Large-scale Simple Question Answering with Memory Networks

Antoine Bordes
Facebook AI Research
770 Broadway
New York, NY. USA
abordes@fb.com

Nicolas Usunier
Facebook AI Research
112, avenue de Wagram
75017 Paris, France
usunier@fb.com

Sumit Chopra, Jason Weston
Facebook AI Research
770 Broadway
New York, NY. USA
{spchopra, jase}@fb.com

Abstract

Training large-scale question answering systems is complicated because training sources usually cover a small portion of the range of possible questions. This paper studies the impact of multitask and transfer learning for *simple question answering*; a setting for which the reasoning required to answer is quite easy, as long as one can retrieve the correct evidence given a question, which can be difficult in large-scale conditions. To this end, we introduce a new dataset of 100k questions that we use in conjunction with existing benchmarks. We conduct our study within the framework of Memory Networks (Weston et al., 2015) because this perspective allows us to eventually scale up to more complex reasoning, and show that Memory Networks can be successfully trained to achieve excellent performance.

1 Introduction

Open-domain Question Answering (QA) systems aim at providing the exact answer(s) to questions formulated in natural language, without restriction of domain. While there is a long history of QA systems that search for textual documents or on the Web and extract answers from them (see e.g. (Voorhees and Tice, 2000; Dumais et al., 2002)), recent progress has been made with the release of large Knowledge Bases (KBs) such as Freebase, which contain consolidated knowledge stored as atomic facts, and extracted from different sources, such as free text, tables in webpages or collaborative input. Existing approaches for QA from KBs use learnable components to either transform the question into a structured KB query (Berant et al., 2013) or learn to embed questions and facts in a low dimensional vector space and retrieve the answer by computing similarities in this embedding space (Bordes et al., 2014a). However, while most recent efforts have focused on designing systems with higher reasoning capabilities, that could jointly retrieve and use multiple facts to answer, the simpler problem of answering questions that refer to a single fact of the KB, which we call *Simple Question Answering* in this paper, is still far from solved.

Hence, existing benchmarks are small; they mostly cover the head of the distributions of facts, and are restricted in their question types and their syntactic and lexical variations. As such, it is still unknown how much the existing systems perform outside the range of the specific question templates of a few, small benchmark datasets, and it is also unknown whether learning on a single dataset transfers well on other ones, and whether such systems can learn from different training sources, which we believe is necessary to capture the whole range of possible questions.

Besides, the actual need for reasoning, i.e. constructing the answer from more than a single fact from the KB, depends on the actual structure of the KB. As we shall see, for instance, a simple preprocessing of Freebase tremendously increases the coverage of simple QA in terms of possible questions that can be answered with a single fact, including list questions that expect more than a single answer. In fact, the task of simple QA itself might already cover a wide range of practical usages, if the KB is properly organized.

This paper presents two contributions. First, as an effort to study the coverage of existing systems and the possibility to train jointly on different data sources via multitasking, we collected the first large-scale dataset of questions and answers based on a KB, called SimpleQuestions. This dataset, which is presented in Section 2, contains more than 100k questions written by human anno-

What American cartoonist is the creator of Andy Lippincott?	(andy_lippincott, character_created_by, garry_trudeau)
Which forest is Fires Creek in?	(fires_creek, containedby, nantahala_national_forest)
What is an active ingredient in childrens earache relief ?	(childrens_earache_relief, active_ingredients, capsicum)
What does Jimmy Neutron do?	(jimmy_neutron, fictional_character_occupation, inventor)
What dietary restriction is incompatible with kimchi?	(kimchi, incompatible_with_dietary_restrictions, veganism)

Table 1: **Examples of simple QA**. Questions and corresponding facts have been extracted from the new dataset SimpleQuestions introduced in this paper. Actual answers are underlined.

tators and associated to Freebase facts, while the largest existing benchmark, WebQuestions, contains less than 6k questions created automatically using the Google suggest API.

Second, in sections 3 and 4, we present an embedding-based QA system developed under the framework of Memory Networks (MemNNs) (Weston et al., 2015; Sukhbaatar et al., 2015). Memory Networks are learning systems centered around a memory component that can be read and written to, with a particular focus on cases where the relationship between the input and response languages (here natural language) and the storage language (here, the facts from KBs) is performed by embedding all of them in the same vector space. The setting of the simple QA corresponds to the elementary operation of performing a single lookup in the memory. While our model bares similarity with previous embedding models for QA (Bordes et al., 2014b; Bordes et al., 2014a), using the framework of MemNNs opens the perspective to more involved inference schemes in future work, since MemNNs were shown to perform well on complex reasoning toy QA tasks (Weston et al., 2015). We discuss related work in Section 5.

