Document Gated Reader for Open-Domain Question Answering

Bingning Wang[1,2], Ting Yao[2], Qi Zhang[2], Jingfang Xu[2]
Zhixing Tian[1], Kang Liu[1], Jun Zhao[1]
1. National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences
Beijing, 100190, China
2. Sogou Inc.
Beijing, 100084, China
wangbingning,yaoting,qizhang,xujingfang@sogou-inc.com
zhixing.tian,kliu,jzhao@nlpr.ia.ac.cn

ABSTRACT

Open-domain question answering focuses on using diverse information resources to answer any types of question. Recent years, with the development of large-scale data set and various deep neural networks models, some recent advances in open domain question answering system first utilize the distantly supervised dataset as the knowledge resource, then apply deep learning based machine comprehension techniques to generate the right answers, which achieves impressive results compared with traditional feature-based pipeline methods.

However, these deep learning based methods suffer from the inferior quality of the distantly supervised data, and the answer score is un-normalized among multiple documents. Furthermore, unlike previous open-domain question answering system, they process each document independently which may ignore the valuable information in the context. In this paper, we propose a document gated reader to generate the right answer from multiple documents. We propose a document-level gate operation to determine the question-document relevance and embed it into the answer generation process, and optimize it with the global normalization objective. We also develop a bootstrapping scheme to obtain high-quality training data. Experimental results on several question answering datasets show the advantage of the proposed methods.

KEYWORDS

Machine Comprehension; Open-Domain Question Answering; Deep Neural Networks

ACM Reference Format:
Bingning Wang, Ting Yao, Qi Zhang, Jingfang Xu and Zhixing Tian, Kang Liu, Jun Zhao. 2019. Document Gated Reader for Open-Domain Question Answering. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'19), July 21–25, 2019, Paris, France.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3331184.3331190

SIGIR '19, July 21–25, 2019, Paris, France
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-6172-9/19/07…$15.00
https://doi.org/10.1145/3331184.3331190

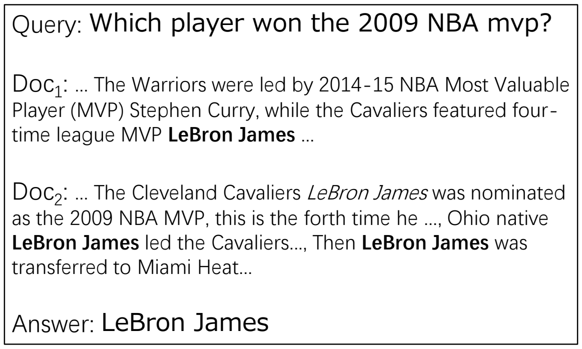
1 INTRODUCTION

Open domain question answering (ODQA) is a long-term research interest in the NLP community [20, 37]. Compared with other types of question answering such as knowledge base QA, the information resource we utilize is diverse, such as unstructured wiki documents [53] or insurance clause [15]. Some research in ODQA was initially fueled by evaluations such as Text REtrieval Conferences[1](TRECs) and the Cross-Language Evaluation Forum[2], etc.

Recently, with the development of large-scale question answering datasets such as SQuAD [38] and QUASAR [13], some new techniques based on deep learning have been proposed to tackle the large-scale ODQA problem [8, 29, 48, 52], which is also referred as neural open-domain question answering. In this application, given a query, they first retrieve a document that is most related to the query and then apply machine comprehension methods on the selected document to generate the answer. For example, Chen et al. [8] use a bi-gram TF-IDF document retriever to select the relevant documents from Wikipedia, then apply an LSTM based document reader on these selected documents to generate the answer. However, it is a pipeline process where the document retrieval and answer generation are uncorrelated. Wang et al. [48] propose a policy gradient based Reinforcement learning model to train the retriever and reader jointly in a single model. Although achieves a better result, in the answer generation process they only consider one documents, which would be insufficient to answer some complicated question. Lin et al. [29] proposed a joint and aggregation model and it achieves the state of the art result in neural open-domain question answering application.

Nevertheless, there are some limitations in the previous methods. First of all, some of the previous works often select the top-ranked documents to answer the question [8, 14]. Unfortunately, it is evident that the existing IR models are not precise, so the answer may not exist in these top-ranked documents. Some other works progress beyond this limitation by introducing a ranker [48, 52], where the document selection and the answer generation are combined to derive the right answer. However, the answer generation process in their models is un-normalized, which may cause the right answer to be assigned with low probability. Specifically, they predict the answer by: $p(a, d|q) = p(d|q) \cdot p(a|q, d)$, where q, d, a denote the question, document and answer respectively. The ranker $p(d|q)$ and the reader $p(a|q, d)$ have their own objective during training, but these two probabilities are multiplied to form the probability of

[1] http://trec.nist.gov
[2] http://www.clef-campaign.org


Query: Which player won the 2009 NBA mvp?

