Question Answering on Freebase via Relation Extraction and Textual Evidence

Kun Xu[1], **Siva Reddy**[2], **Yansong Feng**[1,*], **Songfang Huang**[3] and **Dongyan Zhao**[1]

[1]Institute of Computer Science & Technology, Peking University, Beijing, China
[2]School of Informatics, University of Edinburgh, UK
[3]IBM China Research Lab, Beijing, China
{xukun, fengyansong, zhaody}@pku.edu.cn
siva.reddy@ed.ac.uk
huangsf@cn.ibm.com

Abstract

Existing knowledge-based question answering systems often rely on small annotated training data. While shallow methods like relation extraction are robust to data scarcity, they are less expressive than the deep meaning representation methods like semantic parsing, thereby failing at answering questions involving multiple constraints. Here we alleviate this problem by empowering a relation extraction method with additional evidence from Wikipedia. We first present a neural network based relation extractor to retrieve the candidate answers from Freebase, and then infer over Wikipedia to validate these answers. Experiments on the WebQuestions question answering dataset show that our method achieves an F_1 of 53.3%, a substantial improvement over the state-of-the-art.

1 Introduction

Since the advent of large structured knowledge bases (KBs) like Freebase (Bollacker et al., 2008), YAGO (Suchanek et al., 2007) and DBpedia (Auer et al., 2007), answering natural language questions using those structured KBs, also known as KB-based question answering (or KB-QA), is attracting increasing research efforts from both natural language processing and information retrieval communities.

The state-of-the-art methods for this task can be roughly categorized into two streams. The first is based on semantic parsing (Berant et al., 2013; Kwiatkowski et al., 2013), which typically learns a grammar that can parse natural language to a sophisticated meaning representation language. But such sophistication requires a lot of annotated training examples that contains compositional struc-

tures, a practically impossible solution for large KBs such as Freebase. Furthermore, mismatches between grammar predicted structures and KB structure is also a common problem (Kwiatkowski et al., 2013; Berant and Liang, 2014; Reddy et al., 2014).

On the other hand, instead of building a formal meaning representation, information extraction methods retrieve a set of candidate answers from KB using relation extraction (Yao and Van Durme, 2014; Yih et al., 2014; Yao, 2015; Bast and Haussmann, 2015) or distributed representations (Bordes et al., 2014; Dong et al., 2015). Designing large training datasets for these methods is relatively easy (Yao and Van Durme, 2014; Bordes et al., 2015; Serban et al., 2016). These methods are often good at producing an answer irrespective of their correctness. However, handling compositional questions that involve multiple entities and relations, still remains a challenge. Consider the question *what mountain is the highest in north america*. Relation extraction methods typically answer with all the mountains in *North America* because of the lack of sophisticated representation for the mathematical function *highest*. To select the correct answer, one has to retrieve all the heights of the mountains, and sort them in descending order, and then pick the first entry. We propose a method based on textual evidence which can answer such questions without solving the mathematic functions implicitly.

Knowledge bases like Freebase capture real world facts, and Web resources like Wikipedia provide a large repository of sentences that validate or support these facts. For example, a sentence in Wikipedia says, *Denali (also known as Mount McKinley, its former official name) is the highest mountain peak in North America, with a summit elevation of 20,310 feet (6,190 m) above sea level.* To answer our example question against a KB using a relation extractor, we can use this sentence

as external evidence, filter out wrong answers and pick the correct one.

Using textual evidence not only mitigates representational issues in relation extraction, but also alleviates the data scarcity problem to some extent. Consider the question, *who was queen isabella's mother.* Answering this question involves predicting two constraints hidden in the word *mother.* One constraint is that the answer should be the *parent* of *Isabella*, and the other is that the answer's *gender* is *female.* Such words with multiple latent constraints have been a pain-in-the-neck for both semantic parsing and relation extraction, and requires larger training data (this phenomenon is coined as sub-lexical compositionality by Wang et al. (2015)). Most systems are good at triggering the *parent* constraint, but fail on the other, i.e., the answer entity should be *female.* Whereas the textual evidence from Wikipedia, *. . . her mother was Isabella of Barcelos . . .*, can act as a further constraint to answer the question correctly.

We present a novel method for question answering which infers on both structured and unstructured resources. Our method consists of two main steps as outlined in §2. In the first step we extract answers for a given question using a structured KB (here Freebase) by jointly performing entity linking and relation extraction (§3). In the next step we validate these answers using an unstructured resource (here Wikipedia) to prune out the wrong answers and select the correct ones (§4). Our evaluation results on a benchmark dataset WebQuestions show that our method outperforms existing state-of-the-art models. Details of our experimental setup and results are presented in §5. Our code, data and results can be downloaded from `https://github.com/syxu828/QuestionAnsweringOverFB`.