We report experimental results in Section 6, where we show that our model achieves excellent results on the benchmark WebQuestions. We also show that it can learn from two different QA datasets to improve its performance on both. We also present the first successful application of transfer learning for QA. Using the Reverb KB and QA datasets, we show that Reverb facts can be added to the memory and used to answer *without retraining*, and that MemNNs achieve better results than some systems designed on this dataset.

2 Simple Question Answering

Knowledge Bases contain facts expressed as triples `(subject, relationship, object)`, where `subject` and `object` are entities and `relationship` describes the type of (directed) link between these entities. The simple QA problem we address here consist in finding the answer to questions that can be rephrased as queries of the form `(subject, relationship, ?)`, asking for all objects linked to `subject` by `relationship`. The question *What do Jamaican people speak ?*, for instance, could be rephrased as the Freebase query `(jamaica, language_spoken, ?)`. In other words, fetching a single fact from a KB is sufficient to answer correctly.

The term *simple QA* refers to the simplicity of the reasoning process needed to answer questions, since it involves a single fact. However, this does not mean that the QA problem is easy per se, since retrieving this single supporting fact can be very challenging as it involves to search over millions of alternatives given a query expressed in natural language. Table 1 shows that, with a KB with many types of relationships like Freebase, the range of questions that can be answered with a single fact is already very broad. Besides, as we shall see, modiying slightly the structure of the KB can *make some QA problems simpler* by adding direct connections between entities and hence allow to bypass the need for more complex reasoning.

2.1 Knowledge Bases

We use the KB Freebase[1] as the basis of our QA system, our source of facts and answers. All Freebase entities and relationships are typed and the lexicon for types and relationships is closed. Freebase data is collaboratively collected and curated, to ensure a high reliability of the facts. Each entity has an internal identifier and a set of strings that are usually used to refer to that entity in text, termed *aliases*. We consider two extracts of Freebase, whose statistics are given in Table 2. FB2M, which was used in (Bordes et al., 2014a), contains about 2M entities and 5k relationships. FB5M, is much larger with about 5M entities and more than 7.5k relationships.

We also use the KB Reverb as a secondary source of facts to study how well a model trained to answer questions using Freebase facts could

[1] `www.freebase.com`

	FB2M	FB5M	Reverb
ENTITIES	2,150,604	4,904,397	2,044,752
RELATIONSHIPS	6,701	7,523	601,360
ATOMIC FACTS	14,180,937	22,441,880	14,338,214
FACTS (grouped)	10,843,106	12,010,500	–

Table 2: **Knowledge Bases** used in this paper. FB2M and FB5M are two versions of Freebase.

be used to answer using Reverb's as well, without being trained on Reverb data. This is a pure setting of *transfer learning*. Reverb is interesting for this experiment because it differs a lot from Freebase. Its data was extracted automatically from text with minimal human intervention and is highly unstructured: entities are unique strings and the lexicon for relationships is open. This leads to many more relationships, but entities with multiple references are not deduplicated, ambiguous referents are not resolved, and the reliability of the stored facts is much lower than in Freebase. We used the full extraction from (Fader et al., 2011), which contains 2M entities and 600k relationships.

2.2 The SimpleQuestions dataset

Existing resources for QA such as WebQuestions (Berant et al., 2013) are rather small (few thousands questions) and hence do not provide a very thorough coverage of the variety of questions that could be answered using a KB like Freebase, even in the context of simple QA. Hence, in this paper, we introduce a new dataset of much larger scale for the task of simple QA called SimpleQuestions.[2] This dataset consists of a total of 108,442 questions written in natural language by human English-speaking annotators each paired with a corresponding fact from FB2M that provides the answer and explains it. We randomly shuffle these questions and use 70% of them (75910) as training set, 10% as validation set (10845), and the remaining 20% as test set. Examples of questions and facts are given in Table 1.

