Information Extraction over Structured Data: Question Answering with Freebase

Xuchen Yao [1] and **Benjamin Van Durme** [1,2]
[1]Center for Language and Speech Processing
[2]Human Language Technology Center of Excellence
Johns Hopkins University
Baltimore, MD, USA

Abstract

Answering natural language questions using the Freebase knowledge base has recently been explored as a platform for advancing the state of the art in open domain semantic parsing. Those efforts map questions to sophisticated meaning representations that are then attempted to be matched against viable answer candidates in the knowledge base. Here we show that relatively modest information extraction techniques, when paired with a web-scale corpus, can outperform these sophisticated approaches by roughly 34% relative gain.

1 Introduction

Question answering (QA) from a knowledge base (KB) has a long history within natural language processing, going back to the 1960s and 1970s, with systems such as `Baseball` (Green Jr et al., 1961) and `Lunar` (Woods, 1977). These systems were limited to closed-domains due to a lack of knowledge resources, computing power, and ability to robustly understand natural language. With the recent growth in KBs such as `DBPedia` (Auer et al., 2007), `Freebase` (Bollacker et al., 2008) and `Yago2` (Hoffart et al., 2011), it has become more practical to consider answering questions across wider domains, with commercial systems including `Google Now`, based on Google's `Knowledge Graph`, and `Facebook Graph Search`, based on social network connections.

The AI community has tended to approach this problem with a focus on first understanding the intent of the question, via shallow or deep forms of semantic parsing (c.f. §3 for a discussion). Typically questions are converted into some meaning representation (e.g., the lambda calculus), then mapped to database queries. Performance is thus bounded by the accuracy of the original semantic parsing, and the well-formedness of resultant database queries.[1]

The Information Extraction (IE) community approaches QA differently: first performing relatively coarse information retrieval as a way to triage the set of possible answer candidates, and only then attempting to perform deeper analysis.

Researchers in semantic parsing have recently explored QA over Freebase as a way of moving beyond closed domains such as GeoQuery (Tang and Mooney, 2001). While making semantic parsing more robust is a laudable goal, here we provide a more rigorous IE baseline against which those efforts should be compared: we show that "traditional" IE methodology can significantly outperform prior state-of-the-art as reported in the semantic parsing literature, with a relative gain of 34% F_1 as compared to Berant et al. (2013).

2 Approach

We will view a KB as an interlinked collection of "topics". When given a question about one or several topics, we can select a "view" of the KB concerning only involved topics, then inspect every related node within a few hops of relations to the topic node in order to extract the answer. We call such a view a *topic graph* and assume answers can be found within the graph. We aim to maximally automate the answer extraction process, by massively combining discriminative features for both the question and the topic graph. With a high performance learner we have found that a system with millions of features can be trained within hours, leading to intuitive, human interpretable features. For example, we learn that given a question concerning money, such as: what money is used in

[1]As an example, 50% of errors of the CCG-backed (Kwiatkowski et al., 2013) system were contributed by parsing or structural matching failure.

ukraine, the expected answer type is likely currency. We formalize this approach in §4.

One challenge for natural language querying against a KB is the relative informality of queries as compared to the grammar of a KB. For example, for the question: who cheated on celebrity A, answers can be retrieved via the Freebase relation celebrity.infidelity.participant, but the connection between the phrase cheated on and the formal KB relation is not explicit. To alleviate this problem, the best attempt so far is to map from `ReVerb` (Fader et al., 2011) predicate-argument triples to Freebase relation triples (Cai and Yates, 2013; Berant et al., 2013). Note that to boost precision, `ReVerb` has already pruned down less frequent or credible triples, yielding not as much coverage as its text source, `ClueWeb`. Here we instead directly mine relation mappings from `ClueWeb` and show that both direct relation mapping precision and indirect QA F_1 improve by a large margin. Details in §5.

Finally, we tested our system, jacana-freebase,[2] on a realistic dataset generously contributed by Berant et al. (2013), who collected thousands of commonly asked questions by crawling the `Google Suggest` service. Our method achieves state-of-the-art performance with F_1 at 42.0%, a 34% relative increase from the previous F_1 of 31.4%.

3 Background

QA from a KB faces two prominent challenges: model and data. The model challenge involves finding the best meaning representation for the question, converting it into a query and executing the query on the KB. Most work approaches this via the bridge of various intermediate representations, including combinatory categorial grammar (Zettlemoyer and Collins, 2005, 2007, 2009; Kwiatkowski et al., 2010, 2011, 2013), synchronous context-free grammars (Wong and Mooney, 2007), dependency trees (Liang et al., 2011; Berant et al., 2013), string kernels (Kate and Mooney, 2006; Chen and Mooney, 2011), and tree transducers (Jones et al., 2012). These works successfully showed their effectiveness in QA, despite the fact that most of them require hand-labeled logic annotations. More recent research started to minimize this direct supervision by using latent meaning representations (Berant et al., 2013; Kwiatkowski et al., 2013) or distant supervision (Krishnamurthy and Mitchell, 2012).

We instead attack the problem of QA from a KB from an IE perspective: we learn directly the pattern of QA pairs, represented by the dependency parse of questions and the Freebase structure of answer candidates, without the use of intermediate, general purpose meaning representations.

