Semantic Parsing via Paraphrasing

Jonathan Berant
Stanford University
joberant@stanford.edu

Percy Liang
Stanford University
pliang@cs.stanford.edu

Abstract

A central challenge in semantic parsing is handling the myriad ways in which knowledge base predicates can be expressed. Traditionally, semantic parsers are trained primarily from text paired with knowledge base information. Our goal is to exploit the much larger amounts of raw text not tied to any knowledge base. In this paper, we turn semantic parsing on its head. Given an input utterance, we first use a simple method to deterministically generate a set of candidate logical forms with a canonical realization in natural language for each. Then, we use a paraphrase model to choose the realization that best paraphrases the input, and output the corresponding logical form. We present two simple paraphrase models, an *association* model and a *vector space* model, and train them jointly from question-answer pairs. Our system PARASEMPRE improves state-of-the-art accuracies on two recently released question-answering datasets.

1 Introduction

We consider the semantic parsing problem of mapping natural language utterances into logical forms to be executed on a knowledge base (KB) (Zelle and Mooney, 1996; Zettlemoyer and Collins, 2005; Wong and Mooney, 2007; Kwiatkowski et al., 2010). Scaling semantic parsers to large knowledge bases has attracted substantial attention recently (Cai and Yates, 2013; Berant et al., 2013; Kwiatkowski et al., 2013), since it drives applications such as question answering (QA) and information extraction (IE).

Semantic parsers need to somehow associate natural language phrases with logical predicates, e.g., they must learn that the constructions *"What does X do for a living?"*, *"What is X's profession?"*, and *"Who is X?"*, should all map to the logical predicate `Profession`. To learn these mappings, traditional semantic parsers use data which pairs natural language with the KB. However, this leaves untapped a vast amount of text not related to the KB. For instance, the utterances *"Where is ACL in 2014?"* and *"What is the location of ACL 2014?"* cannot be used in traditional semantic parsing methods, since the KB does not contain an entity `ACL2014`, but this pair clearly contains valuable linguistic information. As another reference point, out of 500,000 relations extracted by the ReVerb Open IE system (Fader et al., 2011), only about 10,000 can be aligned to Freebase (Berant et al., 2013).

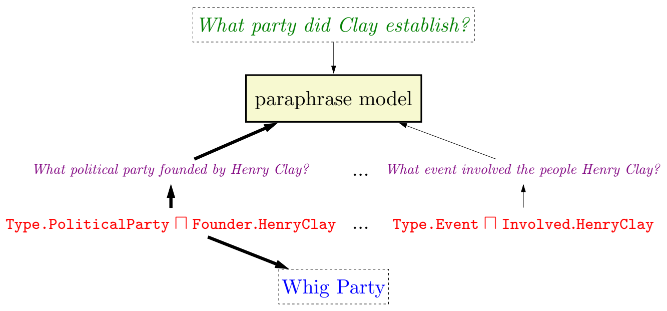
What party did Clay establish?
paraphrase model
What political party founded by Henry Clay?
...
What event involved the people Henry Clay?
Type.PoliticalParty ⊓ Founder.HenryClay
...
Type.Event ⊓ Involved.HenryClay
Whig Party

Figure 1: Semantic parsing via paraphrasing: For each candidate logical form (in red), we generate canonical utterances (in purple). The model is trained to paraphrase the input utterance (in green) into the canonical utterances associated with the correct denotation (in blue).

In this paper, we present a novel approach for semantic parsing based on paraphrasing that can exploit large amounts of text not covered by the KB (Figure 1). Our approach targets factoid questions with a modest amount of compositionality. Given an input utterance, we first use a simple deterministic procedure to construct a manageable set of candidate logical forms (ideally, we would generate canonical utterances for all possible logical forms, but this is intractable). Next, we heuris-

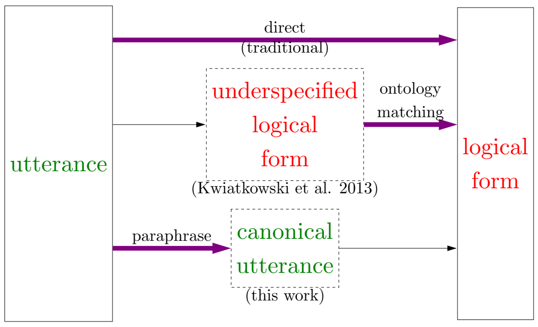

Figure 2: The main challenge in semantic parsing is coping with the *mismatch* between language and the KB. (a) Traditionally, semantic parsing maps utterances directly to logical forms. (b) Kwiatkowski et al. (2013) map the utterance to an underspecified logical form, and perform ontology matching to handle the mismatch. (c) We approach the problem in the other direction, generating canonical utterances for logical forms, and use paraphrase models to handle the mismatch.

tically generate *canonical utterances* for each logical form based on the text descriptions of predicates from the KB. Finally, we choose the canonical utterance that best paraphrases the input utterance, and thereby the logical form that generated it. We use two complementary paraphrase models: an *association model* based on aligned phrase pairs extracted from a monolingual parallel corpus, and a *vector space model*, which represents each utterance as a vector and learns a similarity score between them. The entire system is trained jointly from question-answer pairs only.

Our work relates to recent lines of research in semantic parsing and question answering. Kwiatkowski et al. (2013) first maps utterances to a domain-independent *intermediate logical form*, and then performs ontology matching to produce the final logical form. In some sense, we approach the problem from the opposite end, using an *intermediate utterance*, which allows us to employ paraphrasing methods (Figure 2). Fader et al. (2013) presented a QA system that maps questions onto simple queries against Open IE extractions, by learning paraphrases from a large monolingual parallel corpus, and performing a single paraphrasing step. We adopt the idea of using paraphrasing for QA, but suggest a more general paraphrase model and work against a formal KB (Freebase).