2 Our Method

Figure 1 gives an overview of our method for the question "*who did shaq first play for*". We have two main steps: (1) inference on Freebase (*KB-QA* box); and (2) further inference on Wikipedia (*Answer Refinement* box). Let us take a close look into step 1. Here we perform *entity linking* to identify a topic entity in the question and its possible Freebase entities. We employ a *relation extractor* to predict the potential Freebase relations that could exist between the entities in the question and the answer entities. Later we perform a *joint inference* step over the entity linking and relation extraction
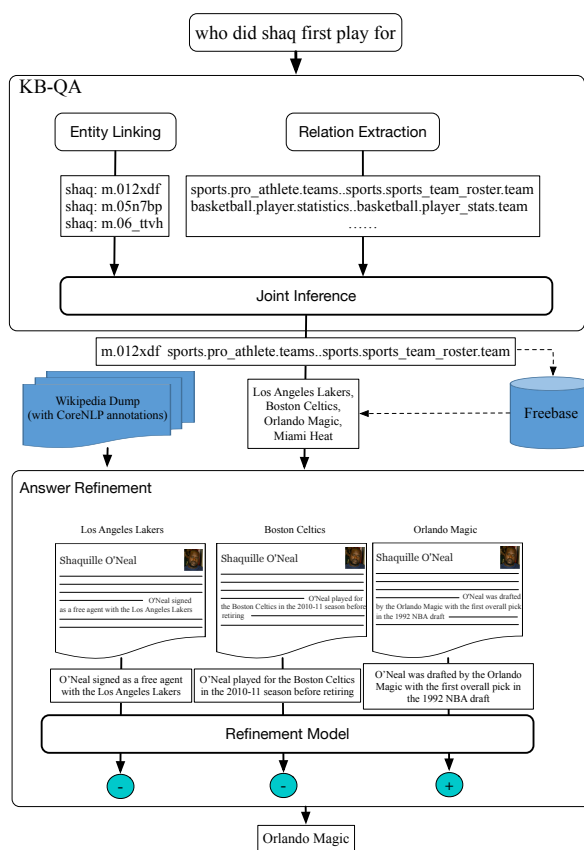

Figure 1: An illustration of our method to find answers for the given question *who did shaq first play for*.

results to find the best entity-relation configuration which will produce a list of candidate answer entities. In the step 2, we refine these candidate answers by applying an *answer refinement* model which takes the Wikipedia page of the topic entity into consideration to filter out the wrong answers and pick the correct ones.

While the overview in Figure 1 works for questions containing single Freebase relation, it also works for questions involving multiple Freebase relations. Consider the question *who plays anakin skywalker in star wars 1*. The actors who are the answers to this question should satisfy the following constraints: (1) the actor played *anakin skywalker*; and (2) the actor played in *star wars 1*. Inspired by Bao et al. (2014), we design a dependency tree-based method to handle such multi-relational questions. We first decompose the original question into a set of *sub-questions* using syntactic patterns which are listed in Appendix. The final answer set of the original question is obtained by intersecting the answer sets of all its sub-questions. For the

example question, the sub-questions are *who plays anakin skywalker* and *who plays in star wars 1*. These sub-questions are answered separately over Freebase and Wikipedia, and the intersection of their answers to these sub-questions is treated as the final answer.

3 Inference on Freebase

Given a sub-question, we assume the question word[1] that represents the answer has a distinct KB relation r with an entity e found in the question, and predict a single KB triple $(e, r, ?)$ for each sub-question (here $?$ stands for the answer entities). The QA problem is thus formulated as an information extraction problem that involves two sub-tasks, i.e., entity linking and relation extraction. We first introduce these two components, and then present a joint inference procedure which further boosts the overall performance.

3.1 Entity Linking

For each question, we use hand-built sequences of part-of-speech categories to identify all possible named entity mention spans, e.g., the sequence *NN* (*shaq*) may indicate an entity. For each mention span, we use the entity linking tool S-MART[2] (Yang and Chang, 2015) to retrieve the top 5 entities from Freebase. These entities are treated as candidate entities that will eventually be disambiguated in the joint inference step. For a given mention span, S-MART first retrieves all possible entities of Freebase by surface matching, and then ranks them using a statistical model, which is trained on the frequency counts with which the surface form occurs with the entity.

3.2 Relation Extraction

We now proceed to identify the relation between the answer and the entity in the question. Inspired by the recent success of neural network models in KB question-answering (Yih et al., 2015; Dong et al., 2015), and the success of syntactic dependencies for relation extraction (Liu et al., 2015; Xu et al., 2015), we propose a Multi-Channel Convolutional Neural Network (MCCNN) which could exploit both syntactic and sentential information for relation extraction.
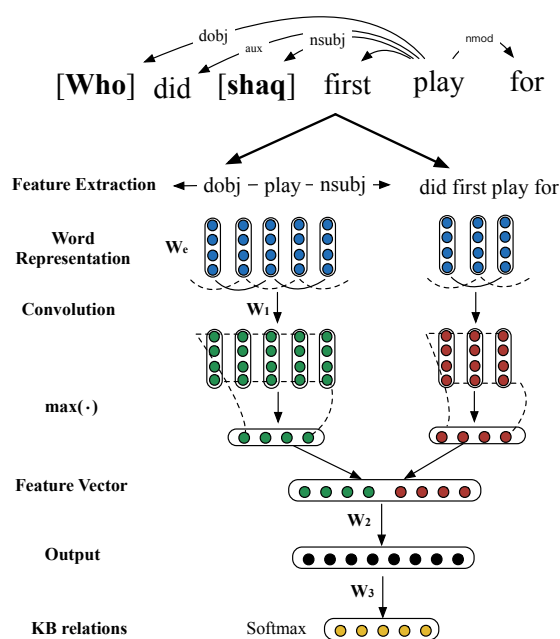

Figure 2: Overview of the multi-channel convolutional neural network for relation extraction. W_e is the word embedding matrix, W_1 is the convolution matrix, W_2 is the activation matrix and W_3 is the classification matrix.