The data challenge is more formally framed as ontology or (textual) schema matching (Hobbs, 1985; Rahm and Bernstein, 2001; Euzenat and Shvaiko, 2007): matching structure of two ontologies/databases or (in extension) mapping between KB relations and NL text. In terms of the latter, Cai and Yates (2013) and Berant et al. (2013) applied pattern matching and relation intersection between Freebase relations and predicate-argument triples from the `ReVerb` OpenIE system (Fader et al., 2011). Kwiatkowski et al. (2013) expanded their CCG lexicon with Wiktionary word tags towards more domain independence. Fader et al. (2013) learned question paraphrases from aligning multiple questions with the same answers generated by `WikiAnswers`. The key factor to their success is to have a huge text source. Our work pushes the data challenge to the limit by mining directly from `ClueWeb`, a 5TB collection of web data.

Finally, the KB community has developed other means for QA without semantic parsing (Lopez et al., 2005; Frank et al., 2007; Unger et al., 2012; Yahya et al., 2012; Shekarpour et al., 2013). Most of these work executed SPARQL queries on interlinked data represented by RDF (Resource Description Framework) triples, or simply performed triple matching. Heuristics and manual templates were also commonly used (Chu-Carroll et al., 2012). We propose instead to learn discriminative features from the data with shallow question analysis. The final system captures intuitive patterns of QA pairs automatically.

4 Graph Features

Our model is inspired by an intuition on how everyday people search for answers. If you asked someone: what is the name of justin bieber brother,[3] and gave them access to Freebase, that person might first determine that the question

[2] `https://code.google.com/p/jacana`

[3] All examples used in this paper come from the training data crawled from `Google Suggest`. They are lowercased and some contain typos.

is about Justin Bieber (or his brother), go to Justin Bieber's Freebase page, and search for his brother's name. Unfortunately Freebase does not contain an exact relation called brother, but instead sibling. Thus further inference (i.e., brother $\leftrightarrow$ male sibling) has to be made. In the following we describe how we represent this process.

4.1 Question Graph

In answering our example query a person might take into consideration multiple constraints. With regards to the question, we know we are looking for the name of a person based on the following:

- the dependency relation nsubj(what, name) and prep_of(name, brother) indicates that the question seeks the information of a name;[4]
- the dependency relation prep_of(name, brother) indicates that the name is about a brother (but we do not know whether it is a person name yet);
- the dependency relation nn(brother, bieber) and the facts that, (i) Bieber is a person and (ii) a person's brother should also be a person, indicate that the name is about a person.

This motivates the design of dependency-based features. We show one example in Figure 1(a), left side. The following linguistic information is of interest:

- question word (*qword*), such as what/who/how many. We use a list of 9 common qwords. [5]
- question focus (*qfocus*), a cue of expected answer types, such as name/money/time. We keep our analysis simple and do not use a question classifier, but simply extract the noun dependent of qword as qfocus.
- question verb (*qverb*), such as is/play/take, extracted from the main verb of the question. Question verbs are also good hints of answer types. For instance, play is likely to be followed by an instrument, a movie or a sports team.
- question topic (*qtopic*). The topic of the question helps us find relevant Freebase pages. We simply apply a named entity recognizer to find the question topic. Note that there can be more than one topic in the question.

Then we convert the dependency parse into a more generic question graph, in the following steps:

[4] We use the Stanford collapsed dependency form.

[5] who, when, what, where, how, which, why, whom, whose.

1. if a node was tagged with a question feature, then replace this node with its question feature, e.g., what $\rightarrow$ qword=what;
2. (special case) if a qtopic node was tagged as a named entity, then replace this node with its its named entity form, e.g., bieber $\rightarrow$ qtopic=person;
3. drop any leaf node that is a determiner, preposition or punctuation.

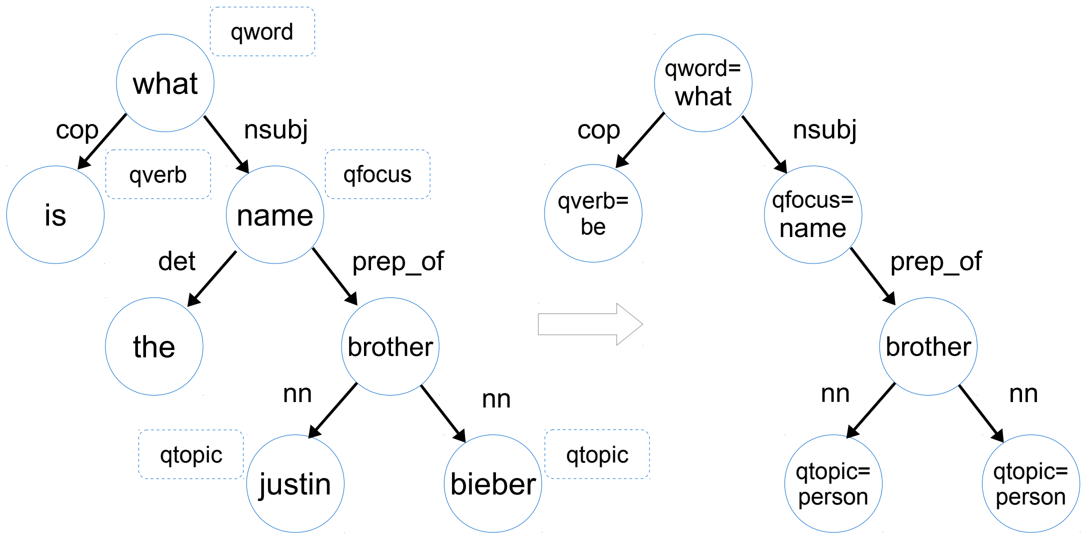
The converted graph is shown in Figure 1(a), right side. We call this a *question feature graph*, with every node and relation a potential feature for this question. Then features are extracted in the following form: with s the source and t the target node, for every edge $e(s,t)$ in the graph, extract s, t, $s \mid t$ and $s \mid e \mid t$ as features. For the edge, prep_of(qfocus=name, brother), this would mean the following features: qfocus=name, brother, qfocus=name|brother, and qfocus=name|prep_of|brother.