We apply our semantic parser on two datasets: WEBQUESTIONS (Berant et al., 2013), which contains 5,810 question-answer pairs with common questions asked by web users; and FREE917 (Cai and Yates, 2013), which has 917 questions manually authored by annotators. On WEBQUESTIONS, we obtain a relative improvement of 12% in accuracy over the state-of-the-art, and on FREE917 we match the current best performing system. The source code of our system PARASEMPRE is released at `http://www-nlp.stanford.edu/software/sempre/`.

2 Setup

Our task is as follows: Given (i) a knowledge base $\mathcal{K}$, and (ii) a training set of question-answer pairs $\{(x_i, y_i)\}_{i=1}^n$, output a semantic parser that maps new questions x to answers y via latent logical forms z. Let $\mathcal{E}$ denote a set of *entities* (e.g., `BillGates`), and let $\mathcal{P}$ denote a set of *properties* (e.g., `PlaceOfBirth`). A *knowledge base* $\mathcal{K}$ is a set of *assertions* $(e_1, p, e_2) \in \mathcal{E} \times \mathcal{P} \times \mathcal{E}$ (e.g., $(\texttt{BillGates}, \texttt{PlaceOfBirth}, \texttt{Seattle})$). We use the Freebase KB (Google, 2013), which has 41M entities, 19K properties, and 596M assertions.

To query the KB, we use a logical language called *simple* λ*-DCS*. In simple λ-DCS, an entity (e.g., `Seattle`) is a unary predicate (i.e., a subset of $\mathcal{E}$) denoting a singleton set containing that entity. A property (which is a binary predicate) can be joined with a unary predicate; e.g., `Founded.Microsoft` denotes the entities that are Microsoft founders. In $\texttt{PlaceOfBirth.Seattle} \sqcap \texttt{Founded.Microsoft}$, an intersection operator allows us to denote the set of Seattle-born Microsoft founders. A reverse operator reverses the order of arguments: $\mathbf{R}[\texttt{PlaceOfBirth}].\texttt{BillGates}$ denotes Bill Gates's birthplace (in contrast to `PlaceOfBirth.Seattle`). Lastly, $\texttt{count}(\texttt{Founded.Microsoft})$ denotes set cardinality, in this case, the number of Microsoft founders. The denotation of a logical form z with respect to a KB $\mathcal{K}$ is given by $[\![z]\!]_{\mathcal{K}}$. For a formal description of simple λ-DCS, see Liang (2013) and Berant et al. (2013).

3 Model overview

We now present the general framework for semantic parsing via paraphrasing, including the model and the learning algorithm. In Sections 4 and 5, we provide the details of our implementation.

Canonical utterance construction Given an utterance x and the KB, we construct a set of candidate logical forms $\mathcal{Z}_x$, and then for each $z \in \mathcal{Z}_x$ generate a small set of canonical natural language utterances $\mathcal{C}_z$. Our goal at this point is only to generate a manageable set of logical forms containing the correct one, and then generate an appropriate canonical utterance from it. This strategy is feasible in factoid QA where compositionality is low, and so the size of $\mathcal{Z}_x$ is limited (Section 4).

Paraphrasing We score the canonical utterances in $\mathcal{C}_z$ with respect to the input utterance x using a paraphrase model, which offers two advantages. First, the paraphrase model is decoupled from the KB, so we can train it from large text corpora. Second, natural language utterances often do not express predicates explicitly, e.g., the question *"What is Italy's money?"* expresses the binary predicate `CurrencyOf` with a possessive construction. Paraphrasing methods are well-suited for handling such text-to-text gaps. Our framework accommodates any paraphrasing method, and in this paper we propose an *association model* that learns to associate natural language phrases that co-occur frequently in a monolingual parallel corpus, combined with a *vector space model*, which learns to score the similarity between vector representations of natural language utterances (Section 5).

Model We define a discriminative log-linear model that places a probability distribution over pairs of logical forms and canonical utterances (c, z), given an utterance x:

$$p_\theta(c, z \mid x) = \frac{\exp\{\phi(x, c, z)^\top \theta\}}{\sum_{z' \in \mathcal{Z}_x, c' \in \mathcal{C}_z} \exp\{\phi(x, c', z')^\top \theta\}},$$

where $\theta \in \mathbb{R}^b$ is the vector of parameters to be learned, and $\phi(x, c, z)$ is a feature vector extracted from the input utterance x, the canonical utterance c, and the logical form z. Note that the candidate set of logical forms $\mathcal{Z}_x$ and canonical utterances $\mathcal{C}_x$ are constructed during the canonical utterance construction phase.

The model score decomposes into two terms:

$$\phi(x, c, z)^\top \theta = \phi_{\text{pr}}(x, c)^\top \theta_{\text{pr}} + \phi_{\text{lf}}(x, z)^\top \theta_{\text{lf}},$$

where the parameters θ_{pr} define the paraphrase model (Section 5), which is based on features extracted from text only (the input and canonical utterance). The parameters θ_{lf} correspond to semantic parsing features based on the logical form and input utterance, and are briefly described in this section.

Many existing paraphrase models introduce latent variables to describe the *derivation* of c from x, e.g., with transformations (Heilman and Smith, 2010; Stern and Dagan, 2011) or alignments (Haghighi et al., 2005; Das and Smith, 2009; Chang et al., 2010). However, we opt for a simpler paraphrase model without latent variables in the interest of efficiency.