We collected SimpleQuestions in two phases. The first phase consisted of shortlisting the set of facts from Freebase to be annotated with questions. We used FB2M as background KB and removed all facts with undefined relationship type i.e. containing the word `freebase`. We also removed all facts for which the (subject, relationship) pair had more than a threshold number of objects. This filtering step is crucial to remove facts which would result in trivial uninformative questions, such as, *Name a person who is an actor?*. The threshold was set to 10.

In the second phase, these selected facts were sampled and delivered to human annotators to generate questions from them. For the sampling, each fact was associated with a probability which defined as a function of its relationship frequency in the KB: to favor variability, facts with relationship appearing more frequently were given lower probabilities. For each sampled facts, annotators were shown the facts along with hyperlinks to `freebase.com` to provide some context while framing the question. Given this information, annotators were asked to phrase a question involving the subject and the relationship of the fact, with the answer being the object. The annotators were explicitly instructed to phrase the question differently as much as possible, if they encounter multiple facts with similar relationship. They were also given the option of skipping facts if they wish to do so. This was very important to avoid the annotators to write a boiler plate questions when they had no background knowledge about some facts.

3 Memory Networks for Simple QA

A Memory Network consists of a memory (an indexed array of objects) and a neural network that is trained to query it given some inputs (usually questions). It has four components: *Input map* (I), *Generalization* (G), *Output map* (O) and *Response* (R) which we detail below. But first, we describe the MemNNs workflow used to set up a model for simple QA. This proceeds in three steps:

1. Storing Freebase: this first phase parses Freebase (either FB2M or FB5M depending on the setting) and stores it in memory. It uses the *Input* module to preprocess the data.

2. Training: this second phase trains the MemNN to answer question. This uses *Input*, *Output* and *Response* modules, the training concerns mainly the parameters of the embedding model at the core of the *Output* module.

3. Connecting Reverb: this third phase adds new facts coming from Reverb to the memory. This is done after training to test the ability of MemNNs to handle new facts without having to be re-trained. It uses the *Input* module to preprocess Reverb facts and the *Generalization* module to connect them to the facts already stored.

[2] The dataset is available from `http://fb.ai/babi`.

After these three stages, the MemNN is ready to answer any question by running the I, O and R modules in turn. We now detail the implementation of the four modules.

3.1 Input module

This module preprocesses the three types of data that are input to the network: Freebase facts that are used to populate the memory, questions that the system need to answer, and Reverb facts that we use, in a second phase, to extend the memory.

Preprocessing Freebase The Freebase data is initially stored as atomic facts involving single entities as subject and object, plus a relationship between them. However, this storage needs to be adapted to the QA task in two aspects.

First, in order to answer list questions, which expect more than one answer, we redefine a fact as being a triple containing a subject, a relationship, and the set of all objects linked to the subject by the relationship. This *grouping* process transforms atomic facts into grouped facts, which we simply refer to as *facts* in the following. Table 2 shows the impact of this grouping: on FB2M, this decreases the number of facts from 14M to 11M and, on FB5M, from 22M to 12M.

Second, the underlying structure of Freebase is a hypergraph, in which more than two entities can be linked. For instance dates can be linked together with two entities to specify the time period over which the link was valid. The underlying triple storage involves *mediator nodes* for each such fact, effectively making entities linked through paths of length 2, instead of 1. To obtain direct links between entities in such cases, we created a single fact for these facts by removing the intermediate node and using the second relationship as the relationship for the new condensed fact. This step reduces the need for searching the answer outside the immediate neighborhood of the subject referred to in the question, widely increasing the scope of the simple QA task on Freebase. On WebQuestions, a benchmark not primarily designed for simple QA, removing mediator nodes allows to jump from around 65% to 86% of questions that can be answered with a single fact.

Preprocessing Freebase facts A fact with k objects $y = (s, r, \{o_1, ..., o_k\})$ is represented by a bag-of-symbol vector $f(y)$ in $\mathbb{R}^{N_S}$, where N_S is the number of entities and relationships. Each dimension of $f(y)$ corresponds to a relationship or an entity (independent of whether it appears as subject or object). The entries of the subject and of the relationship have value 1, and the entries of the objects are set to $1/k$. All other entries are 0.

Preprocessing questions A question q is mapped to a bag-of-ngrams representation $g(q)$ of dimension $\mathbb{R}^{N_V}$ where N_V is the size of the vocabulary. The vocabulary contains all individual words that appear in the questions of our datasets, together with the aliases of Freebase entities, each alias being a single n-gram. The entries of $g(q)$ that correspond to words and n-grams of q are equal to 1, all other ones are set to 0.

