Dataset and Neural Recurrent Sequence Labeling Model for Open-Domain Factoid Question Answering

Peng Li, Wei Li, Zhengyan He, Xuguang Wang, Ying Cao, Jie Zhou, Wei Xu
Baidu Research - Institute of Deep Learning
{lipeng17,liwei26,hezhengyan,wangxuguang,caoying03,
zhoujie01,wei.xu}@baidu.com

Abstract

While question answering (QA) with neural network, i.e. neural QA, has achieved promising results in recent years, lacking of large scale real-word QA dataset is still a challenge for developing and evaluating neural QA system. To alleviate this problem, we propose a large scale human annotated real-world QA dataset WebQA with more than 42k questions and 556k evidences. As existing neural QA methods resolve QA either as sequence generation or classification/ranking problem, they face challenges of expensive softmax computation, unseen answers handling or separate candidate answer generation component. In this work, we cast neural QA as a sequence labeling problem and propose an end-to-end sequence labeling model, which overcomes all the above challenges. Experimental results on WebQA show that our model outperforms the baselines significantly with an F1 score of 74.69% with word-based input, and the performance drops only 3.72 F1 points with more challenging character-based input.

1 Introduction

Question answering (QA) with neural network, i.e. neural QA, is an active research direction along the road towards the long-term AI goal of building general dialogue agents (Weston et al., 2016). Unlike conventional methods, neural QA does not rely on feature engineering and is (at least nearly) end-to-end trainable. It reduces the requirement for domain specific knowledge significantly and makes domain adaption easier. Therefore, it has attracted intensive attention in recent years.

Resolving QA problem requires several fundamental abilities including reasoning, memorization, etc. Various neural methods have been proposed to improve such abilities, including neural tensor networks (Socher et al., 2013), recursive networks (Iyyer et al., 2014), convolution neural networks (Yih et al., 2014; Dong et al., 2015; Yin et al., 2015), attention models (Hermann et al., 2015; Yin et al., 2015; Santos et al., 2016), and memories (Graves et al., 2014; Weston et al., 2015; Kumar et al., 2016; Bordes et al., 2015; Sukhbaatar et al., 2015), etc. These methods achieve promising results on various datasets, which demonstrates the high potential of neural QA. However, we believe there are still two major challenges for neural QA:

System development and/or evaluation on real-world data: Although several high quality and well-designed QA datasets have been proposed in recent years, there are still problems about using them to develop and/or evaluate QA system under real-world settings due to data size and the way they are created. For example, bAbI (Weston et al., 2016) and the 30M Factoid Question-Answer Corpus (Serban et al., 2016) are artificially synthesized; the TREC datasets (Harman and Voorhees, 2006), Free917 (Cai and Yates, 2013) and WebQuestions (Berant et al., 2013) are human generated but only have few thousands of questions; SimpleQuestions (Bordes et al., 2015) and the CNN and Daily Mail news datasets (Hermann et al., 2015) are large but generated under controlled conditions. Thus, a new large-scale real-world QA dataset is needed.

A new design choice for answer production besides sequence generation and classifica-

tion/ranking: Without loss of generality, the methods used for producing answers in existing neural QA works can be roughly categorized into the sequence generation type and the classification/ranking type. The former generates answers word by word, e.g. (Weston et al., 2016; Kumar et al., 2016; Hermann et al., 2015). As it generally involves $\mathrm{softmax}$ computation over a large vocabulary, the computational cost is remarkably high and it is hard to produce answers with out-of-vocabulary word. The latter produces answers by classification over a predefined set of answers, e.g. (Sukhbaatar et al., 2015), or ranking given candidates by model score, e.g. (Yin et al., 2015). Although it generally has lower computational cost than the former, it either also has difficulties in handling unseen answers or requires an extra candidate generating component which is hard for end-to-end training. Above all, we need a new design choice for answer production that is both computationally effective and capable of handling unseen words/answers.

In this work, we address the above two challenges by a new dataset and a new neural QA model. Our contributions are two-fold:

- We propose a new **large-scale real-world** factoid QA dataset WebQA with more than 42k questions and 566k evidences, where an evidence is a piece of text that contains relevant information to answer the question. On one hand, our dataset is an order of magnitude larger than existing real-world QA datasets (Harman and Voorhees, 2006; Cai and Yates, 2013; Berant et al., 2013), which are generally insufficient to train an end-to-end QA system. On the other hand, all questions in our dataset are asked by *real-world users in daily life*, which is significantly more close to real-world settings than those generated under controlled conditions (Bordes et al., 2015; Hermann et al., 2015). Besides, as we also provide multiple human annotated evidences for each question, the dataset can be used in research such as evidence ranking and answer sentence selection as well.

- We introduce an end-to-end sequence labeling technique into neural QA as a new design choice for answer production. Mimicking how humans find answers using search engine, we use conditional random field (CRF) (Lafferty et al., 2001) to label the answer of a question from retrieved evidence. We avoid feature engineering by computing features with a neural model jointly trained with CRF. As our model does not rely on predefined vocabulary or candidates, it can handle unseen words/answers easily and get rid of expensive $\mathrm{softmax}$ computation.

Q:	Who is the first wife of Albert Einstein ?
E:	Einstein/O married/O his/O first/O wife/O Mileva/B Marić/I in/O 1903/O
A:	Mileva Marić

Figure 1: Factoid QA as sequence labeling.