Logical form features The parameters θ_{lf} correspond to the following features adopted from Berant et al. (2013). For a logical form z, we extract the size of its denotation $[\![z]\!]_\mathcal{K}$. We also add all binary predicates in z as features. Moreover, we extract a popularity feature for predicates based on the number of instances they have in $\mathcal{K}$. For Freebase entities, we extract a popularity feature based on the entity frequency in an entity linked subset of Reverb (Lin et al., 2012). Lastly, Freebase formulas have types (see Section 4), and we conjoin the type of z with the first word of x, to capture the correlation between a word (e.g., *"where"*) with the Freebase type (e.g., `Location`).

Learning As our training data consists of question-answer pairs (x_i, y_i), we maximize the log-likelihood of the correct answer. The probability of an answer y is obtained by marginalizing over canonical utterances c and logical forms z whose denotation is y. Formally, our objective function $\mathcal{O}(\theta)$ is as follows:

$$\mathcal{O}(\theta) = \sum_{i=1}^{n} \log p_\theta(y_i \mid x_i) - \lambda \|\theta\|_1,$$

$$p_\theta(y \mid x) = \sum_{z \in \mathcal{Z}_x : y = [\![z]\!]_\mathcal{K}} \sum_{c \in \mathcal{C}_z} p_\theta(c, z \mid x).$$

The strength λ of the L_1 regularizer is set based on cross-validation. We optimize the objective by initializing the parameters θ to zero and running AdaGrad (Duchi et al., 2010). We approximate the set of pairs of logical forms and canonical utterances with a beam of size 2,000.

4 Canonical utterance construction

We construct canonical utterances in two steps. Given an input utterance x, we first construct a set of logical forms $\mathcal{Z}_x$, and then generate canonical utterances from each $z \in \mathcal{Z}_x$. Both steps are performed with a small and simple set of deterministic rules, which suffices for our datasets, as they consist of factoid questions with a modest amount of compositional structure. We describe these rules below for completeness. Due to its soporific effect though, we advise the reader to skim it quickly.

Candidate logical forms We consider logical forms defined by a set of templates, summarized in Table 1. The basic template is a join of a binary and an entity, where a binary can either be one property $p.e$ (#1 in the table) or two properties $p_1.p_2.e$ (#2). To handle cases of events involving multiple arguments (e.g., *"Who did Brad Pitt play in Troy?"*), we introduce the template $p.(p_1.e_1 \sqcap p_2.e_2)$ (#3), where the main event is modified by more than one entity. Logical forms can be further modified by a unary "filter", e.g., the answer to *"What composers spoke French?"* is a set of composers, i.e., a subset of all people (#4). Lastly, we handle aggregation formulas for utterances such as *"How many teams are in the NCAA?"* (#5).

To construct candidate logical forms $\mathcal{Z}_x$ for a given utterance x, our strategy is to find an entity in x and grow the logical form from that entity. As we show later, this procedure actually produces a set with better coverage than constructing logical forms recursively from spans of x, as is done in traditional semantic parsing. Specifically, for every span of x, we take at most 10 entities whose Freebase descriptions approximately match the span. Then, we join each entity e with all type-compatible[1] binaries b, and add these logical forms to $\mathcal{Z}_x$ (#1 and #2).

To construct logical forms with multiple entities (#3) we do the following: For any logical form $z = p.p_1.e_1$, where p_1 has type signature $(t_1, *)$, we look for other entities e_2 that were matched in x. Then, we add the logical form $p.(p_1.e_1 \sqcap p_2.e_2)$, if there exists a binary p_2 with a compatible type signature (t_1, t_2), where t_2 is one of e_2's types. For example, for the logical form `Character.Actor.BradPitt`, if we match the entity `Troy` in x, we obtain `Character.(Actor.BradPitt ⊓ Film.Troy)`. We further modify logical forms by intersecting with a unary filter (#4): given a formula z with some Freebase type (e.g., `People`), we look at all Freebase sub-types t (e.g., `Composer`), and check whether one of their Freebase descriptions (e.g., *"composer"*) appears in x. If so, we add the formula `Type.`$t \sqcap z$ to $\mathcal{Z}_x$. Finally, we check whether x is an aggregation formula by identifying whether it starts with phrases such as *"how many"* or *"number of"* (#5).

[1]Entities in Freebase are associated with a set of types, and properties have a type signature (t_1, t_2) We use these types to compute an expected type t for any logical form z.

On WEBQUESTIONS, this results in 645 formulas per utterance on average. Clearly, we can increase the expressivity of this step by expanding the template set. For example, we could handle superlative utterances (*"What NBA player is tallest?"*) by adding a template with an `argmax` operator.

Utterance generation While mapping general language utterances to logical forms is hard, we observe that it is much easier to generate a *canonical* natural language utterances of our choice given a logical form. Table 2 summarizes the rules used to generate canonical utterances from the template $p.e$. Questions begin with a question word, are followed by the Freebase description of the expected answer type ($d(t)$), and followed by Freebase descriptions of the entity ($d(e)$) and binary ($d(p)$). To fill in auxiliary verbs, determiners, and prepositions, we parse the description $d(p)$ into one of NP, VP, PP, or NP VP. This determines the generation rule to be used.

Each Freebase property p has an explicit property p' equivalent to the reverse $\mathbf{R}[p]$ (e.g., `ContainedBy` and $\mathbf{R}$[`Contains`]). For each logical form z, we also generate using equivalent logical forms where p is replaced with $\mathbf{R}[p']$. Reversed formulas have different generation rules, since entities in these formulas are in the subject position rather than object position.