Preprocessing Reverb facts In our experiments with Reverb, each fact $y = (s, r, o)$ is represented as a vector $h(y) \in \mathbb{R}^{N_S+N_V}$. This vector is a bag-of-symbol for the subject s and the object o, and a bag-of-words for the relationship r. The exact composition of h is provided by the *Generalization* module, which we describe now.

3.2 Generalization module

This module is responsible for adding new elements to the memory. In our case, the memory has a multigraph structure where each node is a Freebase entity and labeled arcs in the multigraph are Freebase relationships: after their preprocessing, all Freebase facts are stored using this structure.

We also consider the case where new facts, with a different structure (i.e. new kinds of relationship), are provided to the MemNNs by using Reverb. In this case, the generalization module is then used to connect Reverb facts to the Freebase-based memory structure, in order to make them usable and searchable by the MemNN.

To link the subject and the object of a Reverb fact to Freebase entities, we use precomputed entity links (Lin et al., 2012). If such links do not give any result for an entity, we search for Freebase entities with at least one alias that matches the Reverb entity string. These two processes allowed to match 17% of Reverb entities to Freebase ones. The remainder of entities were encoded using bag-of-words representation of their strings, since we had no other way of matching them to Freebase entities. All Reverb relationships were encoded using bag-of-words of their strings. Using this approximate process, we are able to store each Reverb fact as a bag-of-symbols (words or Freebase entities) all already seen by the MemNN during its training phase based on

Freebase. We can then hope that what had been learned there could also be successfully used to query Reverb facts.

3.3 Output module

The output module performs the memory lookups given the input to return the *supporting facts* destined to eventually provide the answer given a question. In our case of simple QA, this module only returns a single supporting fact. To avoid scoring all the stored facts, we first perform an approximate entity linking step to generate a small set of candidate facts. The supporting fact is the candidate fact that is most similar to the question according to an embedding model.

Candidate generation To generate candidate facts, we match n-grams of words of the question to aliases of Freebase entities and select a few matching entities. All facts having one of these entities as subject are scored in a second step.

We first generate all possible n-grams from the question, removing those that contain an interrogative pronoun or 1-grams that belong to a list of stopwords. We only keep the n-grams which are an alias of an entity, and then discard all n-grams that are a subsequence of another n-gram, except if the longer n-gram only differs by *in*, *of*, *for* or *the* at the beginning. We finally keep the two entities with the most links in Freebase retrieved for each of the five longest matched n-grams.

Scoring Scoring is performed using an embedding model. Given two embedding matrices $\mathbf{W}_V \in \mathbb{R}^{d \times N_V}$ and $\mathbf{W}_S \in \mathbb{R}^{d \times N_S}$, which respectively contain, in columns, the d-dimensional embeddings of the words/n-grams of the vocabulary and the embeddings of the Freebase entities and relationships, the similarity between question q and a Freebase candidate fact y is computed as:

$$S_{QA}(q, y) = \cos(\mathbf{W}_V g(q), \mathbf{W}_S f(y)) \,,$$

with $\cos()$ the cosine similarity. When scoring a fact y from Reverb, we use the same embeddings and build the matrix $\mathbf{W}_{VS} \in \mathbb{R}^{d \times (N_V + N_S)}$, which contains the concatenation in columns of $\mathbf{W}_V$ and $\mathbf{W}_S$, and also compute the cosine similarity:

$$S_{RVB}(q, y) = \cos(\mathbf{W}_V g(q), \mathbf{W}_{VS} h(y)) \,.$$

The dimension d is a hyperparameter, and the embedding matrices $\mathbf{W}_V$ and $\mathbf{W}_S$ are the parameters learned with the training algorithm of Section 4.

3.4 Response module

In Memory Networks, the *Response* module post-processes the result of the *Output* module to compute the intended answer. In our case, it returns the set of objects of the selected supporting fact.

4 Training

This section details how we trained the scoring function of the *Output* module using a multitask training process on four different sources of data.

First, in addition to the new SimpleQuestions dataset described in Section 2, we also used WebQuestions, a benchmark for QA introduced in (Berant et al., 2013): questions are labeled with answer strings from aliases of Freebase entities, and many questions expect multiple answers. Table 3 details the statistics of both datasets.