We show with examples why these features make sense later in §6 Table 6. Furthermore, the reason that we have kept some lexical features, such as brother, is that we hope to learn from training a high correlation between brother and some Freebase relations and properties (such as sibling and male) if we do not possess an external resource to help us identify such a correlation.

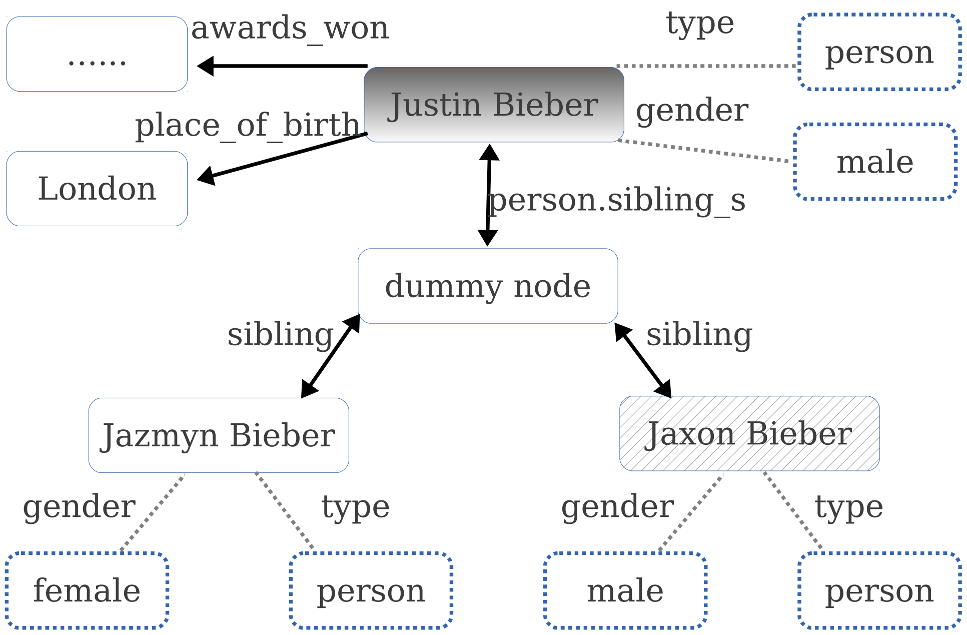
4.2 Freebase Topic Graph

Given a topic, we selectively roll out the Freebase graph by choosing those nodes within a few hops of relationship to the *topic node*, and form a *topic graph*. Besides incoming and/or outgoing relationships, nodes also have *properties*: a string that describes the attribute of a node, for instance, node type, gender or height (for a person). One major difference between relations and properties is that both arguments of a relation are nodes, while only one argument of a property is a node, the other a string. Arguments of relations are usually interconnected, e.g., London can be the place_of_birth for Justin Bieber, or capital_of the UK. Arguments of properties are attributes that are only "attached" to certain nodes and have no outgoing edges. Figure 1(b) shows an example.

Both relationship and property of a node are important to identifying the answer. They connect the nodes with the question and describe some unique characteristics. For instance, without the properties type:person and gender:male,



(a) Dependence parse with annotated question features in dashed boxes (left) and converted feature graph (right) with only relevant and general information about the original question kept. Note that the left is a real but incorrect parse.



(b) A view of Freebase graph on the Justin Bieber topic with nodes in solid boxes and properties in dashed boxes. The hatching node, Jaxon Bieber, is the answer. Freebase uses a dummy parent node for a list of nodes with the same relation.

Figure 1: Dependency parse and excerpted Freebase topic graph on the question what is the name of justin bieber brother.

we would not have known the node Jaxon Bieber represents a male person. These properties, along with the sibling relationship to the topic node, are important cues for answering the question. Thus for the Freebase graph, we use relations (with directions) and properties as features for each node.

Additionally, we have analyzed how Freebase relations map back to the question. Some of the mapping can be simply detected as paraphrasing or lexical overlap. For example, the person.parents relationship helps answering questions about parenthood. However, most Freebase relations are framed in a way that is not commonly addressed in natural language questions. For instance, for common celebrity gossip questions like who cheated on celebrity A, it is hard for a system to find the Freebase relation celebrity.infidelity.participant as the target relation if it had not observed this pattern in training.