We generate the description $d(t)$ from the Freebase description of the type of z (this handles #4). For the template $p_1.p_2.e$ (#2), we have a similar set of rules, which depends on the syntax of $d(p_1)$ and $d(p_2)$ and is omitted for brevity. The template $p.(p_1.e_1 \sqcap p_2.e_2)$ (#3) is generated by appending the prepositional phrase `in` $d(e_2)$, e.g, *"What character is the character of Brad Pitt in Troy?"*. Lastly, we choose the question phrase *"How many"* for aggregation formulas (#5), and *"What"* for all other formulas.

We also generate canonical utterances using an alignment lexicon, released by Berant et al. (2013), which maps text phrases to Freebase binary predicates. For a binary predicate b mapped from text phrase $d(b)$, we generate the utterance

#	Template	Example	Question
1	$p.e$	`Directed.TopGun`	*Who directed Top Gun?*
2	$p_1.p_2.e$	`Employment.EmployerOf.SteveBalmer`	*Where does Steve Balmer work?*
3	$p.(p_1.e_1 \sqcap p_2.e_2)$	`Character.(Actor.BradPitt ⊓ Film.Troy)`	*Who did Brad Pitt play in Troy?*
4	$\texttt{Type}.t \sqcap z$	`Type.Composer ⊓ SpeakerOf.French`	*What composers spoke French?*
5	$\texttt{count}(z)$	`count(BoatDesigner.NatHerreshoff)`	*How many ships were designed by Nat Herreshoff?*

Table 1: Logical form templates, where p, p_1, p_2 are Freebase properties, e, e_1, e_2 are Freebase entities, t is a Freebase type, and z is a logical form.

	$d(p)$ **Categ.**	**Rule**	**Example**
$p.e$	NP	`WH` $d(t)$ `has` $d(e)$ `as` NP ?	*What **election contest** has **George Bush** as **winner?***
	VP	`WH` $d(t)$ `(AUX)` VP $d(e)$?	*What **radio station serves area New-York**?*
	PP	`WH` $d(t)$ PP $d(e)$?	*What **beer from region Argentina**?*
	NP VP	`WH` $d(t)$ VP `the` NP $d(e)$?	*What **mass transportation system served** the **area Berlin**?*
$\mathbf{R}(p).e$	NP	`WH` $d(t)$ `is the` NP `of` $d(e)$?	*What **location** is the **place of birth** of **Elvis Presley**?*
	VP	`WH` $d(t)$ `AUX` $d(e)$ VP ?	*What **film** is **Brazil featured in**?*
	PP	`WH` $d(t)$ $d(e)$ PP ?	*What **destination Spanish steps near travel destination**?*
	NP VP	`WH` NP `is` VP `by` $d(e)$?	*What **structure** is **designed** by Herod?*

Table 2: Generation rules for templates of the form $p.e$ and $\mathbf{R}[p].e$ based on the syntactic category of the property description. Freebase descriptions for the type, entity, and property are denoted by $d(t)$, $d(e)$ and $d(p)$ respectively. The surface form of the auxiliary `AUX` is determined by the POS tag of the verb inside the VP tree.

`WH` $d(t)$ $d(b)$ $d(e)$ `?`. On the WEBQUESTIONS dataset, we generate an average of 1,423 canonical utterances c per input utterance x. In Section 6, we show that an even simpler method of generating canonical utterances by concatenating Freebase descriptions hurts accuracy by only a modest amount.

5 Paraphrasing

Once the candidate set of logical forms paired with canonical utterances is constructed, our problem is reduced to scoring pairs (c, z) based on a paraphrase model. The NLP paraphrase literature is vast and ranges from simple methods employing surface features (Wan et al., 2006), through vector space models (Socher et al., 2011), to latent variable models (Das and Smith, 2009; Wang and Manning, 2010; Stern and Dagan, 2011).

In this paper, we focus on two paraphrase models that emphasize simplicity and efficiency. This is important since for each question-answer pair, we consider thousands of canonical utterances as potential paraphrases. In contrast, traditional paraphrase detection (Dolan et al., 2004) and Recognizing Textual Entailment (RTE) tasks (Dagan et al., 2013) consider examples consisting of only a single pair of candidate paraphrases.

Our paraphrase model decomposes into an *association model* and a *vector space model*:

$$\phi_{\text{pr}}(x, c)^\top \theta_{\text{pr}} = \phi_{\text{as}}(x, c)^\top \theta_{\text{as}} + \phi_{\text{vs}}(x, c)^\top \theta_{\text{vs}}.$$

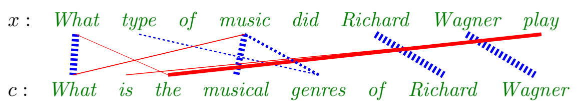
x : What type of music did Richard Wagner play
c : What is the musical genres of Richard Wagner

Figure 3: Token associations extracted for a paraphrase pair. Blue and dashed (red and solid) indicate positive (negative) score. Line width is proportional to the absolute value of the score.

5.1 Association model

The goal of the association model is to determine whether x and c contain phrases that are likely to be paraphrases. Given an utterance $x = \langle x_0, x_1, .., x_{n-1} \rangle$, we denote by $x_{i:j}$ the span from token i to token j. For each pair of utterances (x, c), we go through all spans of x and c and identify a set of pairs of potential paraphrases $(x_{i:j}, c_{i':j'})$, which we call *associations*. (We will describe how associations are identified shortly.) We then define features on each association; the weighted combination of these features yields a score. In this light, associations can be viewed as soft paraphrase rules. Figure 3 presents examples of associations extracted from a paraphrase pair and visualizes the learned scores. We can see that our model learns a positive score for associating *"type"* with *"genres"*, and a negative score for associating *"is"* with *"play"*.

