the relation and the question words. The last set of features includes the answer type as indicated by the last word in the Freebase relation (Xu et al., 2016).

We used the Adam optimizer for training with an initial learning rate of 0.001, two momentum parameters [0.99, 0.999], and batch size 1. The dimensions of the word embeddings, LSTM states, entity embeddings and relation embeddings are $[50, 100, 100, 100]$. The word embeddings were initialized with Glove embeddings (Pennington et al., 2014). All other embeddings were randomly initialized.

4.3 Results

Experimental results on the four datasets are summarized in Tables 3–6. We present comparisons of our system which we call SCANNER (as a shorthand for **S**ymboli**C** me**AN**i**N**g r**E**p**R**esentation) against a variety of models previously described in the literature.

GEOQUERY results are shown in Table 5. The first block contains symbolic systems, whereas neural models are presented in the second block. We report accuracy which is defined as the proportion of the utterance that are correctly parsed to their gold standard logical forms. All previous neural systems (Dong and Lapata, 2016; Jia and Liang, 2016) treat semantic parsing as a sequence transduction problem and use LSTMs to directly map utterances to logical forms. SCANNER yields performance improvements over these

[4] http://developers.google.com/freebase/

Models	F1
Berant et al. (2013)	35.7
Yao and Van Durme (2014)	33.0
Berant and Liang (2014)	39.9
Bast and Haussmann (2015)	49.4
Berant and Liang (2015)	49.7
Reddy et al. (2016)	50.3
Bordes et al. (2014)	39.2
Dong et al. (2015)	40.8
Yih et al. (2015)	52.5
Xu et al. (2016)	53.3
Neural Baseline	48.3
SCANNER	49.4

Table 3: WEBQUESTIONS results.

Models	F1
SEMPRE (Berant et al., 2013)	10.80
PARASEMPRE (Berant and Liang, 2014)	12.79
JACANA (Yao and Van Durme, 2014)	5.08
Neural Baseline	16.24
SCANNER	17.02

Table 4: GRAPHQUESTIONS results. Numbers for comparison systems are from Su et al. (2016).

Models	Accuracy
Zettlemoyer and Collins (2005)	79.3
Zettlemoyer and Collins (2007)	86.1
Kwiatkowksi et al. (2010)	87.9
Kwiatkowski et al. (2011)	88.6
Kwiatkowski et al. (2013)	88.0
Zhao and Huang (2015)	88.9
Liang et al. (2011)	91.1
Dong and Lapata (2016)	84.6
Jia and Liang (2016)	85.0
Jia and Liang (2016) with extra data	89.1
SCANNER	86.7

Table 5: GEOQUERY results.

Models	F1
Unsupervised CCG (Bisk et al., 2016)	24.8
Semi-supervised CCG (Bisk et al., 2016)	28.4
Neural baseline	28.6
Supervised CCG (Bisk et al., 2016)	30.9
Rule-based system (Bisk et al., 2016)	31.4
SCANNER	31.5

Table 6: SPADES results.

systems when using comparable data sources for training. Jia and Liang (2016) achieve better results with synthetic data that expands GEOQUERY; we could adopt their approach to improve model performance, however, we leave this to future work.

Table 6 reports SCANNER's performance on SPADES. For all Freebase related datasets we use average F1 (Berant et al., 2013) as our evaluation metric. Previous work on this dataset has used a semantic parsing framework similar to ours where natural language is converted to an intermediate syntactic representation and then grounded to Freebase. Specifically, Bisk et al. (2016) evaluate the effectiveness of four different CCG parsers on the semantic parsing task when varying the amount of supervision required. As can be seen, SCANNER outperforms all CCG variants (from unsupervised to fully supervised) without having access to any manually annotated derivations or lexicons. For fair comparison, we also built a neural baseline that encodes an utterance with a recurrent neural network and then predicts a grounded meaning representation directly (Ture and Jojic, 2016; Yih et al., 2016). Again, we observe that SCANNER outperforms this baseline.