Experimental results show that our model outperforms baselines with a large margin on both WebQA and the CNN and Daily Mail news datasets (Hermann et al., 2015), indicating that it is effective and generalizable to multiple languages. Furthermore, our model even achieves an F1 score of 70.97% on character-based input, which is comparable with the 74.69% F1 score on word-based input, demonstrating that our model is robust.

2 Factoid QA as Sequence Labeling

In this work, we focus on open-domain factoid QA. Taking Figure 1 as an example, we formalize the problem as follows: given each question Q, we have one or more evidences E, and the task is to produce the answer A, where an evidence is a piece of text of any length that contains relevant information to answer the question. The advantage of this formalization is that evidences can be retrieved from web or unstructured knowledge base, which can improve system coverage significantly.

Inspired by (Yao et al., 2013), we introduce end-to-end sequence labeling as a new design choice for answer production in neural QA. Given a question and an evidence, we use CRF (Lafferty et al., 2001) to assign a label to each word in the evidence to indicate whether the word is at the beginning (B), inside (I) or outside (O) of the answer (see Figure 1 for example). The key difference between our work and (Yao et al., 2013) is that (Yao et al., 2013) needs a lot work on feature engineering which further relies

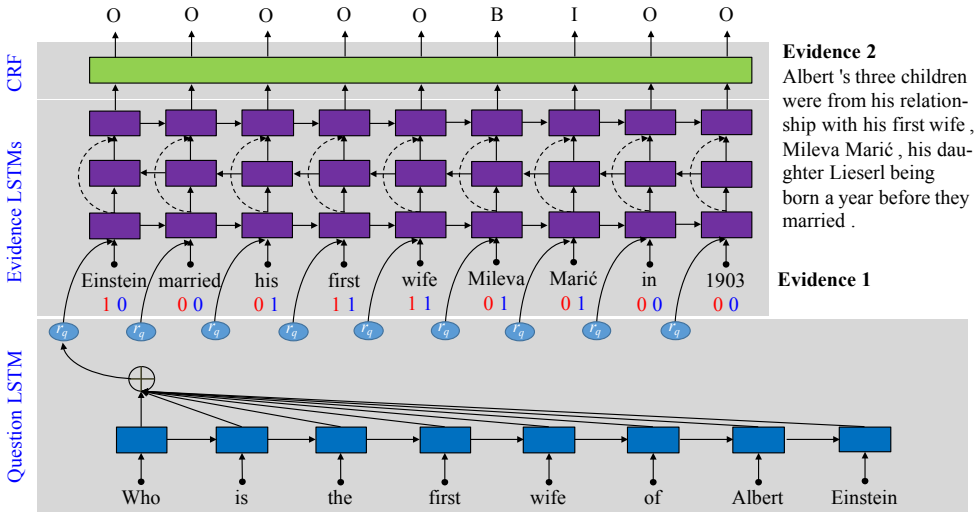

Figure 2: Neural recurrent sequence labeling model for factoid QA. The model consists of three components: "Question LSTM" for computing question representation (r_q), "Evidence LSTMs" for analyzing evidence, and "CRF" for producing label sequence which indicates whether each word in the evidence is at the beginning (B), inside (I) or outside (O) of the answer. Each word in the evidence is also equipped with two 0-1 features (see Section 3.4). We plot r_q multiple times for clarity.

on POS/NER tagging, dependency parsing, question type analysis, etc. While we avoid feature engineering, and only use one *single* model to solve the problem. Furthermore, compared with sequence generation and classification/ranking methods for answer production, our method avoids expensive $\mathrm{softmax}$ computation and can handle unseen answers/words naturally in a principled way.

Formally, we formalize QA as a sequence labeling problem as follows: suppose we have a vocabulary V of size $|V|$, given question $\mathbf{x^q} = (x_1^q, x_2^q, \cdots, x_N^q)$ and evidence $\mathbf{x^e} = (x_1^e, x_2^e, \cdots, x_M^e)$, where x_i^q and x_j^e are one-hot vectors of dimension $|V|$, and N and M are the number of words in the question and evidence respectively. The problem is to find the label sequence $\hat{\mathbf{y}}$ which maximizes the conditional probability under parameter θ

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} p_\theta(\mathbf{y}|\mathbf{x^q}, \mathbf{x^e}). \quad (1)$$

In this work, we model $p_\theta(\mathbf{y}|\mathbf{x^q}, \mathbf{x^e})$ by a neural network composed of LSTMs and CRF.

3 Recurrent Sequence Labeling Model

3.1 Overview

Figure 2 shows the structure of our model. The model consists of three components: (1) question LSTM for computing question representation; (2) evidence LSTMs for evidence analysis; and (3) a CRF layer for sequence labeling. The question LSTM in a form of a single layer LSTM equipped with a single time attention takes the question as input and generates the question representation r_q. The three-layer evidence LSTMs takes the evidence, question representation r_q and optional features as input and produces "features" for the CRF layer. The CRF layer takes the "features" as input and produces the label sequence. The details will be given in the following sections.