We define associations in x and c primarily by looking up phrase pairs in a phrase table constructed using the PARALEX corpus (Fader et al., 2013). PARALEX is a large monolingual parallel

Category	Description
Assoc.	lemma($x_{i:j}$) $\wedge$ lemma($c_{i':j'}$) pos($x_{i:j}$) $\wedge$ pos($c_{i':j'}$) lemma($x_{i:j}$) = lemma($c_{i':j'}$)? pos($x_{i:j}$) = pos($c_{i':j'}$)? lemma($x_{i:j}$) and lemma($c_{i':j'}$) are synonyms? lemma($x_{i:j}$) and lemma($c_{i':j'}$) are derivations?
Deletions	Deleted lemma and POS tag

Table 3: Full feature set in the association model. $x_{i:j}$ and $c_{i':j'}$ denote spans from x and c. pos($x_{i:j}$) and lemma($x_{i:j}$) denote the POS tag and lemma sequence of $x_{i:j}$.

corpora, containing 18 million pairs of question paraphrases from `wikianswers.com`, which were tagged as having the same meaning by users. PARALEX is suitable for our needs since it focuses on *question* paraphrases. For example, the phrase *"do for a living"* occurs mostly in questions, and we can extract associations for this phrase from PARALEX. Paraphrase pairs in PARALEX are word-aligned using standard machine translation methods. We use the word alignments to construct a phrase table by applying the consistent phrase pair heuristic (Och and Ney, 2004) to all 5-grams. This results in a phrase table with approximately 1.3 million phrase pairs. We let $\mathcal{A}$ denote this set of mined candidate associations.

For a pair (x, c), we also consider as candidate associations the set $\mathcal{B}$ (represented implicitly), which contains token pairs $(x_i, c_{i'})$ such that x_i and $c_{i'}$ share the same lemma, the same POS tag, or are linked through a *derivation* link on WordNet (Fellbaum, 1998). This allows us to learn paraphrases for words that appear in our datasets but are not covered by the phrase table, and to handle nominalizations for phrase pairs such as *"Who designed the game of life?"* and *"What game designer is the designer of the game of life?"*.

Our model goes over all possible spans of x and c and constructs all possible associations from $\mathcal{A}$ and $\mathcal{B}$. This results in many poor associations (e.g., *"play"* and *"the"*), but as illustrated in Figure 3, we learn weights that discriminate good from bad associations. Table 3 specifies the full set of features. Note that unlike standard paraphrase detection and RTE systems, we use lexicalized features, firing approximately 400,000 features on WEBQUESTIONS. By extracting POS features, we obtain soft syntactic rules, e.g., the feature "JJ N $\wedge$ N" indicates that omitting adjectives before nouns is possible. Once associations are constructed, we mark tokens in x and c that were not part of any association, and extract deletion features for their lemmas and POS tags. Thus, we learn that deleting pronouns is acceptable, while deleting nouns is not.

To summarize, the association model links phrases of two utterances in multiple overlapping ways. During training, the model learns which associations are characteristic of paraphrases and which are not.

5.2 Vector space model

The association model relies on having a good set of candidate associations, but mining associations suffers from coverage issues. We now introduce a vector space (VS) model, which assigns a vector representation for each utterance, and learns a scoring function that ranks paraphrase candidates.

We start by constructing vector representations of words. We run the WORD2VEC tool (Mikolov et al., 2013) on lower-cased Wikipedia text (1.59 billion tokens), using the CBOW model with a window of 5 and hierarchical softmax. We also experiment with publicly released word embeddings (Huang et al., 2012), which were trained using both local and global context. Both result in k-dimensional vectors ($k = 50$). Next, we construct a vector $v_x \in \mathbb{R}^k$ for each utterance x by simply averaging the vectors of all content words (nouns, verbs, and adjectives) in x.

We can now estimate a paraphrase score for two utterances x and c via a weighted combination of the components of the vector representations:

$$v_x^\top W v_c = \sum_{i,j=1}^{k} w_{ij} v_{x,i} v_{c,j}$$

where $W \in \mathbb{R}^{k\times k}$ is a parameter matrix. In terms of our earlier notation, we have $\theta_{\text{vs}} = \text{vec}(W)$ and $\phi_{\text{vs}}(x, c) = \text{vec}(v_x v_c^\top)$, where vec$(\cdot)$ unrolls a matrix into a vector. In Section 6, we experiment with W equal to the identity matrix, constraining W to be diagonal, and learning a full W matrix.

The VS model can identify correct paraphrases in cases where it is hard to directly associate phrases from x and c. For example, the answer to *"Where is made Kia car?"* (from WEBQUESTIONS), is given by the canonical utterance *"What city is Kia motors a headquarters of?"*. The association model does not associate *"made"* and *"headquarters"*, but the VS model is able to determine that these utterances are semantically related. In other cases, the VS model cannot distinguish correct paraphrases from incorrect ones. For

Dataset	# examples	# word types
FREE917	917	2,036
WEBQUESTIONS	5,810	4,525

Table 4: Statistics on WEBQUESTIONS and FREE917.

example, the association model identifies that the paraphrase for *"What type of music did Richard Wagner Play?"* is *"What is the musical genres of Richard Wagner?"*, by relating phrases such as *"type of music"* and *"musical genres"*. The VS model ranks the canonical utterance *"What composition has Richard Wagner as lyricist?"* higher, as this utterance is also in the music domain. Thus, we combine the two models to benefit from their complementary nature.