Thus assuming there is an alignment model that is able to tell how likely one relation maps to the original question, we add extra alignment-based features for the incoming and outgoing relation of each node. Specifically, for each relation rel in a topic graph, we compute $P(rel \mid question)$ to rank the relations. Finally the ranking (e.g., top 1/2/5/10/100 and beyond) of each relation is used as features instead of a pure probability. We describe such an alignment model in § 5.

4.3 Feature Production

We combine question features and Freebase features (per node) by doing a pairwise concatenation. In this way we hope to capture the association between question patterns and answer nodes. For instance, in a loglinear model setting, we expect to learn a high feature weight for features like:

qfocus=money|node_type=currency

and a very low weight for:

qfocus=money|node_type=person.

This combination greatly enlarges the total number of features, but owing to progress in large-scale machine learning such feature spaces are less of a concern than they once were (concrete numbers in § 6 Model Tuning).

5 Relation Mapping

In this section we describe a "translation" table between Freebase relations and NL words was built.

5.1 Formula

The objective is to find the most likely relation a question prompts. For instance, for the question who is the father of King George VI, the most likely relation we look for is people.person.parents. To put it more formally, given a question Q of a word vector $\mathbf{w}$, we want to find out the relation R that maximizes the probability $P(R \mid Q)$.

More interestingly, for the question who is the father of the Periodic Table, the actual relation that encodes its original meaning is law.invention.inventor, rather than people.person.parents. This simple example points out that *every* part of the question could change what the question inquires eventually. Thus we need to count for *each* word w in Q. Due to the bias and incompleteness of any data source, we approximate the true probability of P with $\tilde{P}$ under our specific model. For the simplicity of computation, we assume conditional independence between words and apply Naive Bayes:

$$\begin{aligned}\tilde{P}(R \mid Q) &\propto \tilde{P}(Q \mid R)\tilde{P}(R)\\ &\approx \tilde{P}(\mathbf{w} \mid R)\tilde{P}(R)\\ &\approx \prod_w \tilde{P}(w \mid R)\tilde{P}(R)\end{aligned}$$

where $\tilde{P}(R)$ is the prior probability of a relation R and $\tilde{P}(w \mid R)$ is the conditional probability of word w given R.

It is possible that we do not observe a certain relation R when computing the above equation. In this case we back off to the "sub-relations": a relation R is a concatenation of a series of sub-relations $R = \mathbf{r} = r_1.r_2.r_3.\ldots$. For instance, the sub-relations of people.person.parents are people, person, and parents. Again, we assume conditional independence between sub-relations and apply Naive Bayes:

$$\begin{aligned}\tilde{P}_{\text{backoff}}(R \mid Q) &\approx \tilde{P}(\mathbf{r} \mid Q)\\ &\approx \prod_r \tilde{P}(r \mid Q)\\ &\propto \prod_r \tilde{P}(Q \mid r)\tilde{P}(r)\\ &\approx \prod_r \prod_w \tilde{P}(w \mid r)\tilde{P}(r)\end{aligned}$$

One other reason that we estimated $\tilde{P}(w \mid r)$ and $\tilde{P}(r)$ for sub-relations is that Freebase relations share some common structures in between them. For instance, both people.person.parents and fictional_universe.fictional_character.parents indicate the parent relationship but the latter is much less commonly annotated. We hope that the shared sub-relation, parents, can help better estimate for the less annotated. Note that the backoff model would have a much smaller value than the original, due to double multiplication $\prod_r \prod_w$. In practice we normalize it by the sub-relations size to keep it at the same scale with $\tilde{P}(R \mid Q)$.

Finally, to estimate the prior and conditional probability, we need a massive data collection.

5.2 Steps

The ClueWeb09[6] dataset is a collection of 1 billion webpages (5TB compressed in raw HTML) in 10 languages by Carnegie Mellon University in 2009. FACC1, the Freebase Annotation of the ClueWeb Corpus version 1 (Gabrilovich et al., 2013), contains index and offset of Freebase entities within the English portion of ClueWeb. Out of all 500 million English documents, 340 million were automatically annotated with at least one entity, with an average of 15 entity mentions per document. The precision and recall of annotation were estimated at $80-85\%$ and $70-85\%$ (Orr et al., 2013).

Given these two resources, for each binary Freebase relation, we can find a collection of sentences each of which contains both of its arguments, then simply learn how words in these sentences are associated with this relation, i.e., $\tilde{P}(w \mid R)$ and $\tilde{P}(w \mid r)$. By counting how many times each relation R was annotated, we can estimate $\tilde{P}(R)$ and $\tilde{P}(r)$. The learning task can be framed in the following short steps:

1. We split each HTML document by sentences (Kiss and Strunk, 2006) using NLTK (Bird and Loper, 2004) and extracted those with at least two Freebase entities which has at least one direct established relation according to Freebase.
2. The extraction formed two parallel corpora, one with "relation - sentence" pairs (for estimating $\tilde{P}(w \mid R)$ and $\tilde{P}(R)$) and the other with "subrelations - sentence" pairs (for $\tilde{P}(w \mid r)$ and $\tilde{P}(r)$). Each corpus has 1.2 billion pairs.
3. The tricky part was to align these 1.2 billion pairs. Since the relations on one side of these pairs are not *natural* sentences, we ran the most simple IBM alignment Model 1 (Brown et al., 1993) to estimate the translation probability with GIZA++ (Och and Ney, 2003). To speed up, the 1.2 billion pairs were split into 100 even chunks. We ran 5 iterations of EM on each one and finally aligned the 1.2 billion pairs from both directions. To symmetrize the alignment, common MT heuristics INTERSECTION, UNION, GROW-DIAG-FINAL, and GROW-DIAG-FINAL-AND (Koehn, 2010) were separately applied and evaluated later.
4. Treating the aligned pairs as *observation*, the co-occurrence matrix between aligning relations and words was computed. There were 10,484 relations and sub-relations in all, and we kept the top 20,000 words.
5. From the co-occurrence matrix we computed $\tilde{P}(w \mid R)$, $\tilde{P}(R)$, $\tilde{P}(w \mid r)$ and $\tilde{P}(r)$.

[6]http://lemurproject.org/clueweb09/

0	$\le$ **10**	$\le$ **10^2**	$\le$ **10^3**	$\le$ **10^4**	$>$ **10^4**
7.0%	0.7%	1.2%	0.4%	1.3%	89.5%

Table 1: Percentage of answer relations (the incoming relation connected to the answer node) with respect to how many sentences we learned this relation from in CluewebMapping. For instance, the first column says there are 7% of answer relations for which we cannot find a mapping (so we had to use the backoff probability estimation); the last column says there are 89.5% of answer relations that we were able to learn the mapping between this relation and text based on more than 10 thousand relation-sentence pairs. The total number of answer relations is 7886.

Hand-checking the learned probabilities shows both success, failure and some bias. For instance, for the film.actor.film relation (mapping from film names to actor names), the top words given by $\tilde{P}(w \mid R)$ are won, star, among, show. For the film.film.directed_by relation, some important stop words that could indicate this relation, such as by and with, rank directly after director and direct. However, due to significant popular interest in certain news categories, and the resultant catering by websites to those information desires, then for example we also learned a heavily correlated connection between Jennifer Aniston and celebrity.infidelity.victim, and between some other you-know-who names and celebrity.infidelity.participant.

We next formally evaluate how the learned mapping help predict relations from words.

5.3 Evaluation

Both ClueWeb and its Freebase annotation has a bias. Thus we were firstly interested in the coverage of mined relation mappings. As a comparison, we used a dataset of relation mapping contributed by Berant et al. (2013) and Lin et al. (2012). The idea is very similar: they intersected Freebase relations with predicates in (arg1, predicate, arg2) triples extracted from `ReVerb` to learn the mapping between Freebase relations and triple predicates. Note the scale difference: although `ReVerb` was also extracted from ClueWeb09, there were only 15 million triples to intersect with the relations, while we had 1.2 billion alignment pairs. We call this dataset **ReverbMapping** and ours **CluewebMapping**.

The evaluation dataset, WEBQUESTIONS, was also contributed by Berant et al. (2013). It contains 3778 training and 2032 test questions collected from the Google Suggest service. All questions were annotated with answers from Freebase. Some questions have more than one answer, such as what to see near sedona arizona?.

We evaluated on the training set in two aspects: coverage and prediction performance. We define *answer node* as the node that is the answer and *answer relation* as the relation from the answer node to its direct parent. Then we computed how much and how well the answer relation was triggered by ReverbMapping and CluewebMapping. Thus for the question, who is the father of King George VI, we ask two questions: does the mapping, 1. (coverage) contain the answer relation people.person.parents? 2. (precision) predict the answer relation from the question?

Table 1 shows the coverage of CluewebMapping, which covers 93.0% of all answer relations. Among them, we were able to learn the rule mapping using more than 10 thousand relation-sentence pairs for *each* of the 89.5% of all answer relations. In contrast, ReverbMapping covers 89.7% of the answer relations.

Next we evaluated the prediction performance, using the evaluation metrics of information retrieval. For each question, we extracted all relations in its corresponding topic graph, and ranked each relation with whether it is the answer relation. For instance, for the previous example question, we want to rank the relation people.person.parents as number 1. We computed standard MAP (Mean Average Precision) and MRR (Mean Reciprocal Rank), shown in Table 2(a). As a simple baseline, "word overlap" counts the overlap between relations and the question. CluewebMapping ranks each relation by $\tilde{P}(R \mid Q)$. ReverbMapping does the same, except that we took a uniform distribution on $\tilde{P}(w \mid R)$ and $\tilde{P}(R)$ since the contributed dataset did not include co-occurrence counts to estimate these probabilities.[7] Note that the median rank from CluewebMapping is only 12, indicating that half of all answer relations are ranked in the top 12.

Table 2(b) further shows the percentage of answer relations with respect to their ranking. CluewebMapping successfully ranked 19% of answer relations as top 1. A sample of these includes person.place_of_birth, location.containedby, country.currency_used, regular_tv_appearance.actor, etc. These percentage numbers are good clue for feature design: for instance, we may be confident in a relation if it is ranked top 5 or 10 by CluewebMapping.