3.2 Long Short-Term Memory (LSTM)

Following (Graves, 2013), we define $(s', y') = LSTM(x, s, y)$ as a function mapping its input x, previous state s and output y to current state s' and output y':

$$\begin{aligned} i &= \sigma(W_{xi}x + W_{yi}y + W_{si}s + b_i) && (2)\\ f &= \sigma(W_{xf}x + W_{yf}y + W_{sf}s + b_f) && (3)\\ s' &= fs + i\sigma(W_{xs}x + W_{ys}y + b_s) && (4)\\ o &= \sigma(W_{xo}x + W_{yo}y + W_{so}s' + b_o) && (5)\\ y' &= o\tanh(s') && (6) \end{aligned}$$

where $W_* \in \mathbb{R}^{H\times H}$ are parameter matrices, $b_* \in \mathbb{R}^H$ are biases, H is LSTM layer width, σ is the

sigmoid function, i, f and o are the input gate, forget gate and output gate respectively.

3.3 Question LSTM

The question LSTM consists of a single-layer LSTM [1] and a single-time attention model. The question $\mathbf{x^q} = (x_1^q, x_2^q, \cdots, x_N^q)$ is fed into the LSTM to produce a sequence of vector representations $q_1, q_2, \cdots, q_N$

$$(s_i^q, q_i) = LSTM(\overline{E}x_i^q, s_{i-1}^q, q_{i-1}) \quad (7)$$

where $\overline{E} \in \mathbb{R}^{D \times |V|}$ is the embedding matrix and D is word embedding dimension. Then a weight α_i is computed by the single-time attention model for each q_i

$$\alpha_i = \text{softmax}\left(v_q^{\mathrm{T}} \tanh(W_a q_i)\right) \quad (8)$$

where $v_q \in \mathbb{R}^D$ and $W_a \in \mathbb{R}^{D \times D}$. And finally the weighted average r_q of q_i is used as the representation of the question

$$r_q = \sum_i \alpha_i q_i. \quad (9)$$

3.4 Evidence LSTMs

The three-layer evidence LSTMs processes evidence $\mathbf{x^e} = (x_1^e, x_2^e, \cdots, x_M^e)$ to produce "features" for the CRF layer.

The first LSTM layer takes evidence $\mathbf{x^e}$, question representation r_q and optional features as input. We find the following two simple common word indicator features are effective:

- **Question-Evidence common word feature (q-e.comm)**: for each word in the evidence, the feature has value 1 when the word also occurs in the question, otherwise 0. The intuition is that words occurring in questions tend not to be part of the answers for factoid questions.
- **Evidence-Evidence common word feature (e-e.comm)**: for each word in the evidence, the feature has value 1 when the word occurs in another evidence, otherwise 0. The intuition is that words shared by two or more evidences are more likely to be part of the answers.

[1] Multi-layer LSTMs can be used but no gain was observed.

Although counterintuitive, we found non-binary e-e.comm feature values does not work well. Because the more evidences we considered, the more words tend to get non-zero feature values, and the less discriminative the feature is.

The second LSTM layer stacks on top of the first LSTM layer, but processes its output in a reverse order. The third LSTM layer stacks upon the first and second LSTM layers with cross layer links, and its output serves as features for CRF layer.

Formally, the computations are defined as follows

$$\begin{aligned} x_{e1} &= [\overline{E}x_j^e; r_q; \overline{F}_1 g_j^1; \overline{F}_2 g_j^2] && (10)\\ (s_j^1, e_j^1) &= LSTM(x_{e1}, s_{j-1}^1, e_{j-1}^1) && (11)\\ (s_j^2, e_j^2) &= LSTM(e_j^1, s_{j+1}^2, e_{j+1}^2) && (12)\\ (s_j^3, e_j^3) &= LSTM([e_j^1; e_j^2], s_{j-1}^3, e_{j-1}^3) && (13) \end{aligned}$$

where g_j^1 and g_j^2 are one-hot feature vectors, $\overline{F}_1 \in \mathbb{R}^{D_1 \times 2}$ and $\overline{F}_2 \in \mathbb{R}^{D_2 \times 2}$ are embeddings for the features, and D_1 and D_2 are the feature embedding dimensions. Note that we use the same word embedding matrix $\overline{E}$ as in question LSTM.

3.5 Sequence Labeling

Following (Huang et al., 2015; Zhou and Xu, 2015), we use CRF on top of evidence LSTMs for sequence labeling. The probability of a label sequence $\mathbf{y}$ given question $\mathbf{x^q}$ and evidence $\mathbf{x^e}$ is computed as

$$p_\theta(\mathbf{y}|\mathbf{x^q}, \mathbf{x^e}) \propto \exp(\sum_j \mu[y_{j-1}, y_j] + \sum_j e_j[y_j]) \quad (14)$$

where $e_j = W_e e_j^3$, $W_e \in \mathbb{R}^{L \times D}$, L is the number of label types, $\mu[i, j]$ is the transition weight from label i to j, and $e_j[i]$ is the i-th value of vector e_j.

4 Training

The objective function of our model is

$$L_\theta(T) = -\sum_i \log\left(p_\theta(\tilde{\mathbf{y}}_i|\mathbf{x}_i^\mathbf{q}, \mathbf{x}_i^\mathbf{e})\right) + \frac{1}{2}\lambda||\theta||^2$$

where $\tilde{\mathbf{y}}_i$ is the golden label sequence, and $T = \{(\tilde{\mathbf{y}}_i, \mathbf{x}_i^\mathbf{q}, \mathbf{x}_i^\mathbf{e}\}$ is training set.