We also train on automatic questions generated from the KB, that is FB2M or FB5M depending on the setting, which are essential to learn embeddings for the entities not appearing in either WebQuestions or SimpleQuestions. Statistics of FB2M or FB5M are given in Table 2; we generated one training question per fact following the same process as that used in (Bordes et al., 2014a).

Following previous work such as (Fader et al., 2013), we also use the indirect supervision signal of pairs of question paraphrases. We used a subset of the large set of paraphrases extracted from WIKIANSWERS and introduced in (Fader et al., 2014). Our Paraphrases dataset is made of 15M clusters containing 2 or more paraphrases each.

4.1 Multitask training

As in previous work on embedding models and Memory Networks (Bordes et al., 2014a; Bordes et al., 2014b; Weston et al., 2015), the embeddings are trained with a ranking criterion. For QA datasets the goal is that in the embedding space, a supporting fact is more similar to the question than any other *non-supporting* fact. For the paraphrase dataset, a question should be more similar to one of its paraphrases than to any another question.

The multitask learning of the embedding matrices $\mathbf{W}_V$ and $\mathbf{W}_S$ is performed by alternating stochastic gradient descent (SGD) steps over the loss function on the different datasets. For the QA datasets, given a question/supporting fact pair

(q, y) and a non-supporting fact y', we perform a step to minimize the loss function

$$\ell_{QA}(q, y, y') = \left[\gamma - S_{QA}(q, y) + S_{QA}(q, y')\right]_+ ,$$

where $[.]_+$ is the positive part and γ is a margin hyperparameter. For the paraphrase dataset, the similarity score between two questions q and q' is also the cosine between their embeddings, i.e. $S_{QQ}(q, q') = \cos(\mathbf{W}_V g(q), \mathbf{W}_V g(q'))$, and given a paraphrase pair (q, q') and another question q'', the loss is:

$$\ell_{QQ}(q, q', q'') = \left[\gamma - S_{QQ}(q, q') + S_{QQ}(q, q'')\right]_+ .$$

The embeddings (i.e. the columns of $\mathbf{W}_V$ and $\mathbf{W}_S$) are projected onto the L_2 unit ball after each update. At each time step, a sample from the paraphrase dataset is drawn with probability 0.2 (this probability is arbitrary). Otherwise, a sample from one of the three QA datasets, chosen uniformly at random, is taken. We use the WARP loss (Weston et al., 2010) to speed up training, and Adagrad (Duchi et al., 2011) as SGD algorithm multi-threaded with `HogWild!` (Recht et al., 2011). Training takes 2-3 hours on 20 threads.

4.2 Distant supervision

Unlike for SimpleQuestions or the synthetic QA data generated from Freebase, for WebQuestions only answer strings are provided for questions: the supporting facts are unknown.

In order to generate the supervision, we use the candidate fact generation algorithm of Section 3.3. For each candidate fact, the aliases of its objects are compared to the set of provided answer strings. The fact(s) which can generate the maximum number of answer strings from their objects' aliases are then kept. If multiple facts are obtained for the same question, the ones with the minimal number of objects are considered as supervision facts. This last selection avoids favoring irrelevant relationships that would be kept only because they point to many objects but would not be specific enough. If no answer string could be found from the objects of the initial candidates, the question is discarded from the training set.

Future work should investigate the process of weak supervised training of MemNNs recently introduced in (Sukhbaatar et al., 2015) that allows to train them without any supervision coming from the supporting facts.

	WebQuestions	SimpleQuestions	Reverb
TRAIN	3,000	75,910	–
VALID.	778	10,845	–
TEST	2,032	21,687	691

Table 3: **Training and evaluation datasets**. Questions automatically generated from the KB and paraphrases can also be used in training.

4.3 Generating negative examples

As in (Bordes et al., 2014a; Bordes et al., 2014b), learning is performed with gradient descent, so that negative examples (non-supporting facts or non-paraphrases) are generated according to a randomized policy during training.

For paraphrases, given a pair (q, q'), a non-paraphrase pair is generated as (q, q'') where q'' is a random question of the dataset, not belonging to the cluser of q. For question/supporting fact pairs, we use two policies. The default policy to obtain a non-supporting fact is to corrupt the answer fact by exchanging its subject, its relationship or its object(s) with that of another fact chosen uniformly at random from the KB. In this policy, the element of the fact to corrupt is chosen randomly, with a small probability (0.3) of corrupting more than one element of the answer fact. The second policy we propose, called *candidates as negatives*, is to take as non-supporting fact a randomly chosen fact from the set of candidate facts. While the first policy is standard in learning embeddings, the second one is more original, and, as we see in the experiments, gives slightly better performance.