3.2.1 MCCNNs for Relation Classification

In MCCNN, we use two channels, one for syntactic information and the other for sentential information. The network structure is illustrated in Figure 2. Convolution layer tackles an input of varying length returning a fixed length vector (we use max pooling) for each channel. These fixed length vectors are concatenated and then fed into a *softmax* classifier, the output dimension of which is equal to the number of predefined relation types. The value of each dimension indicates the confidence score of the corresponding relation.

Syntactic Features   We use the shortest path between an entity mention and the question word in the dependency tree[3] as input to the first channel. Similar to Xu et al. (2015), we treat the path as a concatenation of vectors of words, dependency edge directions and dependency labels, and feed it to the convolution layer. Note that, the entity mention and the question word are excluded from the dependency path so as to learn a more general relation representation in syntactic level. As shown in Figure 2, the dependency path between *who* and *shaq* is $\leftarrow dobj - play - nsubj \rightarrow$.

Sentential Features This channel takes the words in the sentence as input excluding the question word and the entity mention. As illustrated in Figure 2, the vectors for *did, first, play* and *for* are fed into this channel.

3.2.2 Objective Function and Learning

The model is learned using pairs of question and its corresponding gold relation from the training data. Given an input question x with an annotated entity mention, the network outputs a vector $o(x)$, where the entry $o_k(x)$ is the probability that there exists the k-th relation between the entity and the expected answer. We denote $t(x) \in \mathbb{R}^{K \times 1}$ as the target distribution vector, in which the value for the gold relation is set to 1, and others to 0. We compute the cross entropy error between $t(x)$ and $o(x)$, and further define the objective function over the training data as:

$$ J(\theta) = -\sum_x \sum_{k=1}^{K} t_k(x) \log o_k(x) + \lambda ||\theta||_2^2 $$

where θ represents the weights, and λ the $L2$ regularization parameters. The weights θ can be efficiently computed via back-propagation through network structures. To minimize $J(\theta)$, we apply stochastic gradient descent (SGD) with AdaGrad (Duchi et al., 2011).

3.3 Joint Entity Linking & Relation Extraction

A pipeline of entity linking and relation extraction may suffer from error propagations. As we know, entities and relations have strong selectional preferences that certain entities do not appear with certain relations and vice versa. Locally optimized models could not exploit these implicit *bi-directional* preferences. Therefore, we use a joint model to find a globally optimal entity-relation assignment from local predictions. The key idea behind is to leverage various clues from the two local models and the KB to rank a correct entity-relation assignment higher than other combinations. We describe the learning procedure and the features below.

3.3.1 Learning

Suppose the pair (e_{gold}, r_{gold}) represents the gold entity/relation pair for a question q. We take all our entity and relation predictions for q, create a list of entity and relation pairs $\{(e_0, r_0), (e_1, r_1), ..., (e_n, r_n)\}$ from q and rank them using an SVM rank classifier (Joachims, 2006) which is trained to predict a rank for each pair. Ideally higher rank indicates the prediction is closer to the gold prediction. For training, SVM rank classifier requires a ranked or scored list of entity-relation pairs as input. We create the training data containing ranked input pairs as follows: if both $e_{pred} = e_{gold}$ and $r_{pred} = r_{gold}$, we assign it with a score of 3. If only the entity or relation equals to the gold one (i.e., $e_{pred} = e_{gold}, r_{pred} \neq r_{gold}$ or $e_{pred} \neq e_{gold}, r_{pred} = r_{gold}$), we assign a score of 2 (encouraging partial overlap). When both entity and relation assignments are wrong, we assign a score of 1.

3.3.2 Features

For a given entity-relation pair, we extract the following features which are passed as an input vector to the SVM ranker above:

Entity Clues. We use the score of the predicted entity returned by the entity linking system as a feature. The number of word overlaps between the entity mention and entity's Freebase name is also included as a feature. In Freebase, most entities have a relation *fb:description* which describes the entity. For instance, in the running example, *shaq* is linked to three potential entities *m.06_ttvh (Shaq Vs. Television Show)*, *m.05n7bp (Shaq Fu Video Game)* and *m.012xdf (Shaquille O'Neal)*. Interestingly, the word *play* only appears in the description of *Shaquille O'Neal* and it occurs three times. We count the content word overlap between the given question and the entity's description, and include it as a feature.