In summary, while the association model aligns particular phrases to one another, the vector space model provides a soft vector-based representation for utterances.

6 Empirical evaluation

In this section, we evaluate our system on WEBQUESTIONS and FREE917. After describing the setup (Section 6.1), we present our main empirical results and analyze the components of the system (Section 6.2).

6.1 Setup

We use the WEBQUESTIONS dataset (Berant et al., 2013), which contains 5,810 question-answer pairs. This dataset was created by crawling questions through the Google Suggest API, and then obtaining answers using Amazon Mechanical Turk. We use the original train-test split, and divide the training set into 3 random 80%–20% splits for development. This dataset is characterized by questions that are commonly asked on the web (and are not necessarily grammatical), such as *"What character did Natalie Portman play in Star Wars?"* and *"What kind of money to take to Bahamas?"*.

The FREE917 dataset contains 917 questions, authored by two annotators and annotated with logical forms. This dataset contains questions on rarer topics (for example, *"What is the engine in a 2010 Ferrari California?"* and *"What was the cover price of the X-men Issue 1?"*), but the phrasing of questions tends to be more rigid compared to WEBQUESTIONS. Table 4 provides some statistics on the two datasets. Following Cai and Yates (2013), we hold out 30% of the data for the final test, and perform 3 random 80%-20% splits of the training set for development. Since we train from question-answer pairs, we collect answers by executing the gold logical forms against Freebase.

We execute λ-DCS queries by converting them into SPARQL and executing them against a copy of Freebase using the Virtuoso database engine. We evaluate our system with accuracy, that is, the proportion of questions we answer correctly. We run all questions through the Stanford CoreNLP pipeline (Toutanova and Manning, 2003; Finkel et al., 2005; Klein and Manning, 2003).

We tuned the L_1 regularization strength, developed features, and ran analysis experiments on the development set (averaging across random splits). On WEBQUESTIONS, without L_1 regularization, the number of non-zero features was 360K; L_1 regularization brings it down to 17K.

6.2 Results

We compare our system to Cai and Yates (2013) (CY13), Berant et al. (2013) (BCFL13), and Kwiatkowski et al. (2013) (KCAZ13). For BCFL13, we obtained results using the SEMPRE package[2] and running Berant et al. (2013)'s system on the datasets.

Table 5 presents results on the test set. We achieve a substantial relative improvement of 12% in accuracy on WEBQUESTIONS, and match the best results on FREE917. Interestingly, our system gets an *oracle accuracy* of 63% on WEBQUESTIONS compared to 48% obtained by BCFL13, where the oracle accuracy is the fraction of questions for which at least one logical form in the candidate set produced by the system is correct. This demonstrates that our method for constructing candidate logical forms is reasonable. To further examine this, we ran BCFL13 on the development set, allowing it to use only predicates from logical forms suggested by our logical form construction step. This improved oracle accuracy on the development set to 64.5%, but accuracy was 32.2%. This shows that the improvement in accuracy should not be attributed only to better logical form generation, but also to the paraphrase model.

We now perform more extensive analysis of our system's components and compare it to various baselines.

Component ablation We ablate the association model, the VS model, and the entire paraphrase

[2]http://www-nlp.stanford.edu/software/sempre/

	FREE917	WEBQUESTIONS
CY13	59.0	–
BCFL13	62.0	35.7
KCAZ13	68.0	–
This work	**68.5**	**39.9**

Table 5: Results on the test set.

	FREE917	WEBQUESTIONS
Our system	**73.9**	**41.2**
–VSM	71.0	40.5
–ASSOCIATION	52.7	35.3
–PARAPHRASE	31.8	21.3
SIMPLEGEN	73.4	40.4
Full matrix	52.7	35.3
Diagonal	50.4	30.6
Identity	50.7	30.4
JACCARD	69.7	31.3
EDIT	40.8	24.8
WDDC06	71.0	29.8

Table 6: Results for ablations and baselines on development set.

Full	*do*	*people*	*czech*	*republic*	*speak*
offical	0.7	8.09	15.34	21.62	24.44
language	3.86	-3.13	7.81	2.58	14.74
czech	0.67	16.55			2.76
republic	-8.71	12.47			-10.75

Diagonal	*do*	*people*	*czech*	*republic*	*speak*
offical	2.31	-0.72	1.88	0.27	-0.49
language	0.27	4.72	11.51	12.33	11
czech	1.4	8.13			5.21
republic	-0.16	6.72			9.69

Identity	*do*	*people*	*czech*	*republic*	*speak*
offical	2.26	-1.41	0.89	0.07	-0.58
language	0.62	4.19	11.91	10.78	12.7
czech	2.88	7.31			5.42
republic	-1.82	4.34			9.44

Figure 4: Values of the paraphrase score $v_{x_i}^\top W v_{c_{i'}}$ for all content word tokens x_i and $c_{i'}$, where W is an arbitrary full matrix, a diagonal matrix, or the identity matrix. We omit scores for the words *"czech"* and *"republic"* since they appear in all canonical utterances for this example.

model (using only logical form features). Table 5 shows that our full system obtains highest accuracy, and that removing the association model results in a much larger degradation compared to removing the VS model.