5 Related Work

The first approaches to open-domain QA were search engine-based systems, where keywords extracted from the question are sent to a search engine, and the answer is extracted from the top results (Yahya et al., 2012; Unger et al., 2012). This method has been adapted to KB-based QA (Yahya et al., 2012; Unger et al., 2012), and obtained competitive results with respect to semantic parsing and embedding-based approaches.

Semantic parsing approaches (Cai and Yates, 2013; Berant et al., 2013; Kwiatkowski et al., 2013; Berant and Liang, 2014; Fader et al., 2014) perform a functional parse of the sentence that can be interpreted as a KB query. Even though these approaches are difficult to train at scale

because of the complexity of their inference, their advantage is to provide a deep interpretation of the question. Some of these approaches require little to no question-answer pairs (Fader et al., 2013; Reddy et al., 2014), relying on simple rules to tranform the semantic interpretation to a KB query.

Like our work, embedding-based methods for QA can be seen as simple MemNNs. The algorithms of (Bordes et al., 2014b; Weston et al., 2015) use an approach similar to ours but are based on Reverb rather than Freebase, and relied purely on bag-of-word for both questions and facts. The approach of (Yang et al., 2014) uses a different representation of questions, in which recognized entities are replaced by an *entity* token, and a different training data using entity mentions from WIKIPEDIA. Our model is closest to the one presented in (Bordes et al., 2014a), which is discussed in more details in the experiments.

6 Experiments

This section provides an extensive evaluation of our MemNNs implementation against state-of-the-art QA methods as well as an empirical study of the impact of using multiple training sources on the prediction performance.

6.1 Evaluation and baselines

Table 3 details the dimensions of the test sets of WebQuestions, SimpleQuestions and Reverb which we used for evaluation. On WebQuestions, we evaluate against previous results on this benchmark (Berant et al., 2013; Yao and Van Durme, 2014; Berant and Liang, 2014; Bordes et al., 2014a; Yang et al., 2014) in terms of F1-score as defined in (Berant and Liang, 2014), which is the average, over all test questions, of the F1-score of the sets of predicted answers. Since no previous result was published on SimpleQuestions, we only compare different versions of MemNNs. SimpleQuestions questions are labeled with their entire Freebase fact, so we evaluate in terms of path-level accuracy, in which a prediction is correct if the subject and the relationship were correctly retrieved by the system.

The Reverb test set, based on the KB of the same name and introduced in (Fader et al., 2013) is used for evaluation only. It contains 691 questions. We consider the task of re-ranking a small set of candidate answers, which are Reverb facts and are labeled as correct or incorrect. We compare our approach to the original system (Fader et al., 2013), to (Bordes et al., 2014b) and to the original MemNNs (Weston et al., 2015), in terms of accuracy, which is the percentage of questions for which the top-ranked candidate fact is correct.

6.2 Experimental setup

All models were trained with at least the dataset made of synthetic questions created from the KB. The hyperparameters were chosen to maximize the F1-score on WebQuestions validation set, independently of the testing dataset. The embedding dimension and the learning rate were chosen among $\{64, 128, 256\}$ and $\{1, 0.1, ..., 1.0e{-}4\}$ respectively, and the margin γ was set to 0.1. For each configuration of hyperparameters, the F1-score on the validation set was computed regularly during learning to perform early stopping.

We tested additional configurations for our algorithm. First, in the *Candidates as Negatives* setting (negative facts are sampled from the candidate set, see Section 4), abbreviated CANDS AS NEGS, the experimental protocol is the same as in the default setting but the embeddings are initialized with the best configuration of the default setup. Second, our model shares some similarities with an approach studied in (Bordes et al., 2014a), in which the authors noticed important gains using a subgraph representation of answers. For completeness, we also added such a subgraph representation of objects. In that setting, called *Subgraph*, each object o of a fact is itself represented as a bag-of-entities that encodes the immediate neighborhood of o. This *Subgraph* model is trained similarly as our main approach and only the results of a post-hoc ensemble combination of the two models (where the scores are added) are presented. We also report the results obtained by an ensemble of the 5 best models on validation (subgraph excepted); this is denoted *5 models*.