Relation Clues. The score of relation returned by the MCCNNs is used as a feature. Furthermore, we view each relation as a *document* which consists of the training questions that this relation is expressed in. For a given question, we use the sum of the *tf-idf* scores of its words with respect to the relation as a feature. A Freebase relation r is a concatenation of a series of fragments $r = r_1.r_2.r_3$. For instance, the three fragments of *people.person.parents* are *people, person* and *parents*. The first two fragments indicate the Freebase type of the subject of this relation, and the third fragment indicates the object type, in our case the answer type. We use an indicator feature to denote if the surface form of the third fragment (here *parents*) appears in the question.

Answer Clues. The above two feature classes indicate local features. From the entity-relation (e, r)

2329

pair, we create the query triple $(e, r, ?)$ to retrieve the answers, and further extract features from the answers. These features are non-local since we require both e and r to retrieve the answer. One such feature is using the co-occurrence of the answer type and the question word based on the intuition that question words often indicate the answer type, e.g., the question word *when* usually indicates the answer type *type.datetime*. Another feature is the number of answer entities retrieved.

4 Inference on Wikipedia

We use the best ranked entity-relation pair from the above step to retrieve candidate answers from Freebase. In this step, we validate these answers using Wikipedia as our unstructured knowledge resource where most statements in it are verified for factuality by multiple people.

Our refinement model is inspired by the intuition of how people refine their answers. If you ask someone: *who did shaq first play for*, and give them four candidate answers (*Los Angeles Lakers*, *Boston Celtics*, *Orlando Magic* and *Miami Heat*), as well as access to Wikipedia, that person might first determine that the question is about *Shaquille O'Neal*, then go to *O'Neal*'s Wikipedia page, and search for the sentences that contain the candidate answers as evidence. By analyzing these sentences, one can figure out whether a candidate answer is correct or not.

4.1 Finding Evidence from Wikipedia

As mentioned above, we should first find the Wikipedia page corresponding to the *topic entity* in the given question. We use Freebase API to convert Freebase entity to Wikipedia page. We extract the content from the Wikipedia page and process it with *Wikifier* (Cheng and Roth, 2013) which recognizes Wikipedia entities, which can further be linked to Freebase entities using Freebase API. Additionally we use Stanford CoreNLP (Manning et al., 2014) for tokenization and entity co-reference resolution. We search for the sentences containing the candidate answer entities retrieved from Freebase. For example, the Wikipedia page of *O'Neal* contains a sentence "*O'Neal was drafted by the Orlando Magic with the first overall pick in the 1992 NBA draft*", which is taken into account by the *refinement model* (our inference model on Wikipedia) to discriminate whether *Orlando Magic* is the answer for the given question.

4.2 Refinement Model

We treat the refinement process as a binary classification task over the candidate answers, i.e., correct (positive) and incorrect (negative) answer. We prepare the training data for the refinement model as follows. On the training dataset, we first infer on Freebase to retrieve the candidate answers. Then we use the annotated gold answers of these questions and Wikipedia to create the training data. Specifically, we treat the sentences that contain correct/incorrect answers as positive/negative examples for the refinement model. We use LIBSVM (Chang and Lin, 2011) to learn the weights for classification.

Note that, in the Wikipedia page of the topic entity, we may collect more than one sentence that contain a candidate answer. However, not all sentences are relevant, therefore we consider the candidate answer as correct *if* at least there is one positive evidence. On the other hand, sometimes, we may not find any evidence for the candidate answer. In these cases, we fall back to the results of the KB-based approach.

4.3 Lexical Features

Regarding the features used in LIBSVM, we use the following lexical features extracted from the question and a Wikipedia sentence. Formally, given a question $q = <q_1, ... q_n>$ and an evidence sentence $s = <s_1, ... s_m>$, we denote the tokens of q and s by q_i and s_j, respectively. For each pair (q, s), we identify a set of all possible token pairs (q_i, s_j), the occurrences of which are used as features. As learning proceeds, we hope to learn a higher weight for a feature like (*first*, *drafted*) and a lower weight for (*first*, *played*).

5 Experiments

In this section we introduce the experimental setup, the main results and detailed analysis of our system.

5.1 Training and Evaluation Data

We use the WebQuestions (Berant et al., 2013) dataset, which contains 5,810 questions crawled via Google Suggest service, with answers annotated on Amazon Mechanical Turk. The questions are split into training and test sets, which contain 3,778 questions (65%) and 2,032 questions (35%), respectively. We further split the training questions into 80%/20% for development.

To train the MCCNNs and the joint inference model, we need the gold standard relations of the questions. Since this dataset contains only question-answer pairs and annotated topic entities, instead of relying on gold relations we rely on *surrogate* gold relations which produce answers that have the highest overlap with gold answers. Specifically, for a given question, we first locate the topic entity e in the Freebase graph, then select 1-hop and 2-hop relations connected to the topic entity as relation candidates. The 2-hop relations refer to the n-ary relations of Freebase, i.e., first hop from the subject to a mediator node, and the second from the mediator to the object node. For each relation candidate r, we issue the query $(e, r, ?)$ to the KB, and label the relation that produces the answer with minimal F_1-loss against the gold answer, as the *surrogate* gold relation. From the training set, we collect 461 relations to train the MCCNN, and the target prediction during testing time is over these relations.