We use a minibatch stochastic gradient descent (SGD) (Lecun et al., 1998) algorithm with rmsprop (Tieleman and Hinton, 2012) to minimize the objective function. The initial learning rate is 0.001,

Dataset	Question		Annotated Evidence				Retrieved Evidence	
			Positive		Negative			
	#	Word #	#	Word #	#	Word #	#	Word #
Train	36,145	374,500	140,897	10,757,652	122,206	14,808,758	171,838	7,233,543
Validation	3,018	36,666	5,412	233,911	/	/	60,351	3,633,540
Test	3,024	36,815	5,445	234,258	/	/	60,465	3,620,391

Table 1: Statistics of WebQA dataset.

batch size is 120, and $\lambda = 0.016$. We also apply dropout (Hinton et al., 2012) to the output of all the LSTM layers. The dropout rate is 0.05. All these hyper-parameters are determined empirically via grid search on validation set.

5 WebQA Dataset

In order to train and evaluate open-domain factoid QA system for real-world questions, we build a new Chinese QA dataset named as WebQA. The dataset consists of tuples of (question, evidences, answer), which is similar to example in Figure 1. All the questions, evidences and answers are collected from web. Table 1 shows some statistics of the dataset.

The questions and answers are mainly collected from a large community QA website Baidu Zhidao [2] and a small portion are from hand collected web documents. Therefore, all these questions are indeed **asked by real-world users in daily life instead of under controlled conditions**. All the questions are of single-entity factoid type, which means (1) each question is a factoid question and (2) its answer involves only one entity (but may have multiple words). The question in Figure 1 is a positive example, while the question "Who are the children of Albert Enistein?" is a counter example because the answer involves three persons. The type and correctness of all the question answer pairs are verified by at least two annotators.

All the evidences are retrieved from Internet by using a search engine with questions as queries. We download web pages returned in the first 3 result pages and take all the text pieces which have no more than 5 sentences and include at least one question word as candidate evidences. As evidence retrieval is beyond the scope of this work, we simply use TF-IDF values to re-rank these candidates.

For each question in the training set, we provide the top 10 ranked evidences to annotate ("Annotated Evidence" in Table 1). An evidence is annotated as positive if the question can be answered by just reading the evidence without any other prior knowledge, otherwise negative. Only evidences whose annotations are agreed by at least two annotators are retained. We also provide trivial negative evidences ("Retrieved Evidence" in Table 1), i.e. evidences that do not contain golden standard answers.

For each question in the validation and test sets, we provide one major positive evidence, and maybe an additional positive one to compute features. Both of them are annotated. Raw retrieved evidences are also provided for evaluation purpose ("Retrieved Evidence" in Table 1).

The dataset will be released on the project page `http://idl.baidu.com/WebQA.html`.

6 Evaluation on WebQA Dataset

6.1 Baselines

We compare our model with two sets of baselines:

MemN2N (Sukhbaatar et al., 2015) is an end-to-end trainable version of memory networks (Weston et al., 2015). It encodes question and evidence with a bag-of-word method and stores the representations of evidences in an external memory. A recurrent attention model is used to retrieve relevant information from the memory to answer the question.

Attentive and Impatient Readers (Hermann et al., 2015) use bidirectional LSTMs to encode question and evidence, and do classification over a large vocabulary based on these two encodings. The simpler Attentive Reader uses a similar way as our work to compute attention *for the evidence*. And the more complex Impatient Reader computes attention after processing each question word.

The key difference between our model and the two readers is that they produce answer by doing classification over a large vocabulary, which is com-

[2] `http://zhidao.baidu.com`

putationally expensive and has difficulties in handling unseen words. However, as our model uses an end-to-end trainable sequence labeling technique, it avoids both of the two problems by its nature.

6.2 Evaluation Method

The performance is measured with precision (P), recall (R) and F1-measure (F1) [3]

$$P = \frac{|C|}{|A|}, R = \frac{|C|}{|Q|}, F1 = \frac{2PR}{P+R} \quad (15)$$

where C is the list of correctly answered questions, A is the list of produced answers, and Q is the list of all questions [4].

As WebQA is collected from web, the same answer may be expressed in different surface forms in the golden standard answer and the evidence, e.g. "北京 (Beijing)" v.s. "北京市 (Beijing province)". Therefore, we use two ways to count correctly answered questions, which are referred to as "strict" and "fuzzy" in the tables:

- **Strict matching**: A question is counted if and only if the produced answer is identical to the golden standard answer;
- **Fuzzy matching**: A question is counted if and only if the produced answer is a synonym [5] of the golden standard answer;

And we also consider two evaluation settings:

- **Annotated evidence**: Each question has one major annotated evidence and maybe another annotated evidence for computing q-e.comm and e-e.comm features (Section 3.4);
- **Retrieved evidence**: Each question is provided with at most 20 automatically retrieved evidences (see Section 5 for details). All the evidences will be processed by our model independently and answers are voted by frequency to decide the final result. Note that a large amount of the evidences are negative and our model should not produce any answer for them.

[3] Measures such MAP and MRR are often also used for evaluating QA system. However, as our model gives only conditional probabilities which are not directly comparable for different answers, we will not include these measures in this work.

[4] As the baselines will produce exactly one answer for each question, P, R and F1 will be identical for them.

[5] The synonyms will also be released.

6.3 Model Settings

If not specified, the following hyper-parameters will be used in the reset of this section: LSTM layer width $H = 64$ (Section 3.2), word embedding dimension $D = 64$ (Section 3.3), feature embedding dimension $D_1 = D_2 = 2$ (Section 3.3). The word embeddings are initialized with pre-trained embeddings using a 5-gram neural language model (Bengio et al., 2003) and is fixed during training.