Doc$_1$: ... The Warriors were led by 2014-15 NBA Most Valuable Player (MVP) Stephen Curry, while the Cavaliers featured four-time league MVP **LeBron James** ...

Doc$_2$: ... The Cleveland Cavaliers *LeBron James* was nominated as the 2009 NBA MVP, this is the forth time he ..., Ohio native **LeBron James** led the Cavaliers..., Then **LeBron James** was transferred to Miami Heat...

Answer: LeBron James

Figure 1: Some false positive examples obtained by previous methods. The bold sequences are not related to the question but selected as the positive instance.

(*answer*, *document*) pair (i.e. $p(a, d|q)$) during inference. This objective is not in accordance with the objective in ODQA where we only care about the answer probability $p(a|q)$. As a result, the answer generated in this way is biased.

Secondly, when dealing with the document, previous methods seldom take the context into account. For example, when dealing with a single document and a question, most previous works only process the current document independent from other documents. However, we know that the context information is very important when answering specific questions. Answer accuracy can be improved by using information in multiple documents. In some cases, the answer can only be determined by combining results from multiple documents. Wang et al. [49] shows that evidence repetition and evidence union in answering SearchQA[14] questions could remarkably boost the result. So when processing a single document, it would be better to take the other retrieved documents into account.

Finally, we only have the questions and their corresponding answers when training ODQA models, the ground-truth documents are not provided. Therefore, previous methods [8, 48] try to obtain the corresponding training documents in a distantly supervised way: they first retrieve some documents based on the given question, and then select those that contain the golden answer as the *positive* documents. However, this hypothesis is too strong because those documents may contain some false positive examples. Take the instance in Figure 1 for example: (1) Although Doc$_1$ contains the right answer span, it is not reasonable to answer the question by this document. (2) Doc$_2$ contains multiple answer spans, but some of them are irrelevant to the question. Consequently, the false positive data introduced by previous methods may result in poor performance of the ODQA model.

This paper attempts to overcome the aforementioned limitations in two perspectives. For the model side, we extend the traditional machine comprehension model in a multi-document setting and propose a Document Gated Reader (DGR) for ODQA. In DGR, we first design a convolutional neural network (CNN) based document gate model to determine the relevance score between the question and the document, and we embed it into a machine comprehension reader which is focused on generating the answer. Hence the output of our model considers both the answer and document probability.

Particularly, when processing a single document, we use an LSTM based recurrent neural networks to aggregate the other documents information to the hidden representation of the current document. In this way, each answer in a document is not generated independently but also consider other documents information.

In addition, this model is optimized in a global-normalized manner: instead of a single document normalization, the answer-span score is normalized among all candidate spans that lie in multiple retrieved documents. Besides, we employ the additional document selection label, which corresponds to whether the document is positive or negative, as extra supervision to train the model.

For the data side, rather than merely string matching, we use a bootstrapping scheme to obtain the high-quality distantly supervised data gradually. We first select the data that we are most confident of as the seeds data to train our model. Next, we use the pre-trained model to select an additional document from unlabeled documents. This process is repeatedly adopted until the model does not improve on the development data. With the training proceed, the positive data is more abundant that could strengthen the performance of the QA model.

We perform experiments on several ODQA datasets, and the results show that our DGR achieves a substantial improvement over the state-of-the-art models. We also conduct several experiments to demonstrate: (1) the advantage of the global normalized objective compared with the un-normalized one. (2) The proposed DGR could retrieve relevant documents with higher precision. (4) The open domain question answering result could be improved when we take multiple documents information into account. (4) The advantage of the proposed bootstrapping based data generation scheme.

Overall, there are three contributions in this paper:

- We propose a document gate operation into the machine comprehension architecture, marrying document selection and answer generation in the open-domain question answering application.
- We utilize multi-documents information to determine the question-document relevance. And we propose a global normalization objective to optimize the likelihood of the answer.
- We propose a bootstrapping based data generation scheme to generate high-quality data, which shows advantage compared with naive distantly supervised data.