To conclude, we found that CluewebMapping provides satisfying coverage on the 3778 training questions: only 7% were missing, despite the biased nature of web data. Also, CluewebMapping gives reasonably good precision on its prediction, despite the noisy nature of web data. We move on to fully evaluate the final QA F_1.

6 Experiments

We evaluate the final F_1 in this section. The system of comparison is that of Berant et al. (2013).

Data We re-used WEBQUESTIONS, a dataset collected by Berant et al. (2013). It contains 5810 questions crawled from the Google Suggest service, with answers annotated on Amazon Mechanical Turk. All questions contain at least one answer from Freebase. This dataset has been split by 65%/35% into TRAIN-ALL and TEST. We further randomly divided TRAIN-ALL by 80%/20% to a smaller TRAIN and development set DEV. Note that our DEV set is different from that of Berant et al. (2013), but the final result on TEST is directly comparable. Results are reported in terms of macro F_1 with partial credit (following Berant et al. (2013)) if a predicted answer list does not have a perfect match with all gold answers, as a

[7] The way we used ReverbMapping was not how Berant et al. (2013) originally used it: they employed a discriminative log-linear model to judge relations and that might yield better performance. As a fair comparison, ranking of CluewebMapping under uniform distribution is also included in Table 2(a).

	Median Rank	MAP	MRR
word overlap	471	0.0380	0.0590
ReverbMapping	60	0.0691	0.0829
CluewebMapping	12	0.2074	0.2900
with uniform dist.	61	0.0544	0.0561

(a) Ranking on answer relations. Best result on CluewebMapping was under the GROW-DIAG-FINAL-AND heuristics (row 3) when symmetrizing alignment from both directions. The last row shows ranking of CluewebMapping under uniform distribution (assuming counting on words and relations is not known).

	1	≤ 5	≤ 10	≤ 50	≤ 100	> 100
w. o.	3.5	4.7	2.5	3.9	4.1	81.3
R.M.	2.6	9.1	8.6	26.0	13.0	40.7
C.M.	19.0	19.9	8.9	22.3	7.5	22.4

(b) Percentage of answer relations w.r.t. ranking number (header). w.o.: word overlap; R.M.: ReverbMapping; C.M.: CluewebMapping.

Table 2: Evaluation on answer relation ranking prediction on 3778 training questions.

top	1	2	3	5	10
#	3263	3456	3532	3574	3604
%	86.4	91.5	93.5	94.6	95.4

Table 3: Evaluation on the Freebase Search API: how many questions' top n retrieved results contain the gold standard topic. Total number of questions is 3778 (size of TRAIN-ALL). There were only 5 questions with no retrieved results.

	P	R	F_1
basic	57.3	30.1	39.5
+ word overlap	56.0	31.4	40.2
+ CluewebMapping	59.9	35.4	44.5
+both	59.0	35.4	44.3

Table 4: F_1 on DEV with different feature settings.

lot of questions in WEBQUESTIONS contain more than one answer.

Search With an Information Retrieval (IR) front-end, we need to locate the exact Freebase topic node a question is about. For this purpose we used the Freebase Search API (Freebase, 2013a).All named entities [8] in a question were sent to this API, which returned a ranked list of relevant topics. We also evaluated how well the search API served the IR purpose. WEBQUESTIONS not only has answers annotated, but also which Freebase topic nodes the answers come from. Thus we evaluated the ranking of retrieval with the gold standard annotation on TRAIN-ALL, shown in Table 3. The top 2 results of the Search API contain gold standard topics for more than 90% of the questions and the top 10 results contain more than 95%. We took this as a "good enough" IR front-end and used it on TEST.

Once a topic is obtained we query the Freebase Topic API (Freebase, 2013b) to retrieve all relevant information, resulting in a topic graph. The API returns almost identical information as displayed via a web browser to a user viewing this topic. Given that turkers annotated answers based on the topic page via a browser, this supports the assumption that the same answer would be located in the topic graph, which is then passed to the QA engine for feature extraction and classification.

[8]When no named entities are detected, we fall back to noun phrases.

Model Tuning We treat QA on Freebase as a binary classification task: for each node in the topic graph, we extract features and judge whether it is the answer node. Every question was processed by the Stanford CoreNLP suite with the caseless model. Then the question features (§4.1) and node features (§4.2) were combined (§4.3) for each node. The learning problem is challenging: for about 3000 questions in TRAIN, there are 3 million nodes (1000 nodes per topic graph), and 7 million feature types. We employed a high-performance machine learning tool, `Classias` (Okazaki, 2009). Training usually took around 4 hours. We experimented with various discriminative learners on DEV, including logistic regression, perceptron and SVM, and found L1 regularized logistic regression to give the best result. The L1 regularization encourages sparse features by driving feature weights towards zero, which was ideal for the over-generated feature space. After training, we had around 30 thousand features with non-zero weights, a 200 fold reduction from the original features.