Utterance generation Our system generates relatively natural utterances from logical forms using simple rules based on Freebase descriptions (Section 4). We now consider simply concatenating Freebase descriptions. For example, the logical form **R**`[PlaceOfBirth].ElvisPresley` would generate the utterance *"What location Elvis Presley place of birth?"*. Row SIMPLEGEN in Table 6 demonstrates that we still get good results in this setup. This is expected given that our paraphrase models are not sensitive to the syntactic structure of the generated utterance.

VS model Our system learns parameters for a full W matrix. We now examine results when learning parameters for a full matrix W, a diagonal matrix W, and when setting W to be the identity matrix. Table 6 (third section) illustrates that learning a full matrix substantially improves accuracy. Figure 4 gives an example for a correct paraphrase pair, where the full matrix model boosts the overall model score. Note that the full matrix assigns a high score for the phrases *"official language"* and *"speak"* compared to the simpler models, but other pairs are less interpretable.

Baselines We also compared our system to the following implemented baselines:

- JACCARD: We compute the Jaccard score between the tokens of x and c and define $\phi_{\text{pr}}(x, c)$ to be this single feature.
- EDIT: We compute the token edit distance between x and c and define $\phi_{\text{pr}}(x, c)$ to be this single feature.
- WDDC06: We re-implement 13 features from Wan et al. (2006), who obtained close to state-of-the-art performance on the Microsoft Research paraphrase corpus.[3]

Table 6 demonstrates that we improve performance over all baselines. Interestingly, JACCARD and WDDC06 obtain reasonable performance on FREE917 but perform much worse on WEBQUESTIONS. We surmise this is because questions in FREE917 were generated by annotators prompted by Freebase facts, whereas questions in WEBQUESTIONS originated independently of Freebase. Thus, word choice in FREE917 is often close to the generated Freebase descriptions, allowing simple baselines to perform well.

Error analysis We sampled examples from the development set to examine the main reasons PARASEMPRE makes errors. We notice that in many cases the paraphrase model can be further improved. For example, PARASEMPRE suggests

[3]We implement all features that do not require dependency parsing.

that the best paraphrase for *"What company did Henry Ford work for?"* is *"What written work novel by Henry Ford?"* rather than *"The employer of Henry Ford"*, due to the exact match of the word *"work"*. Another example is the question *"Where is the Nascar hall of fame?"*, where PARASEMPRE suggests that *"What hall of fame discipline has Nascar hall of fame as halls of fame?"* is the best canonical utterance. This is because our simple model allows to associate *"hall of fame"* with the canonical utterance three times. Entity recognition also accounts for many errors, e.g., the entity chosen in *"where was the gallipoli campaign waged?"* is `Galipoli` and not `GalipoliCampaign`. Last, PARASEMPRE does not handle temporal information, which causes errors in questions like *"Where did Harriet Tubman live after the civil war?"*

7 Discussion

In this work, we approach the problem of semantic parsing from a paraphrasing viewpoint. A fundamental motivation and long standing goal of the paraphrasing and RTE communities has been to cast various semantic applications as paraphrasing/textual entailment (Dagan et al., 2013). While it has been shown that paraphrasing methods are useful for question answering (Harabagiu and Hickl, 2006) and relation extraction (Romano et al., 2006), this is, to the best of our knowledge, the first paper to perform semantic parsing through paraphrasing. Our paraphrase model emphasizes simplicity and efficiency, but the framework is agnostic to the internals of the paraphrase method.

On the semantic parsing side, our work is most related to Kwiatkowski et al. (2013). The main challenge in semantic parsing is coping with the *mismatch* between language and the KB. In both Kwiatkowski et al. (2013) and this work, an intermediate representation is employed to handle the mismatch, but while they use a logical representation, we opt for a text-based one. Our choice allows us to benefit from the parallel monolingual corpus PARALEX and from word vectors trained on Wikipedia. We believe that our approach is particularly suitable for scenarios such as factoid question answering, where the space of logical forms is somewhat constrained and a few generation rules suffice to reduce the problem to paraphrasing.

Our work is also related to Fader et al. (2013), who presented a paraphrase-driven question answering system. One can view this work as a generalization of Fader et al. along three dimensions. First, Fader et al. use a KB over natural language extractions rather than a formal KB and so querying the KB does not require a generation step – they paraphrase questions to KB entries directly. Second, they suggest a particular paraphrasing method that maps a test question to a question for which the answer is already known in a single step. We propose a general paraphrasing framework and instantiate it with two paraphrase models. Lastly, Fader et al. handle queries with only one property and entity whereas we generalize to more types of logical forms.

Since our generated questions are passed to a paraphrase model, we took a very simple approach, mostly ensuring that we preserved the semantics of the utterance without striving for the most fluent realization. Research on generation (Dale et al., 2003; Reiter et al., 2005; Turner et al., 2009; Piwek and Boyer, 2012) typically focuses on generating natural utterances for human consumption, where fluency is important.

In conclusion, the main contribution of this paper is a novel approach for semantic parsing based on a simple generation procedure and a paraphrase model. We achieve state-of-the-art results on two recently released datasets. We believe that our approach opens a window of opportunity for learning semantic parsers from raw text not necessarily related to the target KB. With more sophisticated generation and paraphrase, we hope to tackle compositionally richer utterances.

Acknowledgments

We thank Kai Sheng Tai for performing the error analysis. Stanford University gratefully acknowledges the support of the Defense Advanced Research Projects Agency (DARPA) Deep Exploration and Filtering of Text (DEFT) Program under Air Force Research Laboratory (AFRL) contract no. FA8750-13-2-0040. Any opinions, findings, and conclusion or recommendations expressed in this material are those of the authors and do not necessarily reflect the view of the DARPA, AFRL, or the US government. The second author is supported by a Google Faculty Research Award.