6.3 Results

Comparative results The results of the comparative experiments are given in Table 4. On the main benchmark WebQuestions, our best results use all data sources, the bigger extract from Freebase and the CANDS AS NEGS setting. The two ensembles achieve excellent results, with F1-

						WebQuestions F1-SCORE (%)	SimpleQuestions ACCURACY (%)	Reverb ACCURACY (%)
BASELINES								
Random guess						1.9	4.9	35
(Berant et al., 2013)						31.3	n/a	n/a
(Fader et al., 2014)						n/a	n/a	54
(Bordes et al., 2014b)						29.7	n/a	**73**
(Bordes et al., 2014a) – *using path*						35.3	n/a	n/a
(Bordes et al., 2014a) – *using path + subgraph*						39.2	n/a	n/a
(Berant and Liang, 2014)						39.9	n/a	n/a
(Yang et al., 2014)						41.3	n/a	n/a
(Weston et al., 2015) – *the original MemNN*						n/a	n/a	72
MEMORY NETWORKS *(never trained on* Reverb *– only transfer)*								
KB	TRAIN SOURCES			CANDS AS NEGS	ENSEMBLE			
	WQ	SIQ	PRP					
FB2M	yes	yes	yes	–	–	36.2	62.7	n/a
FB5M	–	–	–	–	–	18.7	44.5	52
FB5M	–	–	yes	–	–	22.0	48.1	62
FB5M	–	yes	–	–	–	22.7	61.6	52
FB5M	–	yes	yes	–	–	28.2	61.2	64
FB5M	yes	–	–	–	–	40.1	46.6	58
FB5M	yes	–	yes	–	–	40.4	47.4	61
FB5M	yes	yes	–	–	–	41.0	61.7	52
FB5M	yes	yes	yes	–	–	41.0	62.1	67
FB5M	yes	yes	yes	yes	–	41.2	62.2	65
FB5M	yes	yes	yes	yes	5 models	41.9	**63.9**	*68*
FB5M	yes	yes	yes	yes	Subgraph	**42.2**	62.9	62

Table 4: **Experimental results** for previous models of the literature and variants of Memory Networks. All results are on the test sets. WQ, SIQ and PRP stand for WebQuestions, SimpleQuestions and Paraphrases respectively. More details in the text.

scores of 41.9% and 42.2% respectively. The best published competing approach (Yang et al., 2014) has an F1-score of 41.3%, which is comparable to a single run of our model (41.2%). On the new SimpleQuestions dataset, the best models achieve $62 - 63\%$ accuracy, while the supporting fact is in the candidate set for about 86% of SimpleQuestions questions. This shows that MemNNs are effective at re-ranking the candidates, but also that simple QA is still not solved.

Our approach bares similarity to (Bordes et al., 2014a) - *using path*. They use FB2M, and so their result (35.3% F1-score on WebQuestions) should be compared to our 36.2%. The models are slightly different in that they replace the entity string with the subject entity in the question representation and that we use the cosine similarity instead of the dot product, which gave consistent improvements. Still, the major differences come from how we use Freebase. First, the removal of the mediator nodes allows us to restrict ourselves to single supporting facts, while they search in paths of length 2 with a heuristic to select the paths to follow (otherwise, inference is too costly), which makes our inference simpler and more efficient. Second, using grouped facts, we integrate multiple answers during learning (through the distant supervision), while they use a grouping heuristic at test time. Grouping facts also allows us to scale much better and to train on FB5M. On WebQuestions, not specifically designed as a simple QA dataset, 86% of the questions can now be answered with a single supporting fact, and performance increases significantly (from 36.2% to 41.0% F1-score). Using the bigger FB5M as KB does not change performance on SimpleQuestions because it was based on FB2M, but the results show that our model is robust to the addition of more entities than necessary.

Transfer learning on Reverb In this set of experiments, all Reverb facts are added to the memory, without any retraining, and we test our ability to rerank answers on the companion QA set. Thus, Table 4 (last column) presents the result of our model *without training* on Reverb *against methods specifically developed on that dataset*. Our best results are 67% accuracy (and 68% for the ensemble of 5 models), which are better than the 54% of the original paper and close to the state-of-the-art 73% of (Bordes et al., 2014b). These results show that the Memory Network approach can

integrate and use new entities and links.