Results on WEBQUESTIONS are summarized in Table 3. SCANNER obtains performance on par with the best symbolic systems (see the first block in the table). It is important to note that Bast and Haussmann (2015) develop a question answering system, which contrary to ours cannot produce meaning representations whereas Berant and Liang (2015) propose a sophisticated agenda-based parser which is trained borrowing ideas from imitation learning. SCANNER is conceptually similar to Reddy et al. (2016) who also learn a semantic parser via intermediate representations which they generate based on the output of a dependency parser. SCANNER performs competitively despite not having access to any linguistically-informed syntactic structures. The second block in Table 3 reports the results of several neural systems. Xu et al. (2016) represent the state of the art on WEBQUESTIONS. Their system uses Wikipedia to prune out erroneous candidate answers extracted from Freebase. Our model would also benefit from a similar post-processing step. As in previous experiments, SCANNER outperforms the neural baseline, too.

Finally, Table 4 presents our results on GRAPHQUESTIONS. We report F1 for SCANNER, the neural baseline model, and three symbolic systems presented in Su et al. (2016). SCANNER achieves a new state of the art on this dataset with a gain of 4.23 F1 points over the best previously reported model.

4.4 Analysis of Intermediate Representations

Since a central feature of our parser is that it learns intermediate representations with natural language predicates, we conducted additional experiments in order to inspect their quality. For GEOQUERY

Metrics	Accuracy
Exact match	79.3
Structure match	89.6
Token match	96.5

Table 7: GeoQuery evaluation of ungrounded meaning representations. We report accuracy against a manually created gold standard.

which contains only 280 test examples, we manually annotated intermediate representations for the test instances and evaluated the learned representations against them. The experimental setup aims to shows how humans can participate in improving the semantic parser with feedback at the intermediate stage. In terms of evaluation, we use three metrics shown in Table 7. The first row shows the percentage of exact matches between the predicted representations and the human annotations. The second row refers to the percentage of structure matches, where the predicted representations have the same structure as the human annotations, but may not use the same lexical terms. Among structurally correct predictions, we additionally compute how many tokens are correct, as shown in the third row. As can be seen, the induced meaning representations overlap to a large extent with the human gold standard.

We also evaluated the intermediate representations created by ScanneR on the other three (Freebase) datasets. Since creating a manual gold standard for these large datasets is time-consuming, we compared the induced representations against the output of a syntactic parser. Specifically, we converted the questions to event-argument structures with EasyCCG (Lewis and Steedman, 2014), a high coverage and high accuracy CCG parser. EasyCCG extracts predicate-argument structures with a labeled F-score of 83.37%. For further comparison, we built a simple baseline which identifies predicates based on the output of the Stanford POS-tagger (Manning et al., 2014) following the ordering VBD ≫ VBN ≫ VB ≫ VBP ≫ VBZ ≫ MD.

As shown in Table 8, on Spades and WebQuestions, the predicates learned by our model match the output of EasyCCG more closely than the heuristic baseline. But for GraphQuestions which contains more compositional questions, the mismatch is higher. However, since the key idea of our model is to capture salient meaning for the task at hand rather than strictly obey syntax, we would not expect the predicates induced by our system to entirely agree with those produced by the syntactic parser. To further analyze how the learned predicates differ from syntax-based ones, we grouped utterances in Spades into four types of linguistic constructions: coordination (*conj*), control and raising (*control*), prepositional phrase attachment (*pp*), and subordinate clauses (*subord*). Table 8 also shows the breakdown of matching scores per linguistic construction, with the number of utterances in each type. In Table 9, we provide examples of predicates identified by ScanneR, indicating whether they agree or not with the output of EasyCCG. As a reminder, the task in Spades is to predict the entity masked by a *blank* symbol (__).

Dataset	ScanneR	Baseline
Spades	51.2	45.5
–*conj* (1422)	56.1	66.4
–*control* (132)	28.3	40.5
–*pp* (3489)	46.2	23.1
–*subord* (76)	37.9	52.9
WebQuestions	42.1	25.5
GraphQuestions	11.9	15.3

Table 8: Evaluation of predicates induced by ScanneR against EasyCCG. We report F1(%) across datasets. For Spades, we also provide a breakdown for various utterance types.