We will show that injecting noise data is important for improving performance on retrieved evidence setting in Section 6.5. In the following experiments, 20% of the training evidences will be negative ones randomly selected on the fly, of which 25% are annotated negative evidences and 75% are retrieved trivial negative evidences (Section 5). The percentages are determined empirically. Intuitively, we provide the noise data to teach the model learning to recognize unreliable evidence.

For each evidence, we will randomly sample another evidence from the rest evidences of the question and compare them to compute the e-e.comm feature (Section 3.4). We will develop more powerful models to process multiple evidences in a more principle way in the future.

As the answer for each question in our WebQA dataset only involves one entity (Section 5), we distinguish label Os before and after the first B in the label sequence explicitly to discourage our model to produce multiple answers for a question. For example, the golden labels for the example evidence in Figure 1 will became "Einstein/O1 married/O1 his/O1 first/O1 wife/O1 Mileva/B Marić/I in/O2 1903/O2", where we use "O1" and "O2" to denote label Os before and after the first B [6]. "Fuzzy matching" is also used for computing golden standard labels for training set.

For each setting, we will run three trials with different random seeds and report the average performance in the following sections.

6.4 Comparison with Baselines

As the baselines can only predict one-word answers, we only do experiments on the one-word answer subset of WebQA, i.e. only questions with one-word answers are retained for training, validation and test.

[6] All the words in a negative evidence will get label "O1".

System	Validation (Strict)			Test (Strict)		
	P	R	F1	P	R	F1
MemN2N	52.61	52.61	52.61	50.14	50.14	50.14
Attentive Reader	**65.41**	65.41	65.41	62.46	62.46	62.46
Impatient Reader	63.05	63.05	63.05	59.83	59.83	59.83
Ours	64.46	**87.62**	**74.28**	**63.30**	**87.70**	**73.53**

Table 2: Comparison with baselines on the one-word answer subset of WebQA.

Model	Noise	Annotated Evidence									Retrieved Evidence		
		Strict (Val.)			Strict (Test)			Fuzzy (Test)			Fuzzy (Test, Voting)		
		P	R	F1	P	R	F1	P	R	F1	P	R	F1
Softmax	False	58.21	72.90	64.73	57.42	72.32	64.02	61.53	77.49	68.59	67.53	74.18	70.70
Softmax (k-1)	False	58.28	71.80	64.34	58.22	71.83	64.31	62.48	77.08	69.02	67.06	73.47	70.11
CRF	False	63.19	**79.21**	**70.30**	61.90	**77.33**	68.76	66.00	**82.45**	73.32	68.97	74.64	71.69
Softmax	True	59.74	69.11	64.08	59.38	68.77	63.73	63.58	73.63	68.24	69.75	74.72	72.15
Softmax (k-1)	True	59.84	67.51	63.44	59.76	67.61	63.44	64.02	72.44	67.97	69.11	73.93	71.44
CRF	True	**64.42**	75.84	69.67	**63.72**	76.09	**69.36**	**67.53**	80.63	**73.50**	**72.66**	**76.83**	**74.69**

Table 3: Evaluation results on the entire WebQA dataset.

As shown in Table 2, our model achieves significant higher F1 scores than all the baselines.

The main reason for the relative low performance of MemN2N is that it uses a bag-of-word method to encode question and evidence such that higher order information like word order is absent to the model. We think its performance can be improved by designing more complex encoding methods (Hill et al., 2016) and leave it as a future work.

The Attentive and Impatient Readers only have access to the fixed length representations when doing classification. However, our model has access to the outputs of all the time steps of the evidence LSTMs, and scores the label sequence as a whole. Therefore, our model achieves better performance.

6.5 Evaluation on the Entire WebQA Dataset

In this section, we evaluate our model on the entire WebQA dataset. The evaluation results are shown in Table 3. Although producing multi-word answers is harder, our model achieves comparable results with the one-word answer subset (Table 2), demonstrating that our model is effective for both single-word and multi-word word settings.

"Softmax" in Table 3 means we replace CRF with $\mathrm{softmax}$, i.e. replace Eq. (14) with

$$p_\theta(\mathbf{y}|\mathbf{x^q}, \mathbf{x^e}) = \prod_k \mathrm{softmax}(e_k[y_k]) \qquad (16)$$

CRF outperforms $\mathrm{softmax}$ significantly in all cases. The reason is that $\mathrm{softmax}$ predicts each label independently, suggesting that modeling label transition explicitly is essential for improving performance. A natural choice for modeling label transition in $\mathrm{softmax}$ is to take the last prediction into account as in (Bahdanau et al., 2015). The result is shown in Table 3 as "Softmax(k-1)". However, its performance is only comparable with "Softmax" and significantly lower than CRF. The reason is that we can enumerate all possible label sequences implicitly by dynamic programming for CRF during predicting but this is not possible for "Softmax(k-1)" [7], which indicates CRF is a better choice.

"Noise" in Table 3 means whether we inject noise data or not (Section 6.3). As all evidences are positive under the annotated evidence setting, the ability for recognizing unreliable evidence will be useless. Therefore, the performance of our model with and without noise is comparable under the annotated evidence setting. However, the ability is important to improve the performance under the retrieved evidence setting because a large amount of the retrieved evidences are negative ones. As a result, we observe significant improvement by injecting noise data for this setting.