Importance of data sources The bottom half of Table 4 presents the results on the three datasets when our model is trained with different data sources. We first notice that models trained on a single QA dataset perform poorly on the other datasets (e.g. 46.6% accuracy on SimpleQuestions for the model trained on WebQuestions only), which shows that the performance on WebQuestions does not necessarily guarantee high coverage for simple QA. On the other hand, training on both datasets only improves performance; in particular, the model is able to capture all question patterns of the two datasets; there is no "negative interaction".

While paraphrases do not seem to help much on WebQuestions and SimpleQuestions, except when training only with synthetic questions, they have a dramatic impact on the performance on Reverb. This is because WebQuestions and SimpleQuestions questions follow simple patterns and are well formed, while Reverb questions have more syntactic and lexical variability. Thus, paraphrases are important to avoid overfitting on specific question patterns of the training sets.

7 Conclusion

This paper presents an implementation of MemNNs for the task of large-scale simple QA. Our results demonstrate that, if properly trained, MemNNs are able to handle natural language and a very large memory (millions of entries), and hence can reach state-of-the-art on the popular benchmark WebQuestions.

We want to emphasize that many of our findings, especially those regarding how to format the KB, do not only concern MemNNs but potentially any QA system. This paper also introduced the new dataset SimpleQuestions, which, with 100k examples, is one order of magnitude bigger than WebQuestions: we hope that it will foster interesting new research in QA, simple or not.

References

[Berant and Liang2014] Jonathan Berant and Percy Liang. 2014. Semantic parsing via paraphrasing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL'14)*, Baltimore, USA.

[Berant et al.2013] 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 (EMNLP'13)*, Seattle, USA.

[Bordes et al.2014a] Antoine Bordes, Sumit Chopra, and Jason Weston. 2014a. Question answering with subgraph embeddings. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 615–620, Doha, Qatar, October. Association for Computational Linguistics.

[Bordes et al.2014b] Antoine Bordes, Jason Weston, and Nicolas Usunier. 2014b. Open question answering with weakly supervised embedding models. In *Proceedings of the 7th European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD'14)*, Nancy, France. Springer.

[Cai and Yates2013] Qingqing Cai and Alexander Yates. 2013. Large-scale semantic parsing via schema matching and lexicon extension. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Sofia, Bulgaria, August.

[Duchi et al.2011] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 12.

[Dumais et al.2002] Susan Dumais, Michele Banko, Eric Brill, Jimmy Lin, and Andrew Ng. 2002. Web question answering: Is more always better? In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 291–298. ACM.

[Fader et al.2011] Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In *Proceedings of the Conference of Empirical Methods in Natural Language Processing (EMNLP'11)*, Edinburgh, UK, July 27-31.

[Fader et al.2013] 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 (ACL'13)*, Sofia, Bulgaria.

[Fader et al.2014] Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. 2014. Open question answering over curated and extracted knowledge bases. In *Proceedings of 20th SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'14)*, New York City, USA. ACM.

[Kwiatkowski et al.2013] 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 (EMNLP'13), Seattle, USA, October.

[Lin et al.2012] Thomas Lin, Mausam, and Oren Etzioni. 2012. Entity linking at web scale. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction (AKBC-WEKEX'12)*, Montreal, Canada.

[Recht et al.2011] 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)*., Vancouver, Canada.

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

[Sukhbaatar et al.2015] Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. 2015. Weakly supervised memory networks. *arXiv preprint arXiv:1503.08895*.

[Unger et al.2012] 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 (WWW'12)*, Lyon, France. ACM.

[Voorhees and Tice2000] Ellen M Voorhees and DM Tice. 2000. Overview of the trec-9 question answering track. In *TREC*.

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

[Weston et al.2015] Jason Weston, Sumit Chopra, and Antoine Bordes. 2015. Memory networks. In *Proceedings of the 2014 International Conference on Learning Representations (ICLR)*. arXiv preprint arXiv:1410.3916.

[Yahya et al.2012] 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 the Conference on Empirical Methods in Natural Language Processing*.

[Yang et al.2014] Min-Chul Yang, Nan Duan, Ming Zhou, and Hae-Chang Rim. 2014. Joint relational embeddings for knowledge-based question answering. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 645–650.

[Yao and Van Durme2014] Xuchen Yao and Benjamin Van Durme. 2014. Information extraction over structured data: Question answering with freebase. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL'14)*, Baltimore, USA.