5.2 Experimental Settings

We have 6 dependency tree patterns based on Bao et al. (2014) to decompose the question into sub-questions (See Appendix). We initialize the word embeddings with Turian et al. (2010)'s word representations with dimensions set to 50. The hyper parameters in our model are tuned using the development set. The window size of MCCNN is set to 3. The sizes of the hidden layer 1 and the hidden layer 2 of the two MCCNN channels are set to 200 and 100, respectively. We use the Freebase version of Berant et al. (2013), containing 4M entities and 5,323 relations.

5.3 Results and Discussion

We use the average question-wise F_1 as our evaluation metric.[4] To give an idea of the impact of different configurations of our method, we compare the following with existing methods.

Structured. This method involves inference on Freebase only. First the entity linking (EL) system is run to predict the topic entity. Then we run the relation extraction (RE) system and select the best relation that can occur with the topic entity. We choose this entity-relation pair to predict the answer.

Method	average F_1
Berant et al. (2013)	35.7
Yao and Van Durme (2014)	33.0
Xu et al. (2014)	39.1
Berant and Liang (2014)	39.9
Bao et al. (2014)	37.5
Bordes et al. (2014)	39.2
Dong et al. (2015)	40.8
Yao (2015)	44.3
Bast and Haussmann (2015)	49.4
Berant and Liang (2015)	49.7
Reddy et al. (2016)	50.3
Yih et al. (2015)	52.5
This work	
Structured	44.1
Structured + Joint	47.1
Structured + Unstructured	47.0
Structured + Joint + Unstructured	**53.3**

Table 1: Results on the test set.

Structured + Joint. In this method instead of the above pipeline, we perform joint EL and RE as described in §3.3.

Structured+Unstructured. We use the pipelined EL and RE along with inference on Wikipedia as described in §4.

Structured + Joint + Unstructured. This is our main model. We perform inference on Freebase using joint EL and RE, and then inference on Wikipedia to validate the results. Specifically, we treat the top two predictions of the joint inference model as the candidate subject and relation pairs, and extract the corresponding answers from each pair, take the union, and filter the answer set using Wikipedia.

Table 1 summarizes the results on the test data along with the results from the literature.[5] We can see that joint EL and RE performs better than the default pipelined approach, and outperforms most semantic parsing based models, except (Berant and Liang, 2015) which searches partial logical forms in strategic order by combining imitation learning and agenda-based parsing. In addition, inference on unstructured data helps the default model. The joint EL and RE combined with inference on unstructured data further improves the default pipelined model by 9.2% (from 44.1% to 53.3%), and achieves a new state-of-the-art result beating the previous reported best result of Yih et al. (2015) (with one-tailed t-test significance of $p < 0.05$).

---

[4]We use the evaluation script available at http://www-nlp.stanford.edu/software/sempre.

[5]We use development data for all our ablation experiments. Similar trends are observed on both development and test results.

	Entity Linking Accuracy	Relation Extraction Accuracy
Isolated Model	79.8	45.9
Joint Inference	**83.2**	**55.3**

Table 2: Impact of the joint inference on the development set

Method	average F_1
Structured (syntactic)	38.1
Structured (sentential)	38.7
Structured (syntactic + sentential)	40.1
Structured + Joint (syntactic)	43.6
Structured + Joint (sentential)	44.1
Structured + Joint (syntactic + sentential)	**45.8**

Table 3: Impact of different MCCNN channels on the development set.

5.3.1 Impact of Joint EL & RE

From Table 1, we can see that the joint EL & RE gives a performance boost of 3% (from 44.1 to 47.1). We also analyze the impact of joint inference on the individual components of EL & RE.

We first evaluate the EL component using the gold entity annotations on the development set. As shown in Table 2, for 79.8% questions, our entity linker can correctly find the gold standard topic entities. The joint inference improves this result to 83.2%, a 3.4% improvement. Next we use the *surrogate* gold relations to evaluate the performance of the RE component on the development set. As shown in Table 2, the relation prediction accuracy increases by 9.4% (from 45.9% to 55.3%) when using the joint inference.

5.3.2 Impact of the Syntactic and the Sentential Channels

Table 3 presents the results on the impact of individual and joint channels on the end QA performance. When using a single-channel network, we tune the parameters of only one channel while switching off the other channel. As seen, the sentential features are found to be more important than syntactic features. We attribute this to the short and noisy nature of WebQuestions questions due to which syntactic parser wrongly parses or the shortest dependency path does not contain sufficient information to predict a relation. By using both the channels, we see further improvements than using any one of the channels.

Question & Answers
1. what is the largest nation in europe Before: Kazakhstan, Turkey, Russia, ... After: Russia
2. which country in europe has the largest land area Before: Georgia, France, Russia, ... After: Russian Empire, Russia
3. what year did ray allen join the nba Before: 2007, 2003, 1996, 1993, 2012 After: 1996
4. who is emma stone father Before: Jeff Stone, Krista Stone After: Jeff Stone
5. where did john steinbeck go to college Before: Salinas High School, Stanford University After: Stanford University

Table 4: Example questions and corresponding predicted answers before and after using unstructured inference. *Before* uses (*Structured + Joint*) model, and *After* uses *Structured + Joint + Unstructured* model for prediction. The colors *blue* and *red* indicate correct and wrong answers respectively.