6.6 Effect of Word Embedding

As stated in Section 6.3, the word embedding $\overline{E}$ is initialized with LM embedding and kept fixed in training. We evaluate different initialization and op-

[7] We think the performance of "Softmax(k-1)" can be improved by beam search and leave it as a future work.

Settings		Annotated Evidence			Retrieved Evidence (Voting)		
Initialization	**Joint Training**	**P**	**R**	**F1**	**P**	**R**	**F1**
LM embedding	False	**67.53**	**80.63**	**73.50**	**72.66**	**76.83**	**74.69**
LM embedding	True	65.16	76.76	70.48	70.15	74.09	72.06
Random	True	63.37	71.52	67.19	66.74	70.11	68.38

Table 4: Effect of embedding initialization and training. Only fuzzy matching results are shown.

Annotated Evidence (Fuzzy)			
Settings	**P**	**R**	**F1**
Both	67.53	**80.63**	**73.50**
w/o q-e.comm	64.32	69.26	66.69
w/o e-e.comm	**70.07**	70.30	70.18
Retrieved Evidence (Voting, Fuzzy)			
Settings	**P**	**R**	**F1**
Both	**72.66**	**76.83**	**74.69**
w/o q-e.comm	63.97	67.77	65.81
w/o e-e.comm	71.05	75.69	73.30

Table 5: Effect of q-e.comm and e-e.comm features.

Annotated Evidence (Fuzzy)			
Settings	**P**	**R**	**F1**
attention	67.53	**80.63**	**73.50**
max	**67.80**	78.84	72.90
average	67.30	78.03	72.27
Retrieved Evidence (Voting, Fuzzy)			
Settings	**P**	**R**	**F1**
attention	**72.66**	**76.83**	**74.69**
max	72.08	76.34	74.15
average	71.31	75.41	73.30

Table 6: Effect of question representations.

timization methods in this section. The evaluation results are shown in Table 4. The second row shows the results when the embedding is optimized jointly during training. The performance drops significantly. Detailed analysis reveals that the trainable embedding enlarge trainable parameter number and the model gets over fitting easily. The model acts like a context independent entity tagger to some extend, which is not desired. For example, the model will try to find any location name in the evidence when the word "在哪 (where)" occurs in the question. In contrary, pre-trained fixed embedding forces the model to pay more attention to the latent syntactic regularities. And it also carries basic priors such as "梨 (pear)" is fruit and "李世石 (Lee Sedol)" is a person, thus the model will generalize better to test data with fixed embedding. The third row shows the result when the embedding is randomly initialized and jointly optimized. The performance drops significantly further, suggesting that pre-trained embedding indeed carries meaningful priors.

6.7 Effect of q-e.comm and e-e.comm Features

As shown in Table 5, both the q-e.comm and e-e.comm features are effective, and the q-e.comm feature contributes more to the overall performance. The reason is that the interaction between question and evidence is limited and q-e.comm feature with value 1, i.e. the corresponding word also occurs in the question, is a strong indication that the word may not be part of the answer.

6.8 Effect of Question Representations

In this section, we compare the single-time attention method for computing r_q ($\mathrm{attention}$, Eq. (8, 9)) with two widely used options: element-wise max operation max: $r_q = \max_i q_i$ and element-wise average operation $\mathrm{average}$: $r_q = \frac{1}{N}\sum_i q_i$. Intuitively, $\mathrm{attention}$ can distill information in a more flexible way from $\{q_i\}$, while $\mathrm{average}$ tends to hide the differences between them, and max lies between $\mathrm{attention}$ and $\mathrm{average}$. The results in Table 6 suggest that the more flexible and selective the operation is, the better the performance is.

6.9 Effect of Evidence LSTMs Structures

We investigate the effect of evidence LSTMs layer number, layer width and cross layer links in this section. The results are shown in Figure 3. For fair comparison, we do not use cross layer links in Figure 3 (a) (dotted lines in Figure 2), and highlight the results with cross layer links (layer width 64) with circle and square for retrieved and annotated evidence settings respectively. We can conclude that: (1) generally the deeper and wider the model is, the better the performance is; (2) cross layer links are effective as they make the third evidence LSTM layer see information in both directions.

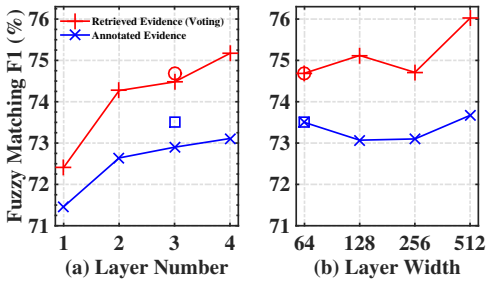

Figure 3: Effect of evidence LSTMs structures. For fair comparison, cross layer links are not used in (a).

Annotated Evidence (Fuzzy)			
Model	**P (%)**	**R (%)**	**F1 (%)**
Word-based	**67.53**	**80.63**	**73.50**
Char-based	67.00	72.80	69.78
Retrieved Evidence (Voting, Fuzzy)			
Model	**P (%)**	**R (%)**	**F1 (%)**
Word-based	**72.66**	**76.83**	**74.69**
Char-based	68.88	73.20	70.97

Table 7: Word-based v.s. character-based input.

6.10 Word-based v.s. Character-based Input

Our model achieves fuzzy matching F1 scores of 69.78% and 70.97% on character-based input in annotated and retrieved evidence settings respectively (Table 7), which are only 3.72 and 3.72 points lower than the corresponding scores on word-based input respectively. The performance is promising, demonstrating that our model is robust and effective.