Also, we did an ablation test on DEV about how additional features on the mapping between Freebase relations and the original questions help, with three feature settings: 1) "basic" features include feature productions read off from the feature graph (Figure 1); 2) "+ word overlap" adds additional features on whether sub-relations have overlap with the question; and 3) "+ CluewebMapping" adds the ranking of relation prediction given the question according to CluewebMapping. Table 4 shows that the additional CluewebMapping

	P	**R**	$\mathbf{F_1}$
Gold Retrieval	45.4	52.2	48.6
Freebase Search API	38.8	45.8	42.0
Berant et al. (2013)	-	-	31.4

Table 5: F_1 on TEST with Gold Retrieval and Freebase Search API as the IR front end. Berant et al. (2013) actually reported *accuracy* on this dataset. However, since their system predicted answers for almost every question (p.c.), it is roughly that precision=recall=F_1=accuracy for them.

features improved overall F_1 by 5%, a 13% relative improvement: a remarkable gain given that the model already learned a strong correlation between question types and answer types (explained more in discussion and Table 6 later).

Finally, the ratio of positive vs. negative examples affect final F_1: the more positive examples, the lower the precision and the higher the recall. Under the original setting, this ratio was about $1 : 275$. This produced precision around 60% and recall around 35% (c.f. Table 4). To optimize for F_1, we down-sampled the negative examples to 20%, i.e., a new ratio of $1 : 55$. This boosted the final F_1 on DEV to 48%. We report the final TEST result under this down-sampled training. In practice the precision/recall balance can be adjusted by the positive/negative ratio.

Test Results Table 5 gives the final F_1 on TEST. "Gold Retrieval" always ranked the correct topic node top 1, a perfect IR front-end assumption. In a more realistic scenario, we had already evaluated that the Freebase Search API returned the correct topic node 95% of the time in its top 10 results (c.f. Table 3), thus we also tested on the top 10 results returned by the Search API. To keep things simple, we did not perform answer voting, but simply extracted answers from the first (ranked by the Search API) topic node with predicted answer(s) found. The final F_1 of 42.0% gives a relative improvement over previous best result (Berant et al., 2013) of 31.4% by one third.

One question of interest is whether our system, aided by the massive web data, can be fairly compared to the semantic parsing approaches (note that Berant et al. (2013) also used ClueWeb indirectly through `ReVerb`). Thus we took out the word overlapping and CluewebMapping based features, and the new F_1 on TEST was 36.9%.

The other question of interest is that whether our system has acquired some level of "machine intelligence": how much does it know what the question inquires? We discuss it below through feature and error analysis.

wgt.	**feature**
5.56	qfocus=money|type=Currency
5.35	qverb=die|type=Cause_Of_Death
5.11	qword=when|type=datetime
4.56	qverb=border|rel=location.adjoins
3.90	qword=why|incoming_relation_rank=top_3
2.94	qverb=go|qtopic=location|type=Tourist_attraction
-3.94	qtopic=location|rel=location.imports_exports.date
-2.93	qtopic=person|rel=education.end_date

Table 6: A sample of the top 50 most positive/negative features. Features are production between question and node features (c.f. Figure 1).

Discussion The combination between questions and Freebase nodes captures some real gist of QA pattern typing, shown in Table 6 with sampled features and weights. Our system learned, for instance, when the question asks for geographic adjacency information (qverb=border), the correct answer relation to look for is location.adjoins. Detailed comparison with the output from Berant et al. (2013) is a work in progress and will be presented in a follow-up report.

7 Conclusion

We proposed an automatic method for Question Answering from structured data source (Freebase). Our approach associates question features with answer patterns described by Freebase and has achieved state-of-the-art results on a balanced and realistic QA corpus. To compensate for the problem of domain mismatch or overfitting, we exploited ClueWeb, mined mappings between KB relations and natural language text, and showed that it helped both relation prediction and answer extraction. Our method employs relatively lightweight machinery but has good performance. We hope that this result establishes a new baseline against which semantic parsing researchers can measure their progress towards deeper language understanding and answering of human questions.

Acknowledgments We thank the Allen Institute for Artificial Intelligence for funding this work. We are also grateful to Jonathan Berant, Tom Kwiatkowski, Qingqing Cai, Adam Lopez, Chris Callison-Burch and Peter Clark for helpful discussion and to the reviewers for insightful comments.

References

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. DBPedia: A nucleus for a web of open data. In *The semantic web*, pages 722–735. Springer.

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

Steven Bird and Edward Loper. 2004. NLTK: The Natural Language Toolkit. In *Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics*.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. ACM.

Peter F Brown, Vincent J Della Pietra, Stephen A Della Pietra, and Robert L Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational linguistics*, 19(2):263–311.

Qingqing Cai and Alexander Yates. 2013. Large-scale semantic parsing via schema matching and lexicon extension. In *Proceedings of ACL*.

David L Chen and Raymond J Mooney. 2011. Learning to Interpret Natural Language Navigation Instructions from Observations. In *AAAI*, volume 2, pages 1–2.

J. Chu-Carroll, J. Fan, B. K. Boguraev, D. Carmel, D. Sheinwald, and C. Welty. 2012. Finding needles in the haystack: Search and candidate generation. *IBM Journal of Research and Development*.

Jérôme Euzenat and Pavel Shvaiko. 2007. *Ontology matching*. Springer.

Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In *Proceedings of EMNLP*.

Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. 2013. Paraphrase-Driven Learning for Open Question Answering. In *Proceedings of ACL*.

Anette Frank, Hans-Ulrich Krieger, Feiyu Xu, Hans Uszkoreit, Berthold Crysmann, Brigitte Jörg, and Ulrich Schäfer. 2007. Question answering from structured knowledge sources. *Journal of Applied Logic*, 5(1):20–48.