5.3.3 Impact of the Inference on Unstructured Data

As shown in Table 1, when structured inference is augmented with the unstructured inference, we see an improvement of 2.9% (from 44.1% to 47.0%). And when *Structured + Joint* uses unstructured inference, the performance boosts by 6.2% (from 47.1% to 53.3%) achieving a new state-of-the-art result. For the latter, we manually analyzed the cases in which unstructured inference helps. Table 4 lists some of these questions and the corresponding answers before and after the unstructured inference. We observed the unstructured inference mainly helps for two classes of questions: (1) questions involving aggregation operations (Questions 1-3); (2) questions involving sub-lexical compositionally (Questions 4-5). Questions 1 and 2 contain the predicate $largest$ an aggregation operator. A semantic parsing method should explicitly handle this predicate to trigger $max(.)$ operator. For Question 3, structured inference predicts the Freebase relation *fb:teams..from* retrieving all the years in which *Ray Allen* has played basketball. Note that *Ray Allen* has joined *Connecticut University*'s team in *1993* and *NBA* from *1996*. To answer this question a semantic parsing system would require a *min(·)* operator along with an additional constraint that the year corresponds to the *NBA*'s term. Interestingly, without having to explicitly model these complex predicates, the unstructured inference helps in answering these questions more accurately. Questions 4-5 involve sub-lexical com-

positionally (Wang et al., 2015) predicates *father* and *college*. For example in Question 5, the user queries for the colleges that *John Steinbeck* attended. However, Freebase defines the relation *fb:education..institution* to describe a person's educational information without discriminating the specific periods such as high school or college. Inference using unstructured data helps in alleviating these representational issues.

5.3.4 Error analysis

We analyze the errors of *Structured + Joint + Unstructured* model. Around 15% of the errors are caused by incorrect entity linking, and around 50% of the errors are due to incorrect relation predictions. The errors in relation extraction are due to (i) insufficient context, e.g., in *what is duncan bannatyne*, neither the dependency path nor sentential context provides enough evidence for the MCCNN model; (ii) unbalanced distribution of relations (3022 training examples for 461 relations) heavily influences the performance of MCCNN model towards frequently seen relations. The remaining errors are the failure of unstructured inference due to insufficient evidence in Wikipedia or misclassification.

Entity Linking. In the entity linking component, we had handcrafted POS tag patterns to identify entity mentions, e.g., DT-JJ-NN (noun phrase), NN-IN-NN (prepositional phrase). These patterns are designed to have high recall. Around 80% of entity linking errors are due to incorrect entity prediction even when the correct mention span was found.

Question Decomposition. Around 136 questions (15%) of dev data contains compositional questions, leading to 292 sub-questions (around 2.1 subquestions for a compositional question). Since our question decomposition component is based on manual rules, one question of interest is how these rules perform on other datasets. By human evaluation, we found these rules achieves 95% on a more general but complex QA dataset QALD-5[6].

5.3.5 Limitations

While our unstructured inference alleviates representational issues to some extent, we still fail at modeling compositional questions such as *who is the mother of the father of prince william* involving

---

[6] http://qald.sebastianwalter.org/index.php?q=5

multi-hop relations and the inter alia. Our current assumption that unstructured data could provide evidence for questions may work only for frequently typed queries or for popular domains like movies, politics and geography. We note these limitations and hope our result will foster further research in this area.

6 Related Work

Over time, the QA task has evolved into two main streams – QA on unstructured data, and QA on structured data. TREC QA evaluations (Voorhees and Tice, 1999) were a major boost to unstructured QA leading to richer datasets and sophisticated methods (Wang et al., 2007; Heilman and Smith, 2010; Yao et al., 2013; Yih et al., 2013; Yu et al., 2014; Yang et al., 2015; Hermann et al., 2015). While initial progress on structured QA started with small toy domains like GeoQuery (Zelle and Mooney, 1996), recent focus has shifted to large scale structured KBs like Freebase, DB-Pedia (Unger et al., 2012; Cai and Yates, 2013; Berant et al., 2013; Kwiatkowski et al., 2013; Xu et al., 2014), and on noisy KBs (Banko et al., 2007; Carlson et al., 2010; Krishnamurthy and Mitchell, 2012; Fader et al., 2013; Parikh et al., 2015). An exciting development in structured QA is to exploit multiple KBs (with different schemas) at the same time to answer questions jointly (Yahya et al., 2012; Fader et al., 2014; Zhang et al., 2016). QALD tasks and linked data initiatives are contributing to this trend.

Our model combines the best of both worlds by inferring over structured and unstructured data. Though earlier methods exploited unstructured data for KB-QA (Krishnamurthy and Mitchell, 2012; Berant et al., 2013; Yao and Van Durme, 2014; Reddy et al., 2014; Yih et al., 2015), these methods do not rely on unstructured data at test time. Our work is closely related to Joshi et al. (2014) who aim to answer noisy telegraphic queries using both structured and unstructured data. Their work is limited in answering single relation queries. Our work also has similarities to Sun et al. (2015) who does question answering on unstructured data but enrich it with Freebase, a reversal of our pipeline. Other line of very recent related work include Yahya et al. (2016) and Savenkov and Agichtein (2016).