As can be seen in Table 8, the matching score is relatively high for utterances involving coordination and prepositional phrase attachments. The model will often identify informative predicates (e.g., nouns) which do not necessarily agree with linguistic intuition. For example, in the utterance *wilhelm_maybach and his son __ started maybach in 1909* (see Table 9), ScanneR identifies the predicate-argument structure *son(wilhelm_maybach)* rather than *started(wilhelm_maybach)*. We also observed that the model struggles with control and subordinate constructions. It has difficulty distinguishing control from raising predicates as exemplified in the utterance *ceo john_thain agreed to leave __* from Table 9, where it identifies the raising predicate *agreed*. For subordinate clauses, Scanner tends to take shortcuts identifying as predicates words closest to the *blank* symbol.

5 Discussion

We presented a neural semantic parser which converts natural language utterances to grounded meaning representations via intermediate predicate-argument structures. Our model

conj	the boeing_company was founded in 1916 and is headquartered in __ , illinois . nstar was founded in 1886 and is based in boston , __ . the __ is owned and operated by zuffa_,_llc , headquarted in las_vegas , nevada . hugh attended __ and then shifted to uppingham_school in england .	__ was incorporated in 1947 and is based in new_york_city . the ifbb was formed in 1946 by president ben_weider and his brother __ . wilhelm_maybach and his son __ started maybach in 1909 . __ was founded in 1996 and is headquartered in chicago .
control	__ threatened to kidnap russ . __ has also been confirmed to play captain_haddock . hoffenberg decided to leave __ . __ is reportedly trying to get impregnated by djimon now . for right now , __ are inclined to trust obama to do just that .	__ agreed to purchase wachovia_corp . ceo john_thain agreed to leave __ . so nick decided to create __ . salva later went on to make the non clown-based horror __ . eddie dumped debbie to marry __ when carrie was 2 .
pp	__ is the home of the university_of_tennessee . chu is currently a physics professor at __ . youtube is based in __ , near san_francisco , california . mathematica is a product of __ .	jobs will retire from __ . the nab is a strong advocacy group in __ . this one starred robert_reed , known mostly as __ . __ is positively frightening as detective bud_white .
subord	the__ is a national testing board that is based in toronto . __ is a corporation that is wholly owned by the city_of_edmonton . unborn is a scary movie that stars __ . __ 's third wife was actress melina_mercouri , who died in 1994 . sure , there were __ who liked the shah .	founded the __ , which is now also a designated terrorist group . __ is an online bank that ebay owns . zoya_akhtar is a director , who has directed the upcoming movie __ . imelda_staunton , who plays __ , is genius . __ is the important president that american ever had . plus mitt_romney is the worst governor that __ has had .

Table 9: Informative predicates identified by SCANNER in various types of utterances. Yellow predicates were identified by both SCANNER and EASYCCG, red predicates by SCANNER alone, and green predicates by EASYCCG alone.

essentially jointly learns how to parse natural language semantics and the lexicons that help grounding. Compared to previous neural semantic parsers, our model is more interpretable as the intermediate structures are useful for inspecting what the model has learned and whether it matches linguistic intuition.

An assumption our model imposes is that ungrounded and grounded representations are structurally isomorphic. An advantage of this assumption is that tokens in the ungrounded and grounded representations are strictly aligned. This allows the neural network to focus on parsing and lexical mapping, sidestepping the challenging structure mapping problem which would result in a larger search space and higher variance. On the negative side, the structural isomorphism assumption restricts the expressiveness of the model, especially since one of the main benefits of adopting a two-stage parser is the potential of capturing domain-independent semantic information via the intermediate representation. While it would be challenging to handle drastically non-isomorphic structures in the current model, it is possible to perform local structure matching, i.e., when the mapping between natural language and domain-specific predicates is many-to-one or one-to-many. For instance, Freebase does not contain a relation representing *daughter*, using instead two relations representing *female* and *child*. Previous work (Kwiatkowski et al., 2013) models such cases by introducing collapsing (for many-to-one mapping) and expansion (for one-to-many mapping) operators. Within our current framework, these two types of structural mismatches can be handled with semi-Markov assumptions (Sarawagi and Cohen, 2005; Kong et al., 2016) in the parsing (i.e., predicate selection) and the grounding steps, respectively. Aside from relaxing strict isomorphism, we would also like to perform cross-domain semantic parsing where the first stage of the semantic parser is shared across domains.