7 Evaluation on CNN and Daily Mail News Datasets

7.1 Dataset

We also evaluate our model on the CNN and Daily Mail News datasets (Hermann et al., 2015). The two datasets are in English and created from CNN and Daily Mail news articles. And all the questions are of cloze style. The following is a example data point from (Hermann et al., 2015):

Evidence:
the *ent381* producer allegedly struck by *ent212* will not press charges against the " *ent153* " host , his lawyer said friday . *ent212* , who hosted one of the most - watched television shows in the world , was dropped by the *ent381* wednesday after an internal investigation by the *ent180* broadcaster found he had subjected producer *ent193* " to an unprovoked physical and verbal attack . " . . .

Question:
producer **X** will not press charges against *ent212* , his lawyer says .

Answer
ent193

Each data point consists of one evidence, one question and one answer. There is one missing entity for each question (denoted by **X** in the example), and the task is to predict the missing entity given the evidence and question. All the entities in the datasets are anonymized and represented by special markers, e.g. *ent381*, to make the task harder.

Although these two datasets are large and useful for research, they are still not suitable for developing and evaluating QA system on real-world data, because the questions are of cloze style and created from summaries by rules instead of being asked by real-world user, which is significantly different from our WebQA dataset.

7.2 Model Settings

The following settings are used in this section: LSTM layer width $H \in \{64, 128\}$ (Section 3.2), word embedding dimension $D = 64$ (Section 3.3), feature embedding $D_1 = 2$ (Section 3.4). As there is only one evidence for each question, we only use q-e.comm feature in this section and do not provide noise data in training. Following the protocol in (Hermann et al., 2015), we shuffle the entity markers in training for fair comparison.

As there is exact one answer for each question, we also use the {B, I, O1, O2} label set in this section and only assign B and I labels to the first occurrence of the golden standard answer in the evidence.

We evaluate two embedding settings in our experiments:

1. Fixed LM embedding: the word embeddings are initialized with pre-trained embeddings using a 5-gram neural language model (Bengio et al., 2003) and is fixed during training. However, as the entity markers are not in the vocabulary, we jointly train their embeddings in training.

System	CNN		Daily Mail	
	Val.	**Test**	**Val.**	**Test**
Single Model				
Attentive Reader (Hermann et al., 2015)	61.6	63.0	70.5	69.0
Impatient Reader (Hermann et al., 2015)	61.8	63.8	69.0	68.0
MemN2Ns (Hill et al., 2016)	63.4	66.8	/	/
L2R Reader (Tian and LI, 2016)	64.3	65.8	69.1	67.3
AS Reader (Kadlec et al., 2016)	68.6	69.5	75.0	73.9
GA Reader (Dhingra et al., 2016)	73.0	73.8	76.7	75.7
Stanford AR (Chen et al., 2016)	72.4	72.4	76.9	75.8
Iterative attention (Sordoni et al., 2016)	72.6	73.3	/	/
AoA Reader (Cui et al., 2016)	73.1	74.4	/	/
QANN (Weissenborn, 2016)	/	73.7	/	77.2
Ensemble Model				
MemN2Ns (Hill et al., 2016)	66.2	69.4	/	/
AS Reader (Kadlec et al., 2016)	73.9	75.4	78.7	77.7
GA Reader (Dhingra et al., 2016)	76.4	77.4	79.1	78.1
Iterative attention (Sordoni et al., 2016)	75.2	76.1	/	/
Ours (Single Model)				
Ours + fixed LM embedding + H=64	75.8	76.0	77.4	76.8
Ours + jointly trained embedding + H=64	75.7	75.4	77.6	77.0
Ours + jointly trained embedding + H=128	**77.7**	**77.1**	**78.9**	**78.0**

Table 8: Evaluation results on the CNN and Daily Mail news datasets.

2. Jointly trained embedding: all the embeddings are randomly initialized and jointly trained.

For each setting, we will run three trials with different random seeds and report the average performance.

7.3 Experimental Results

The results are shown in Table 8. We divide the baselines into two groups by whether or not using model ensemble. Note that we do not use model ensemble in this work. The results show that our model achieves the best single model accuracy on both of the two datasets. It also outperforms all the ensemble models except GA Reader (Dhingra et al., 2016). However, the performance gap is tiny. The above results demonstrate that our model generalizes not only to non-Chinese datasets but also to cloze style questions.

The performance gaps between using fixed LM embedding and jointly trained embedding on the two datasets are significant smaller than that on our WebQA dataset. We think the reason is that all the answers in these two datasets are entity markers and their embeddings are both jointly trained in these two settings.

8 Conclusion and Future Work

In this work, we build a new human annotated real-world QA dataset WebQA for developing and evaluating QA system on real-world QA data. We also propose a new end-to-end recurrent sequence labeling model for QA. Experimental results show that our model outperforms baselines significantly.

There are several future directions we plan to pursue. First, multi-entity factoid and non-factoid QA are also interesting topics. Second, we plan to extend our model to multi-evidence cases. Finally, inspired by Residual Network (He et al., 2016), we will investigate deeper and wider models in the future.

References

[Bahdanau et al.2015] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of ICLR 2015*.

[Bengio et al.2003] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *The Journal of Machine Learning Research*, 3.

[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*, pages 1533–1544, October.