Our work also intersects with relation extraction methods. While these methods aim to predict a relation between two entities in order to pop-

ulate KBs (Mintz et al., 2009; Hoffmann et al., 2011; Riedel et al., 2013), we work with sentence level relation extraction for question answering. Krishnamurthy and Mitchell (2012) and Fader et al. (2014) adopt open relation extraction methods for QA but they require hand-coded grammar for parsing queries. Closest to our extraction method is Yao and Van Durme (2014) and Yao (2015) who also uses sentence level relation extraction for QA. Unlike them, we can predict multiple relations per question, and our MCCNN architecture is more robust to unseen contexts compared to their logistic regression models.

Dong et al. (2015) were the first to use MCCNN for question answering. Yet our approach is very different in spirit to theirs. Dong et al. aim to maximize the similarity between the distributed representation of a question and its answer entities, whereas our network aims to predict Freebase relations. Our search space is several times smaller than theirs since we do not require potential answer entities beforehand (the number of relations is much smaller than the number of entities in Freebase). In addition, our method can explicitly handle compositional questions involving multiple relations, whereas Dong et al. learn latent representation of relation joins which is difficult to comprehend. Moreover, we outperform their method by 7 points even without unstructured inference.

7 Conclusion and Future Work

We have presented a method that could infer both on structured and unstructured data to answer natural language questions. Our experiments reveal that unstructured inference helps in mitigating representational issues in structured inference. We have also introduced a relation extraction method using MCCNN which is capable of exploiting syntax in addition to sentential features. Our main model which uses joint entity linking and relation extraction along with unstructured inference achieves the state-of-the-art results on WebQuestions dataset. A potential application of our method is to improve KB-question answering using the documents retrieved by a search engine.

Since we pipeline structured inference first and then unstructured inference, our method is limited by the coverage of Freebase. Our future work involves exploring other alternatives such as treating structured and unstructured data as two independent resources in order to overcome the knowledge gaps in either of the two resources.

Appendix

The syntax-based patterns for question decomposition are shown in Figure 3. The first four patterns are designed to extract sub-questions from simple questions, while the latter two are designed for complex questions involving clauses.
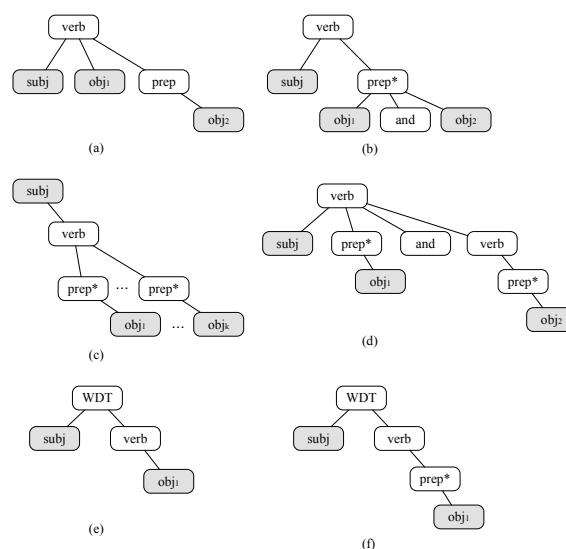

Figure 3: Syntax-based patterns for question decomposition.

References

Sren Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary G. Ives. 2007. Dbpedia: A nucleus for a web of open data. In *ISWC/ASWC*.

Michele Banko, Michael J Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. 2007. Open information extraction for the web. In *IJCAI*.

Junwei Bao, Nan Duan, Ming Zhou, and Tiejun Zhao. 2014. Knowledge-based question answering as machine translation. In *ACL*.

Hannah Bast and Elmar Haussmann. 2015. More accurate question answering on freebase. In *CIKM*.

Jonathan Berant and Percy Liang. 2014. Semantic parsing via paraphrasing. In *ACL*.

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

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *EMNLP*.

Kurt D. Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD*.

Antoine Bordes, Sumit Chopra, and Jason Weston. 2014. Question answering with subgraph embeddings. In *EMNLP*.

Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale simple question answering with memory networks. *CoRR*, abs/1506.02075.

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

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka Jr, and Tom M Mitchell. 2010. Toward an architecture for never-ending language learning. In *AAAI*.

Chih-Chung Chang and Chih-Jen Lin. 2011. LIB-SVM: A library for support vector machines. *ACM TIST*, 2(3):27.

Xiao Cheng and Dan Roth. 2013. Relational inference for wikification. In *ACL*.

Li Dong, Furu Wei, Ming Zhou, and Ke Xu. 2015. Question answering over freebase with multi-column convolutional neural networks. In *ACL-IJCNLP*.

John C. Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159.

Anthony Fader, Luke S. Zettlemoyer, and Oren Etzioni. 2013. Paraphrase-driven learning for open question answering. In *ACL*.

Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. 2014. Open question answering over curated and extracted knowledge bases. In *SIGKDD*.

Michael Heilman and Noah A Smith. 2010. Tree edit models for recognizing textual entailments, paraphrases, and answers to questions. In *NAACL*.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*.

Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *ACL*.

Thorsten Joachims. 2006. Training linear svms in linear time. In *SIGKDD*.

Mandar Joshi, Uma Sawant, and Soumen Chakrabarti. 2014. Knowledge graph and corpus driven segmentation and answer inference for telegraphic entity-seeking queries. In *EMNLP*.

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

Tom Kwiatkowski, Eunsol Choi, Yoav Artzi, and Luke S. Zettlemoyer. 2013. Scaling semantic parsers with on-the-fly ontology matching. In *EMNLP*.

Yang Liu, Furu Wei, Sujian Li, Heng Ji, Ming Zhou, and Houfeng WANG. 2015. A dependency-based neural network for relation classification. In *ACL*.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David Mc-Closky. 2014. The Stanford CoreNLP natural language processing toolkit. In *ACL System Demonstrations*.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *ACL*.

Ankur P. Parikh, Hoifung Poon, and Kristina Toutanova. 2015. Grounded semantic parsing for complex knowledge extraction. In *NAACL*.

Siva Reddy, Mirella Lapata, and Mark Steedman. 2014. Large-scale semantic parsing without question-answer pairs. *Transactions of the Association of Computational Linguistics*, pages 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.

Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M. Marlin. 2013. Relation extraction with matrix factorization and universal schemas. In *NAACL*.

Denis Savenkov and Eugene Agichtein. 2016. When a knowledge base is not enough: Question answering over knowledge bases with external text data. In *SIGIR*.

Iulian Vlad Serban, Alberto García-Durán, Çaglar Gülçehre, Sungjin Ahn, Sarath Chandar, Aaron C. Courville, and Yoshua Bengio. 2016. Generating factoid questions with recurrent neural networks: The 30m factoid question-answer corpus. In *ACL*.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *WWW*.

Huan Sun, Hao Ma, Wen-tau Yih, Chen-Tse Tsai, Jingjing Liu, and Ming-Wei Chang. 2015. Open domain question answering via semantic enrichment. In *WWW*.

Joseph P. Turian, Lev-Arie Ratinov, and Yoshua Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *ACL 2010, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, July 11-16, 2010, Uppsala, Sweden*, pages 384–394.

Christina Unger, Lorenz Bühmann, Jens Lehmann, Axel-Cyrille Ngonga Ngomo, Daniel Gerber, and Philipp Cimiano. 2012. Template-based question answering over rdf data. In *WWW*.

Ellen M Voorhees and Dawn M. Tice. 1999. The trec-8 question answering track report. In *TREC*.

Mengqiu Wang, Noah A Smith, and Teruko Mitamura. 2007. What is the jeopardy model? a quasi-synchronous grammar for qa. In *EMNLP-CoNLL*.

Yushi Wang, Jonathan Berant, and Percy Liang. 2015. Building a semantic parser overnight. In *ACL*.

Kun Xu, Sheng Zhang, Yansong Feng, and Dongyan Zhao. 2014. Answering natural language questions via phrasal semantic parsing. In *Natural Language Processing and Chinese Computing - Third CCF Conference, NLPCC 2014, Shenzhen, China, December 5-9, 2014. Proceedings*, pages 333–344.

Kun Xu, Yansong Feng, Songfang Huang, and Dongyan Zhao. 2015. Semantic relation classification via convolutional neural networks with simple negative sampling. In *EMNLP*.

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

Mohamed Yahya, Denilson Barbosa, Klaus Berberich, Qiuyue Wang, and Gerhard Weikum. 2016. Relationship queries on extended knowledge graphs. In *WSDM*.

Yi Yang and Ming-Wei Chang. 2015. S-mart: Novel tree-based structured learning algorithms applied to tweet entity linking. In *ACL-IJNLP*.

Yi Yang, Wen-tau Yih, and Christopher Meek. 2015. Wikiqa: A challenge dataset for open-domain question answering. In *EMNLP*.

Xuchen Yao and Benjamin Van Durme. 2014. Information extraction over structured data: Question answering with freebase. In *ACL*.

Xuchen Yao, Benjamin Van Durme, and Peter Clark. 2013. Answer extraction as sequence tagging with tree edit distance. In *NAACL*.

Xuchen Yao. 2015. Lean question answering over freebase from scratch. In *NAACL*.

Wen-tau Yih, Ming-Wei Chang, Christopher Meek, and Andrzej Pastusiak. 2013. Question answering using enhanced lexical semantic models. In *ACL*.

Wen-tau Yih, Xiaodong He, and Christopher Meek. 2014. Semantic parsing for single-relation question answering. In *ACL*.

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 *ACL-IJCNLP*.

Lei Yu, Karl Moritz Hermann, Phil Blunsom, and Stephen Pulman. 2014. Deep learning for answer sentence selection. *arXiv preprint arXiv:1412.1632*.

John M Zelle and Raymond J Mooney. 1996. Learning to parse database queries using inductive logic programming. In *AAAI*.

Yuanzhe Zhang, Shizhu He, Kang Liu, and Jun Zhao. 2016. A joint model for question answering over multiple knowledge bases. In *AAAI*.