2 RELATED WORK

Open-domain question answering has been a long-term focus in artificial intelligence that can date back to 1960s, where Green Jr et al. [19] proposed a simple system to answer questions about baseball games. Since then, many works have been done to use diverse data resource to answer any type of questions. For example, Cui et al. [10] proposed a method based on the dependency relation of each keyword in each document to derive the answer probability. Verberne et al. [46] surveyed a variety of learning-to-rank paradigms concluding that a pairwise approach using Support Vector Regression is the most appropriate strategy for the task of CLEF question answering application. However, these methods are limited by small datasets which require sophisticated feature engineering, which could not be generalized to more domains.

Neural Open-domain question answering. Recently, due to the development of deep neural networks and large-scale question answering datasets, the neural networks based ODQA model has become the mainstream. Chen et al. [8] proposed DrQA that uses a bi-gram TF-IDF to retrieve relevant documents and then applies machine reading on the selected document to generate the answer. However, it is a pipeline model that is vulnerable to adversaries. Wang et al. [48] proposed a joint model with reinforcement learning to train the two parts together, but this model is limited by answer un-normalization. Wang et al. [49] proposed an evidence aggregation methods with the hypothesis that if a span exists more times in the retrieved documents, then it is more likely to be the answer. But their aggregation is done after extracting the answer, which may already contain some noise. Clark and Gardner [9] proposed a similar multi-document framework. However, this normalization is done in the output, unlike our model in which we introduce the attention globalization to process each document dependently. Lin et al. [29] proposed a distant supervised open question answering system where they aggregate many candidate answers in one document for prediction. However, they neither normalize the answer scores across multi-documents nor take other document information into account when processing the current document, which is considered by our work via the document gate.

Machine Reading. Recently, more and more researchers start to employ machine reading methods on the task of ODQA, which are also closely related to our work. Since the MCTest [39] was proposed, many researchers have been focused on this specific task. Hermann et al. [21] proposed a large cloze style dataset CNN/Daily Mail where the questions and answers are automatically generated from news article. Recently, some human create large-scale authentic datasets have been released, such as SQuAD [38], NewsQA [44] and MARCO [33]. Models based on these datasets are mostly built upon neural networks, such as self-attention [50], bi-directional attention [40], fusion attention [23], etc. However, in MC we have provided with the groud-truth document, which obviates the need for document selection that is inconsistent with ODQA.

3 DOCUMENT GATED READER

In ODQA, given a question q, we first retrieve a lot of documents D, where D may contain both positive documents d^+ that can be used to answer the question, and negative documents d^- that are noise and irrelevant to the question. Similar with previous settings, we suppose that the right answer is contained in the document, so we only need to predict the start and end position of the answer in the document.

The document gated reader is similar with the prevalent machine comprehension methods which consists of a lot of layers to represent the document and question. In addition, we use a document gate to determine the relevance between the question and the document, and embed it into final answer prediction. We illustrate the conceptual architecture of DGR in Figure 2, and detail the several layers in the following sections.

3.1 Reader

3.1.1 ***Word Embedding Layer.*** The first layer in reader is a word embedding layer that embeds the input words into vectors. Denote the question sequence as $w_1^Q, ..., w_m^Q$ where m is the length of the question, and document as $w_1^D, ..., w_n^D$ where n is the number of words in the document. We use 300d Glove embedding [35] which shows advantage compared with word2vec [12]. In addition, it has shown that the language model based representation and translation based representations could also strengthen the natural language processing results [31, 36]. So we also concatenate Cove[31] and Elmos [36] representations to the word representations. Inspired by Wang et al. [47, 51], for ith word in the document, we also concatenate the aligned question word embedding which is calculated by $\sum_1^m a_{ij}\mathbf{w}_j^Q$ where $\mathbf{w}_j^Q$ is jth question Glove embedding, and $a_{i,j}$ is calculated by: $a_{i,j} \propto \exp[(\mathbf{w}_i^D)^T \cdot \mathbf{w}_j^Q]$. After the above representation, each document word is represented as a 1200d embedding, which is denoted as $\{\mathbf{e}_1, ..., \mathbf{e}_n\}$.