[Bordes et al.2015] Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale simple question answering with memory networks. *arXiv:1506.02075*.

[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)*, pages 423–433, August.

[Chen et al.2016] Danqi Chen, Jason Bolton, and Christopher D. Manning. 2016. A thorough examination of the cnn/daily mail reading comprehension task. *To appear in Proceedings of ACL 2016*.

[Cui et al.2016] Yiming Cui, Zhipeng Chen, Si Wei, Shijin Wang, Ting Liu, and Guoping Hu. 2016. Attention-over-attention neural networks for reading comprehension. *arXiv:1607.04423v2*.

[Dhingra et al.2016] Bhuwan Dhingra, Hanxiao Liu, William W. Cohen, and Ruslan Salakhutdinov. 2016. Gated-attention readers for text comprehension. *arXiv:1606.01549v1*.

[Dong et al.2015] Li Dong, Furu Wei, Ming Zhou, and Ke Xu. 2015. Question answering over Freebase with multi-column convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 260–269, July.

[Graves et al.2014] Alex Graves, Greg Wayne, and Ivo Danihelka. 2014. Neural Turing machines. *arXiv:1410.5401v2*.

[Graves2013] Alex Graves. 2013. Generating sequences with recurrent neural networks. *arXiv:1308.0850v5*.

[Harman and Voorhees2006] Donna K Harman and Ellen M Voorhees. 2006. TREC: An overview. *Annual review of information science and technology*, 40(1):113–155.

[He et al.2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, June.

[Hermann et al.2015] 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 28*, pages 1693–1701.

[Hill et al.2016] Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. 2016. The Goldilocks Principle: Reading children's books with explicit memory representations. In *Proceedings of ICLR 2016*.

[Hinton et al.2012] Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv:1207.0580v1*.

[Huang et al.2015] Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *arXiv:1508.01991v1*.

[Iyyer et al.2014] Mohit Iyyer, Jordan Boyd-Graber, Leonardo Claudino, Richard Socher, and Hal Daumé III. 2014. A neural network for factoid question answering over paragraphs. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 633–644, October.

[Kadlec et al.2016] Rudolf Kadlec, Martin Schmid, Ondrej Bajgar, and Jan Kleindienst. 2016. Text understanding with the attention sum reader network. *To appear in Proceedings of ACL 2016*.

[Kumar et al.2016] Ankit Kumar, Ozan Irsoy, Jonathan Su, James Bradbury, Robert English, Brian Pierce, Peter Ondruska, Ishaan Gulrajani, and Richard Socher. 2016. Ask me anything: Dynamic memory networks for natural language processing. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 1378–1387, June.

[Lafferty et al.2001] John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML '01)*, pages 282–289, San Francisco, CA, USA.

[Lecun et al.1998] Yann Lecun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, Nov.

[Santos et al.2016] Cicero dos Santos, Ming Tan, Bing Xiang, and Bowen Zhou. 2016. Attentive pooling networks. *arXiv:1602.03609v1*.

[Serban et al.2016] Iulian Vlad Serban, Alberto García-Durán, Çaglar Gülçehre, Sungjin Ahn, Sarath Cjandar, Aaron C. Courville, and Yoshua Bengio. 2016. Generating factoid questions with recurrent neural networks: The 30M factoid question-answer corpus. *To appear in Proceedings of ACL 2016*.

[Socher et al.2013] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning

with neural tensor networks for knowledge base completion. In *Advances in Neural Information Processing Systems*, pages 926–934.

[Sordoni et al.2016] Alessandro Sordoni, Phillip Bachman, and Yoshua Bengio. 2016. Iterative alternating neural attention for machine reading. *arXiv:1606.02245v3*.

[Sukhbaatar et al.2015] Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. 2015. End-to-end memory networks. In *Advances in Neural Information Processing Systems 28*, pages 2440–2448.

[Tian and LI2016] Tian Tian and Yuezhang LI. 2016. Machine comprehension based on learning to rank. *arXiv:1605.03284v2*.

[Tieleman and Hinton2012] Tijmen Tieleman and Geoffrey Hinton. 2012. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural Networks for Machine Learning*.

[Weissenborn2016] Dirk Weissenborn. 2016. Separating answers from queries for neural reading comprehension. *arXiv:1607.03316v2*.

[Weston et al.2015] Jason Weston, Sumit Chopra, and Bordes Antoine. 2015. Memory networks. In *Proceedings of ICLR 2015*.

[Weston et al.2016] Jason Weston, Antoine Bordes, Sumit Chopra, and Tomas Mikolov. 2016. Towards AI-complete question answering: A set of prerequisite toy tasks. In *Proceedings of ICLR 2016*.

[Yao et al.2013] Xuchen Yao, Benjamin Van Durme, Chris Callison-Burch, and Peter Clark. 2013. Answer extraction as sequence tagging with tree edit distance. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 858–867, June.

[Yih et al.2014] Wen-tau Yih, Xiaodong He, and Christopher Meek. 2014. Semantic parsing for single-relation question answering. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 643–648, June.

[Yin et al.2015] Wenpeng Yin, Sebastian Ebert, and Hinrich Schütze. 2015. Attention-based convolutional neural network for machine comprehension. *arXiv:1602.04341v1*.

[Zhou and Xu2015] Jie Zhou and Wei Xu. 2015. End-to-end learning of semantic role labeling using recurrent neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1127–1137, Beijing, China, July.