Acknowledgments We would like to thank three anonymous reviewers, members of the Edinburgh ILCC and the IBM Watson, and Abulhair Saparov for feedback. The support of the European Research Council under award number 681760 "Translating Multiple Modalities into Text" is gratefully acknowledged.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of ICLR 2015*. San Diego, California.

Hannah Bast and Elmar Haussmann. 2015. More accurate question answering on Freebase. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, pages 1431–1440.

Emily M Bender, Dan Flickinger, Stephan Oepen, Woodley Packard, and Ann Copestake. 2015. Layers of interpretation: On grammar and compositionality. In *Proceedings of the 11th International Conference on Computational Semantics*. London, UK, pages 239–249.

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

Jonathan Berant and Percy Liang. 2014. Semantic parsing via paraphrasing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Baltimore, Maryland, pages 1415–1425.

Jonathan Berant and Percy Liang. 2015. Imitation learning of agenda-based semantic parsers. *Transactions of the Association for Computational Linguistics* 3:545–558.

Yonatan Bisk, Siva Reddy, John Blitzer, Julia Hockenmaier, and Mark Steedman. 2016. Evaluating induced CCG parsers on grounded semantic parsing. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas, pages 2022–2027.

Antoine Bordes, Sumit Chopra, and Jason Weston. 2014. Question answering with subgraph embeddings. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar, pages 615–620.

Qingqing Cai and Alexander Yates. 2013. Large-scale semantic parsing via schema matching and lexicon extension. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Sofia, Bulgaria, pages 423–433.

Do Kook Choe and Eugene Charniak. 2016. Parsing as language modeling. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas, pages 2331–2336.

Li Dong and Mirella Lapata. 2016. Language to logical form with neural attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany, pages 33–43.

Li Dong, Furu Wei, Ming Zhou, and Ke Xu. 2015. Question answering over Freebase with multi-column convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Beijing, China, pages 260–269.

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A. Smith. 2015. Transition-based dependency parsing with stack long short-term memory. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Beijing, China, pages 334–343.

Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A. Smith. 2016. Recurrent neural network grammars. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. San Diego, California, pages 199–209.

Jeffrey Flanigan, Sam Thomson, Jaime Carbonell, Chris Dyer, and Noah A. Smith. 2014. A discriminative graph-based parser for the abstract meaning representation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Baltimore, Maryland, pages 1426–1436.

Evgeniy Gabrilovich, Michael Ringgaard, and Amarnag Subramanya. 2013. FACC1: Freebase annotation of ClueWeb corpora, version 1 (release date 2013-06-26, format version 1, correction level 0) .

Matt Gardner and Jayant Krishnamurthy. 2017. Open-Vocabulary Semantic Parsing with both Distributional Statistics and Formal Knowledge. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*. San Francisco, California, pages 3195–3201.

Jonas Groschwitz, Alexander Koller, and Christoph Teichmann. 2015. Graph parsing with s-graph grammars. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Beijing, China, pages 1481–1490.

Steve Harris, Andy Seaborne, and Eric Prud'hommeaux. 2013. SPARQL 1.1 query language. *W3C recommendation* 21(10).

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Florian Holzschuher and René Peinl. 2013. Performance of graph query languages: comparison of cypher, gremlin and native access in Neo4j. In *Proceedings of the Joint EDBT/ICDT 2013 Workshops*. ACM, pages 195–204.

Robin Jia and Percy Liang. 2016. Data recombination for neural semantic parsing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany, pages 12–22.

Rohit J. Kate, Yuk Wah Wong, and Raymond J. Mooney. 2005. Learning to Transform Natural to Formal Languages. In *Proceedings for the 20th National Conference on Artificial Intelligence*. Pittsburgh, Pennsylvania, pages 1062–1068.

Lingpeng Kong, Chris Dyer, and Noah A Smith. 2016. Segmental recurrent neural networks. In *Proceedings of ICLR 2016*. San Juan, Puerto Rico.

Tomáš Kočiský, Gábor Melis, Edward Grefenstette, Chris Dyer, Wang Ling, Phil Blunsom, and Karl Moritz Hermann. 2016. Semantic parsing with semi-supervised sequential autoencoders. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas, pages 1078–1087.

Jayant Krishnamurthy and Tom Mitchell. 2012. Weakly supervised training of semantic parsers. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Jeju Island, Korea, pages 754–765.