References

J. Berant, A. Chou, R. Frostig, and P. Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Q. Cai and A. Yates. 2013. Large-scale semantic parsing via schema matching and lexicon extension. In *Association for Computational Linguistics (ACL)*.

M. Chang, D. Goldwasser, D. Roth, and V. Srikumar. 2010. Discriminative learning over constrained latent representations. In *North American Association for Computational Linguistics (NAACL)*.

I. Dagan, D. Roth, M. Sammons, and F. M. Zanzotto. 2013. *Recognizing Textual Entailment: Models and Applications*. Morgan and Claypool Publishers.

R. Dale, S. Geldof, and J. Prost. 2003. Coral: using natural language generation for navigational assistance. In *Australasian computer science conference*, pages 35–44.

D. Das and N. A. Smith. 2009. Paraphrase identification as probabilistic quasi-synchronous recognition. In *Association for Computational Linguistics (ACL)*, pages 468–476.

B. Dolan, C. Quirk, and C. Brockett. 2004. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *International Conference on Computational Linguistics (COLING)*.

J. Duchi, E. Hazan, and Y. Singer. 2010. Adaptive subgradient methods for online learning and stochastic optimization. In *Conference on Learning Theory (COLT)*.

A. Fader, S. Soderland, and O. Etzioni. 2011. Identifying relations for open information extraction. In *Empirical Methods in Natural Language Processing (EMNLP)*.

A. Fader, L. Zettlemoyer, and O. Etzioni. 2013. Paraphrase-driven learning for open question answering. In *Association for Computational Linguistics (ACL)*.

C. Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

J. R. Finkel, T. Grenager, and C. Manning. 2005. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Association for Computational Linguistics (ACL)*, pages 363–370.

Google. 2013. Freebase data dumps (2013-06-09). `https://developers.google.com/freebase/data`.

A. Haghighi, A. Y. Ng, and C. D. Manning. 2005. Robust textual inference via graph matching. In *Empirical Methods in Natural Language Processing (EMNLP)*.

S. Harabagiu and A. Hickl. 2006. Methods for using textual entailment in open-domain question answering. In *Association for Computational Linguistics (ACL)*.

M. Heilman and N. A. Smith. 2010. Tree edit models for recognizing textual entailments, paraphrases, and answers to questions. In *Human Language Technology and North American Association for Computational Linguistics (HLT/NAACL)*, pages 1011–1019.

E. H. Huang, R. Socher, C. D. Manning, and A. Y. Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Association for Computational Linguistics (ACL)*.

D. Klein and C. Manning. 2003. Accurate unlexicalized parsing. In *Association for Computational Linguistics (ACL)*, pages 423–430.

T. Kwiatkowski, L. Zettlemoyer, S. Goldwater, and M. Steedman. 2010. Inducing probabilistic CCG grammars from logical form with higher-order unification. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1223–1233.

T. Kwiatkowski, E. Choi, Y. Artzi, and L. Zettlemoyer. 2013. Scaling semantic parsers with on-the-fly ontology matching. In *Empirical Methods in Natural Language Processing (EMNLP)*.

P. Liang. 2013. Lambda dependency-based compositional semantics. Technical report, ArXiv.

T. Lin, Mausam, and O. Etzioni. 2012. Entity linking at web scale. In *Knowledge Extraction Workshop (AKBC-WEKEX)*.

T. Mikolov, K. Chen, G. Corrado, and Jeffrey. 2013. Efficient estimation of word representations in vector space. Technical report, ArXiv.

F. J. Och and H. Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30:417–449.

P. Piwek and K. E. Boyer. 2012. Varieties of question generation: Introduction to this special issue. *Dialogue and Discourse*, 3:1–9.

E. Reiter, S. Sripada, J. Hunter, J. Yu, and I. Davy. 2005. Choosing words in computer-generated weather forecasts. *Artificial Intelligence*, 167:137–169.

L. Romano, M. kouylekov, I. Szpektor, I. Dagan, and A. Lavelli. 2006. Investigating a generic paraphrase-based approach for relation extraction. In *Proceedings of ECAL*.

R. Socher, E. H. Huang, J. Pennin, C. D. Manning, and A. Ng. 2011. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Advances in Neural Information Processing Systems (NIPS)*, pages 801–809.

A. Stern and I. Dagan. 2011. A confidence model for syntactically-motivated entailment proofs. In *Recent Advances in Natural Language Processing*, pages 455–462.

K. Toutanova and C. D. Manning. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Human Language Technology and North American Association for Computational Linguistics (HLT/NAACL)*.

R. Turner, Y. Sripada, and E. Reiter. 2009. Generating approximate geographic descriptions. In *European Workshop on Natural Language Generation*, pages 42–49.

S. Wan, M. Dras, R. Dale, and C. Paris. 2006. Using dependency-based features to take the "para-farce" out of paraphrase. In *Australasian Language Technology Workshop*.

M. Wang and C. D. Manning. 2010. Probabilistic tree-edit models with structured latent variables for textual entailment and question answering. In *The International Conference on Computational Linguistics*, pages 1164–1172.

Y. W. Wong and R. J. Mooney. 2007. Learning synchronous grammars for semantic parsing with lambda calculus. In *Association for Computational Linguistics (ACL)*, pages 960–967.

M. Zelle and R. J. Mooney. 1996. Learning to parse database queries using inductive logic proramming. In *Association for the Advancement of Artificial Intelligence (AAAI)*, pages 1050–1055.

L. S. Zettlemoyer and M. Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *Uncertainty in Artificial Intelligence (UAI)*, pages 658–666.