Question Answering with Subgraph Embeddings

Antoine Bordes
Facebook AI Research
112 avenue de Wagram,
Paris, France
abordes@fb.com

Sumit Chopra
Facebook AI Research
770 Broadway,
New York, USA
spchopra@fb.com

Jason Weston
Facebook AI Research
770 Broadway,
New York, USA
jase@fb.com

Abstract

This paper presents a system which learns to answer questions on a broad range of topics from a knowledge base using few hand-crafted features. Our model learns low-dimensional embeddings of words and knowledge base constituents; these representations are used to score natural language questions against candidate answers. Training our system using pairs of questions and structured representations of their answers, and pairs of question paraphrases, yields competitive results on a recent benchmark of the literature.

1 Introduction

Teaching machines how to automatically answer questions asked in natural language on any topic or in any domain has always been a long standing goal in Artificial Intelligence. With the rise of large scale structured knowledge bases (KBs), this problem, known as open-domain question answering (or open QA), boils down to being able to query efficiently such databases with natural language. These KBs, such as FREEBASE (Bollacker et al., 2008) encompass huge ever growing amounts of information and ease open QA by organizing a great variety of answers in a structured format. However, the scale and the difficulty for machines to interpret natural language still makes this task a challenging problem.

The state-of-the-art techniques in open QA can be classified into two main classes, namely, information retrieval based and semantic parsing based. Information retrieval systems first retrieve a broad set of candidate answers by querying the search API of KBs with a transformation of the question into a valid query and then use fine-grained detection heuristics to identify the exact answer (Kolomiyets and Moens, 2011; Unger et al., 2012; Yao and Van Durme, 2014). On the other hand, semantic parsing methods focus on the correct interpretation of the meaning of a question by a semantic parsing system. A correct interpretation converts a question into the exact database query that returns the correct answer. Interestingly, recent works (Berant et al., 2013; Kwiatkowski et al., 2013; Berant and Liang, 2014; Fader et al., 2014) have shown that such systems can be efficiently trained under indirect and imperfect supervision and hence scale to large-scale regimes, while bypassing most of the annotation costs.

Yet, even if both kinds of system have shown the ability to handle large-scale KBs, they still require experts to hand-craft lexicons, grammars, and KB schema to be effective. This non-negligible human intervention might not be generic enough to conveniently scale up to new databases with other schema, broader vocabularies or languages other than English. In contrast, (Fader et al., 2013) proposed a framework for open QA requiring almost no human annotation. Despite being an interesting approach, this method is outperformed by other competing methods. (Bordes et al., 2014b) introduced an embedding model, which learns low-dimensional vector representations of words and symbols (such as KBs constituents) and can be trained with even less supervision than the system of (Fader et al., 2013) while being able to achieve better prediction performance. However, this approach is only compared with (Fader et al., 2013) which operates in a simplified setting and has not been applied in more realistic conditions nor evaluated against the best performing methods.

In this paper, we improve the model of (Bordes et al., 2014b) by providing the ability to answer more complicated questions. The main contributions of the paper are: (1) a more sophisticated inference procedure that is both efficient and can consider longer paths ((Bordes et al., 2014b) considered only answers directly connected to the

question in the graph); and (2) a richer representation of the answers which encodes the question-answer path and surrounding subgraph of the KB. Our approach is competitive with the current state-of-the-art on the recent benchmark WEBQUESTIONS (Berant et al., 2013) without using any lexicon, rules or additional system for part-of-speech tagging, syntactic or dependency parsing during training as most other systems do.

2 Task Definition

Our main motivation is to provide a system for open QA able to be trained as long as it has access to: (1) a training set of questions paired with answers and (2) a KB providing a structure among answers. We suppose that all potential answers are entities in the KB and that questions are sequences of words that include one identified KB entity. When this entity is not given, plain string matching is used to perform entity resolution. Smarter methods could be used but this is not our focus.

We use WEBQUESTIONS (Berant et al., 2013) as our evaluation bemchmark. Since it contains few training samples, it is impossible to learn on it alone, and this section describes the various data sources that were used for training. These are similar to those used in (Berant and Liang, 2014).