Freebase. 2013a. Freebase Search API. https://developers.google.com/freebase/v1/search-overview.

Freebase. 2013b. Freebase Topic API. https://developers.google.com/freebase/v1/topic-overview.

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). http://lemurproject.org/clueweb09/FACC1/, June.

Bert F Green Jr, Alice K Wolf, Carol Chomsky, and Kenneth Laughery. 1961. Baseball: an automatic question-answerer. In *Papers presented at the May 9-11, 1961, western joint IRE-AIEE-ACM computer conference*, pages 219–224. ACM.

Jerry R Hobbs. 1985. Ontological promiscuity. In *Proceedings of ACL*.

Johannes Hoffart, Fabian M Suchanek, Klaus Berberich, Edwin Lewis-Kelham, Gerard De Melo, and Gerhard Weikum. 2011. Yago2: exploring and querying world knowledge in time, space, context, and many languages. In *Proceedings of the 20th international conference companion on World Wide Web*, pages 229–232. ACM.

Bevan Keeley Jones, Mark Johnson, and Sharon Goldwater. 2012. Semantic parsing with bayesian tree transducers. In *Proceedings of ACL*.

Rohit J Kate and Raymond J Mooney. 2006. Using string-kernels for learning semantic parsers. In *Proceedings of ACL*.

Tibor Kiss and Jan Strunk. 2006. Unsupervised multilingual sentence boundary detection. *Computational Linguistics*, 32(4):485–525.

Philipp Koehn. 2010. *Statistical Machine Translation*. Cambridge University Press, New York, NY, USA.

Jayant Krishnamurthy and Tom M Mitchell. 2012. Weakly supervised training of semantic parsers. In *Proceedings of EMNLP-CoNLL*.

Tom Kwiatkowski, Luke Zettlemoyer, Sharon Goldwater, and Mark Steedman. 2010. Inducing probabilistic CCG grammars from logical form with higher-order unification. In *Proceedings of EMNLP*, pages 1223–1233.

Tom Kwiatkowski, Luke Zettlemoyer, Sharon Goldwater, and Mark Steedman. 2011. Lexical generalization in CCG grammar induction for semantic parsing. In *Proceedings of EMNLP*.

Tom Kwiatkowski, Eunsol Choi, Yoav Artzi, and Luke Zettlemoyer. 2013. Scaling Semantic Parsers with On-the-fly Ontology Matching. In *Proceedings of EMNLP*.

Percy Liang, Michael I. Jordan, and Dan Klein. 2011. Learning Dependency-Based Compositional Semantics. In *Proceedings of ACL*.

Thomas Lin, Oren Etzioni, et al. 2012. Entity Linking at Web Scale. In *Proceedings of Knowledge Extraction Workshop (AKBC-WEKEX)*, pages 84–88.

Vanessa Lopez, Michele Pasin, and Enrico Motta. 2005. Aqualog: An ontology-portable question answering system for the semantic web. In *The Semantic Web: Research and Applications*, pages 546–562. Springer.

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

Naoaki Okazaki. 2009. Classias: a collection of machine-learning algorithms for classification.

Dave Orr, Amar Subramanya, Evgeniy Gabrilovich, and Michael Ringgaard. 2013. 11 billion clues in 800 million documents: A web research corpus annotated with freebase concepts. http://googleresearch.blogspot.com/2013/07/11-billion-clues-in-800-million.html, July.

Erhard Rahm and Philip A Bernstein. 2001. A survey of approaches to automatic schema matching. *the VLDB Journal*, 10(4):334–350.

Saeedeh Shekarpour, Axel-Cyrille Ngonga Ngomo, and Sören Auer. 2013. Question answering on interlinked data. In *Proceedings of WWW*.

Lappoon R Tang and Raymond J Mooney. 2001. Using multiple clause constructors in inductive logic programming for semantic parsing. In *Machine Learning: ECML 2001*, pages 466–477. Springer.

Christina Unger, Lorenz Bühmann, Jens Lehmann, Axel-Cyrille Ngonga Ngomo, Daniel Gerber, and Philipp Cimiano. 2012. Template-based question answering over RDF data. In *Proceedings of the 21st international conference on World Wide Web*.

Yuk Wah Wong and Raymond J Mooney. 2007. Learning synchronous grammars for semantic parsing with lambda calculus. In *Proceedings of ACL*.

William A Woods. 1977. Lunar rocks in natural english: Explorations in natural language question answering. *Linguistic structures processing*, 5:521–569.

Mohamed Yahya, Klaus Berberich, Shady Elbassuoni, Maya Ramanath, Volker Tresp, and Gerhard Weikum. 2012. Natural language questions for the web of data. In *Proceedings of EMNLP*.

Luke S Zettlemoyer and Michael Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. *Uncertainty in Artificial Intelligence (UAI)*.

Luke S Zettlemoyer and Michael Collins. 2007. Online learning of relaxed CCG grammars for parsing to logical form. In *Proceedings of EMNLP-CoNLL*.

Luke S Zettlemoyer and Michael Collins. 2009. Learning context-dependent mappings from sentences to logical form. In *Proceedings of ACL-CoNLL*.