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* 3:257–270.

Tom Kwiatkowksi, 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*. Cambridge, MA, pages 1223–1233.

Tom Kwiatkowski, Eunsol Choi, Yoav Artzi, and Luke Zettlemoyer. 2013. Scaling Semantic Parsers with On-the-Fly Ontology Matching. In *Proceedings of Empirical Methods on Natural Language Processing*. pages 1545–1556.

Tom Kwiatkowski, Luke Zettlemoyer, Sharon Goldwater, and Mark Steedman. 2011. Lexical generalization in CCG grammar induction for semantic parsing. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. Edinburgh, Scotland, pages 1512–1523.

Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. Rationalizing neural predictions. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas, pages 107–117.

Mike Lewis and Mark Steedman. 2014. A* CCG parsing with a supertag-factored model. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar, pages 990–1000.

Percy Liang. 2013. Lambda dependency-based compositional semantics. *arXiv preprint arXiv:1309.4408* .

Percy Liang, Michael Jordan, and Dan Klein. 2011. Learning dependency-based compositional semantics. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Portland, Oregon, pages 590–599.

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

Arvind Neelakantan, Quoc V Le, Martin Abadi, Andrew McCallum, and Dario Amodei. 2017. Learning a natural language interface with neural programmer. In *Proceedings of ICLR 2017*. Toulon, France.

Panupong Pasupat and Percy Liang. 2015. Compositional semantic parsing on semi-structured tables. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Beijing, China, pages 1470–1480.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar, pages 1532–1543.

Siva Reddy, Mirella Lapata, and Mark Steedman. 2014. Large-scale semantic parsing without question-answer pairs. *Transactions of the Association for Computational Linguistics* 2:377–392.

Siva Reddy, Oscar Täckström, Michael Collins, Tom Kwiatkowski, Dipanjan Das, Mark Steedman, and Mirella Lapata. 2016. Transforming dependency structures to logical forms for semantic parsing. *Transactions of the Association for Computational Linguistics* 4:127–140.

Sunita Sarawagi and William W Cohen. 2005. Semi-markov conditional random fields for information extraction. In *Advances in Neural Information Processing Systems 17*, MIT Press, pages 1185–1192.

Yu Su, Huan Sun, Brian Sadler, Mudhakar Srivatsa, Izzeddin Gur, Zenghui Yan, and Xifeng Yan. 2016. On generating characteristic-rich question sets for qa evaluation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas, pages 562–572.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27*, MIT Press, pages 3104–3112.

Ferhan Ture and Oliver Jojic. 2016. Simple and effective question answering with recurrent neural networks. *arXiv preprint arXiv:1606.05029* .

Oriol Vinyals, Łukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. 2015. Grammar as a foreign language. In *Advances in Neural Information Processing Systems 28*. MIT Press, pages 2773–2781.

Yuk Wah Wong and Raymond Mooney. 2006. Learning for semantic parsing with statistical machine translation. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*. New York City, USA, pages 439–446.

Kun Xu, Siva Reddy, Yansong Feng, Songfang Huang, and Dongyan Zhao. 2016. Question answering on Freebase via relation extraction and textual evidence. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany, pages 2326–2336.

Xuchen Yao and Benjamin Van Durme. 2014. Information extraction over structured data: Question answering with Freebase. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Baltimore, Maryland, pages 956–966.

Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. 2015. Semantic parsing via staged query graph generation: Question answering with knowledge base. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Beijing, China, pages 1321–1331.

Wen-tau Yih, Matthew Richardson, Chris Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Berlin, Germany, pages 201–206.

John M. Zelle and Raymond J. Mooney. 1996. Learning to Parse Database Queries Using Inductive Logic Programming. In *Proceedings of the 13th National Conference on Artificial Intelligence*. Portland, Oregon, pages 1050–1055.

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

Luke S. Zettlemoyer and Michael Collins. 2005. Learning to Map Sentences to Logical Form: Structured Classification with Probabilistic Categorial Grammars. In *Proceedings of 21st Conference in Uncertainilty in Artificial Intelligence*. Edinburgh, Scotland, pages 658–666.

Kai Zhao and Liang Huang. 2015. Type-driven incremental semantic parsing with polymorphism. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Denver, Colorado, pages 1416–1421.