WebQuestions This dataset is built using FREEBASE as the KB and contains 5,810 question-answer pairs. It was created by crawling questions through the Google Suggest API, and then obtaining answers using Amazon Mechanical Turk. We used the original split (3,778 examples for training and 2,032 for testing), and isolated 1k questions from the training set for validation. WEBQUESTIONS is built on FREEBASE since all answers are defined as FREEBASE entities. In each question, we identified one FREEBASE entity using string matching between words of the question and entity names in FREEBASE. When the same string matches multiple entities, only the entity appearing in most triples, i.e. the most popular in FREEBASE, was kept. Example questions (answers) in the dataset include *"Where did Edgar Allan Poe died?"* (`baltimore`) or *"What degrees did Barack Obama get?"* (`bachelor_of_arts`, `juris_doctor`).

Freebase FREEBASE (Bollacker et al., 2008) is a huge and freely available database of general facts; data is organized as triplets (`subject`, `type1.type2.predicate`, `object`), where two entities `subject` and `object` (identified by `mids`) are connected by the relation type `type1.type2.predicate`. We used a subset, created by only keeping triples where one of the entities was appearing in either the WEBQUESTIONS training/validation set or in CLUEWEB extractions. We also removed all entities appearing less than 5 times and finally obtained a FREEBASE set containing 14M triples made of 2.2M entities and 7k relation types.[1] Since the format of triples does not correspond to any structure one could find in language, we decided to transform them into automatically generated questions. Hence, all triples were converted into questions "What is the `predicate` of the `type2 subject`?" (using the `mid` of the subject) with the answer being `object`. An example is *"What is the* `nationality` *of the* `person barack_obama`*?"* (`united_states`). More examples and details are given in a longer version of this paper (Bordes et al., 2014a).

ClueWeb Extractions FREEBASE data allows to train our model on 14M questions but these have a fixed lexicon and vocabulary, which is not realistic. Following (Berant et al., 2013), we also created questions using CLUEWEB extractions provided by (Lin et al., 2012). Using string matching, we ended up with 2M extractions structured as (`subject`, "text string", `object`) with both `subject` and `object` linked to FREEBASE. We also converted these triples into questions by using simple patterns and FREEBASE types. An example of generated question is *"Where* `barack_obama` *was allegedly bear in?"* (`hawaii`).

Paraphrases The automatically generated questions that are useful to connect FREEBASE triples and natural language, do not provide a satisfactory modeling of natural language because of their semi-automatic wording and rigid syntax. To overcome this issue, we follow (Fader et al., 2013) and supplement our training data with an indirect supervision signal made of pairs of question paraphrases collected from the WIKIANSWERS website. On WIKIANSWERS, users can tag pairs of questions as rephrasings of each other: (Fader et al., 2013) harvested a set of 2M distinct questions from WIKIANSWERS, which were grouped into 350k paraphrase clusters.

[1] WEBQUESTIONS contains ∼2k entities, hence restricting FREEBASE to 2.2M entities does not ease the task for us.

3 Embedding Questions and Answers

Inspired by (Bordes et al., 2014b), our model works by learning low-dimensional vector embeddings of words appearing in questions and of entities and relation types of FREEBASE, so that representations of questions and of their corresponding answers are close to each other in the joint embedding space. Let q denote a question and a a candidate answer. Learning embeddings is achieved by learning a scoring function $S(q, a)$, so that S generates a high score if a is the correct answer to the question q, and a low score otherwise. Note that both q and a are represented as a combination of the embeddings of their individual words and/or symbols; hence, learning S essentially involves learning these embeddings. In our model, the form of the scoring function is:

$$S(q, a) = f(q)^\top g(a). \quad (1)$$

Let $\mathbf{W}$ be a matrix of $\mathbb{R}^{k \times N}$, where k is the dimension of the embedding space which is fixed apriori, and N is the dictionary of embeddings to be learned. Let N_W denote the total number of words and N_S the total number of entities and relation types. With $N = N_W + N_S$, the i-th column of $\mathbf{W}$ is the embedding of the i-th element (word, entity or relation type) in the dictionary. The function $f(.)$, which maps the questions into the embedding space $\mathbb{R}^k$ is defined as $f(q) = \mathbf{W}\phi(q)$, where $\phi(q) \in \mathbb{N}^N$, is a sparse vector indicating the number of times each word appears in the question q (usually 0 or 1). Likewise the function $g(.)$ which maps the answer into the same embedding space $\mathbb{R}^k$ as the questions, is given by $g(a) = \mathbf{W}\psi(a)$. Here $\psi(a) \in \mathbb{N}^N$ is a sparse vector representation of the answer a, which we now detail.

3.1 Representing Candidate Answers

We now describe possible feature representations for a single candidate answer. (When there are multiple correct answers, we average these representations, see Section 3.4.) We consider three different types of representation, corresponding to different subgraphs of FREEBASE around it.

(i) *Single Entity*. The answer is represented as a single entity from FREEBASE: $\psi(a)$ is a 1-of-N_S coded vector with 1 corresponding to the entity of the answer, and 0 elsewhere.

(ii) *Path Representation*. The answer is represented as a path from the entity mentioned in the question to the answer entity. In our experiments, we considered 1- or 2-hops paths (i.e. with either 1 or 2 edges to traverse): (`barack_obama`, `people.person.place_of_birth`, `honolulu`) is a 1-hop path and (`barack_obama`, `people.person.place_of_birth`, `location.location.containedby`, `hawaii`) a 2-hops path. This results in a $\psi(a)$ which is a 3-of-N_S or 4-of-N_S coded vector, expressing the start and end entities of the path and the relation types (but not entities) in-between.

(iii) *Subgraph Representation*. We encode both the path representation from (ii), and the entire subgraph of entities connected to the candidate answer entity. That is, for each entity connected to the answer we include both the relation type and the entity itself in the representation $\psi(a)$. In order to represent the answer path differently to the surrounding subgraph (so the model can differentiate them), we double the dictionary size for entities, and use one embedding representation if they are in the path and another if they are in the subgraph. Thus we now learn a parameter matrix $\mathbb{R}^{k \times N}$ where $N = N_W + 2N_S$ (N_S is the total number of entities and relation types). If there are C connected entities with D relation types to the candidate answer, its representation is a $3+C+D$ or $4+C+D$-of-N_S coded vector, depending on the path length.

Our hypothesis is that including more information about the answer in its representation will lead to improved results. While it is possible that all required information could be encoded in the k dimensional embedding of the single entity *(i)*, it is unclear what dimension k should be to make this possible. For example the embedding of a country entity encoding all of its citizens seems unrealistic. Similarly, only having access to the path ignores all the other information we have about the answer entity, unless it is encoded in the embeddings of either the entity of the question, the answer or the relations linking them, which might be quite complicated as well. We thus adopt the subgraph approach. Figure 1 illustrates our model.

3.2 Training and Loss Function

As in (Weston et al., 2010), we train our model using a margin-based ranking loss function. Let $\mathcal{D} = \{(q_i, a_i) : i = 1, \ldots, |\mathcal{D}|\}$ be the training set

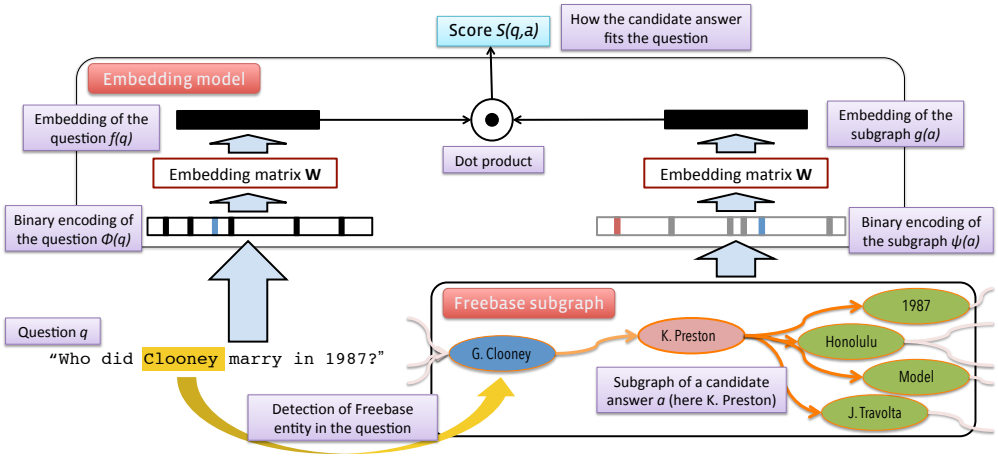

Figure 1: Illustration of the subgraph embedding model scoring a candidate answer: (i) locate entity in the question; (ii) compute path from entity to answer; (iii) represent answer as path plus all connected entities to the answer (the subgraph); (iv) embed both the question and the answer subgraph separately using the learnt embedding vectors, and score the match via their dot product.

of questions q_i paired with their correct answer a_i. The loss function we minimize is

$$\sum_{i=1}^{|\mathcal{D}|} \sum_{\bar{a} \in \bar{\mathcal{A}}(a_i)} \max\{0, m - S(q_i, a_i) + S(q_i, \bar{a})\}, \quad (2)$$

where m is the margin (fixed to 0.1). Minimizing Eq. (2) learns the embedding matrix $\mathbf{W}$ so that the score of a question paired with a correct answer is greater than with any incorrect answer $\bar{a}$ by at least m. $\bar{a}$ is sampled from a set of incorrect candidates $\bar{\mathcal{A}}$. This is achieved by sampling 50% of the time from the set of entities connected to the entity of the question (i.e. other candidate paths), and by replacing the answer entity by a random one otherwise. Optimization is accomplished using stochastic gradient descent, multi-threaded with Hogwild! (Recht et al., 2011), with the constraint that the columns w_i of $\mathbf{W}$ remain within the unit-ball, i.e., $\forall_i, ||w_i||_2 \leq 1$.

3.3 Multitask Training of Embeddings

Since a large number of questions in our training datasets are synthetically generated, they do not adequately cover the range of syntax used in natural language. Hence, we also multi-task the training of our model with the task of paraphrase prediction. We do so by alternating the training of S with that of a scoring function $S_{prp}(q_1, q_2) = f(q_1)^\top f(q_2)$, which uses the same embedding matrix $\mathbf{W}$ and makes the embeddings of a pair of questions (q_1, q_2) similar to each other if they are paraphrases (i.e. if they belong to the same paraphrase cluster), and make them different otherwise. Training S_{prp} is similar to that of S except that negative samples are obtained by sampling a question from another paraphrase cluster.

We also multitask the training of the embeddings with the mapping of the `mids` of FREEBASE entities to the actual words of their names, so that the model learns that the embedding of the `mid` of an entity should be similar to the embedding of the word(s) that compose its name(s).

3.4 Inference

Once $\mathbf{W}$ is trained, at test time, for a given question q the model predicts the answer with:

$$\hat{a} = \text{argmax}_{a' \in \mathcal{A}(q)} S(q, a') \quad (3)$$

where $\mathcal{A}(q)$ is the candidate answer set. This candidate set could be the whole KB but this has both speed and potentially precision issues. Instead, we create a candidate set $\mathcal{A}(q)$ for each question.

We recall that each question contains one identified FREEBASE entity. $\mathcal{A}(q)$ is first populated with all triples from FREEBASE involving this entity. This allows to answer simple factual questions whose answers are directly connected to them (i.e. 1-hop paths). This strategy is denoted C_1.

Since a system able to answer only such questions would be limited, we supplement $\mathcal{A}(q)$ with examples situated in the KB graph at 2-hops from the entity of the question. We do not add all such quadruplets since this would lead to very large candidate sets. Instead, we consider the following general approach: given that we are predicting a path, we can predict its elements in turn using

Method	P@1 (%)	F1 (Berant)	F1 (Yao)
Baselines			
(Berant et al., 2013)	–	31.4	–
(Bordes et al., 2014b)	31.3	29.7	31.8
(Yao and Van Durme, 2014)	–	33.0	42.0
(Berant and Liang, 2014)	–	39.9	43.0
Our approach			
Subgraph & $\mathcal{A}(q) = C_2$	**40.4**	39.2	43.2
Ensemble with (Berant & Liang, 14)	–	**41.8**	**45.7**
Variants			
Without multiple predictions	**40.4**	31.3	34.2
Subgraph & $\mathcal{A}(q) =$ All 2-hops	38.0	37.1	41.4
Subgraph & $\mathcal{A}(q) = C_1$	34.0	32.6	35.1
Path & $\mathcal{A}(q) = C_2$	36.2	35.3	38.5
Single Entity & $\mathcal{A}(q) = C_1$	25.8	16.0	17.8

Table 1: Results on the WEBQUESTIONS test set.

a beam search, and hence avoid scoring all candidates. Specifically, our model first ranks relation types using Eq. (1), i.e. selects which relation types are the most likely to be expressed in q. We keep the top 10 types (10 was selected on the validation set) and only add 2-hops candidates to $\mathcal{A}(q)$ when these relations appear in their path. Scores of 1-hop triples are weighted by 1.5 since they have one less element than 2-hops quadruplets. This strategy, denoted C_2, is used by default.

A prediction a' can commonly actually be a set of candidate answers, not just one answer, for example for questions like *"Who are David Beckham's children?"*. This is achieved by considering a prediction to be all the entities that lie on the same 1-hop or 2-hops path from the entity found in the question. Hence, all answers to the above question are connected to `david_beckham` via the same path (`david_beckham`, `people.person.children`, $\star$). The feature representation of the prediction is then the average over each candidate entity's features (see Section 3.1), i.e. $\psi_{all}(a') = \frac{1}{|a'|}\sum_{a'_j:a'} \psi(a'_j)$ where a'_j are the individual entities in the overall prediction a'. In the results, we compare to a baseline method that can only predict single candidates, which understandly performs poorly.

4 Experiments

We compare our system in terms of F1 score as computed by the official evaluation script[2] (F1 (Berant)) but also with a slightly different F1 definition, termed F1 (Yao) which was used in (Yao and Van Durme, 2014) (the difference being the way that questions with no answers are dealt with), and precision @ 1 (p@1) of the first candidate entity (even when there are a set of correct answers), comparing to recently published systems.[3] The upper part of Table 1 indicates that our approach outperforms (Yao and Van Durme, 2014), (Berant et al., 2013) and (Bordes et al., 2014b), and performs similarly as (Berant and Liang, 2014).

The lower part of Table 1 compares various versions of our model. Our default approach uses the Subgraph representation for answers and C_2 as the candidate answers set. Replacing C_2 by C_1 induces a large drop in performance because many questions do not have answers thatare directly connected to their inluded entity (not in C_1). However, using all 2-hops connections as a candidate set is also detrimental, because the larger number of candidates confuses (and slows a lot) our ranking based inference. Our results also verify our hypothesis of Section 3.1, that a richer representation for answers (using the local subgraph) can store more pertinent information. Finally, we demonstrate that we greatly improve upon the model of (Bordes et al., 2014b), which actually corresponds to a setting with the Path representation and C_1 as candidate set.

We also considered an ensemble of our approach and that of (Berant and Liang, 2014). As we only had access to their test predictions we used the following combination method. Our approach gives a score $S(q, a)$ for the answer it predicts. We chose a threshold such that our approach predicts 50% of the time (when $S(q, a)$ is above its value), and the other 50% of the time we use the prediction of (Berant and Liang, 2014) instead. We aimed for a 50/50 ratio because both methods perform similarly. The ensemble improves the state-of-the-art, and indicates that our models are significantly different in their design.

5 Conclusion

This paper presented an embedding model that learns to perform open QA using training data made of questions paired with their answers and of a KB to provide a structure among answers, and can achieve promising performance on the competitive benchmark WEBQUESTIONS.

[2] Available from www-nlp.stanford.edu/software/sempre/

[3] Results of baselines except (Bordes et al., 2014b) have been extracted from the original papers. For our experiments, all hyperparameters have been selected on the WEBQUESTIONS validation set: k was chosen among $\{64, 128, 256\}$, the learning rate on a log. scale between 10^{-4} and 10^{-1} and we used at most 100 paths in the subgraph representation.

References

Jonathan Berant and Percy Liang. 2014. Semantic parsing via paraphrasing. In *Proceedings of the 52nd Annual Meeting of the ACL*.

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

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*. ACM.

Antoine Bordes, Sumit Chopra, and Jason Weston. 2014a. Question answering with subgraph embeddings. *CoRR*, abs/1406.3676.

Antoine Bordes, Jason Weston, and Nicolas Usunier. 2014b. Open question answering with weakly supervised embedding models. In *Proceedings of ECML-PKDD'14*. Springer.

Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. 2013. Paraphrase-driven learning for open question answering. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 1608–1618, Sofia, Bulgaria.

Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. 2014. Open question answering over curated and extracted knowledge bases. In *Proceedings of KDD'14*. ACM.

Oleksandr Kolomiyets and Marie-Francine Moens. 2011. A survey on question answering technology from an information retrieval perspective. *Information Sciences*, 181(24):5412–5434.

Tom Kwiatkowski, Eunsol Choi, Yoav Artzi, and Luke Zettlemoyer. 2013. Scaling semantic parsers with on-the-fly ontology matching. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, October.

Thomas Lin, Oren Etzioni, et al. 2012. Entity linking at web scale. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, pages 84–88. Association for Computational Linguistics.

Benjamin Recht, Christopher Ré, Stephen J Wright, and Feng Niu. 2011. Hogwild!: A lock-free approach to parallelizing stochastic gradient descent. In *Advances in Neural Information Processing Systems (NIPS 24)*.

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*.

Jason Weston, Samy Bengio, and Nicolas Usunier. 2010. Large scale image annotation: learning to rank with joint word-image embeddings. *Machine learning*, 81(1).

Xuchen Yao and Benjamin Van Durme. 2014. Information extraction over structured data: Question answering with freebase. In *Proceedings of the 52nd Annual Meeting of the ACL*.