3.1.2 ***Low-Level Representation Layer.*** This layer aims at representing the word in the sequence with its contextual information. We use two separated bi-directional LSTMs [22] as the building block for question and document:

$$\begin{aligned}\mathbf{l}_1^Q, ..., \mathbf{l}_m^Q &= \text{Bi-LSTM}^Q(\mathbf{w}_1^Q, ..., \mathbf{w}_m^Q)\\ \mathbf{l}_1^D, ..., \mathbf{l}_n^D &= \text{Bi-LSTM}^D(\mathbf{e}_1, ..., \mathbf{e}_n)\end{aligned} \tag{1}$$

In which each Bi-LSTM could be represented as:

$$\begin{aligned}
\overrightarrow{\mathbf{f}_t} &= \sigma(\overrightarrow{\mathbf{W}}_f\mathbf{x}_{t-1} + \overrightarrow{\mathbf{U}}_f\mathbf{h}_{t-1}) \quad \overleftarrow{\mathbf{f}}_t = \sigma(\overleftarrow{\mathbf{W}}_f\mathbf{x}_{t+1} + \overleftarrow{\mathbf{U}}_f\mathbf{h}_{t+1})\\
\overrightarrow{\mathbf{i}_t} &= \sigma(\overrightarrow{\mathbf{W}}_i\mathbf{x}_{t-1} + \overrightarrow{\mathbf{U}}_i\mathbf{h}_{t-1}) \quad \overleftarrow{\mathbf{i}_t} = \sigma(\overleftarrow{\mathbf{W}}_i\mathbf{x}_{t+1} + \overleftarrow{\mathbf{U}}_i\mathbf{h}_{t+1})\\
\overrightarrow{\mathbf{o}_t} &= \sigma(\overrightarrow{\mathbf{W}}_o\mathbf{x}_{t-1} + \overrightarrow{\mathbf{U}}_o\mathbf{h}_{t-1}) \quad \overleftarrow{\mathbf{o}_t} = \sigma(\overleftarrow{\mathbf{W}}_o\mathbf{x}_{t+1} + \overleftarrow{\mathbf{U}}_o\mathbf{h}_{t+1})\\
\overrightarrow{\mathbf{c}_t} &= \overrightarrow{\mathbf{f}_t} \odot \overrightarrow{\mathbf{c}_{t-1}} + \overrightarrow{\mathbf{i}_t} \odot \sigma(\overrightarrow{\mathbf{W}}_c\mathbf{x}_t + \overrightarrow{\mathbf{U}}_c\mathbf{h}_{t-1})\\
\overleftarrow{\mathbf{c}_t} &= \overleftarrow{\mathbf{f}}_t \odot \overleftarrow{\mathbf{c}_{t+1}} + \overleftarrow{\mathbf{i}}_t \odot \sigma(\overleftarrow{\mathbf{W}}_c\mathbf{x}_t + \overleftarrow{\mathbf{U}}_c\mathbf{h}_{t+1})\\
\overrightarrow{\mathbf{l}_t} &= \overrightarrow{\mathbf{o}_t} \odot \sigma(\overrightarrow{\mathbf{c}_t}) \quad \overleftarrow{\mathbf{l}_t} = \overleftarrow{\mathbf{o}_t} \odot \sigma(\overleftarrow{\mathbf{c}_t})\\
\mathbf{l}_t &= [\overrightarrow{\mathbf{l}_t}; \overleftarrow{\mathbf{l}_t}]
\end{aligned} \tag{2}$$

where we omit the bias vector for simplicity. We concatenate the forward $(\overrightarrow{\mathbf{l}_t})$ and backward $(\overleftarrow{\mathbf{l}_t})$ representation to form the final low-level representation for question and document.

3.1.3 ***Question Attention Layer.*** After the low-level representation layer, we then represent the document and question in an interactive way which is also called *attention* mechanism. It has been verified quantitatively and qualitatively in previous works that the attention is very useful in natural language processing [3, 45], especially in question answering application, the model equipped with attention mechanism could significantly outperform the one without attention [9, 40]. In this paper, we use the non-linear multiplication to calculate the question and document attention score. The output of the document in this layer could be denoted as:

$$\begin{gathered}\alpha_{i,j} = f(\mathbf{U} \cdot \mathbf{l}_i^D)^T \cdot f(\mathbf{V} \cdot \mathbf{l}_j^Q)\\ \mathbf{q}_i = \sum_j^m \alpha_{i,j}\mathbf{l}_j^Q \qquad \mathbf{c}_i = [\mathbf{l}_i^D; \mathbf{q}_i]\end{gathered} \tag{3}$$

where $\mathbf{U}$ and $\mathbf{V}$ are two weight matrices and $f(\cdot)$ is a Relu nonlinear function. $\alpha_{i,j}$ is the attention score between the ith document word and jth question word. $\mathbf{q}_i$ is the corresponding attentive question

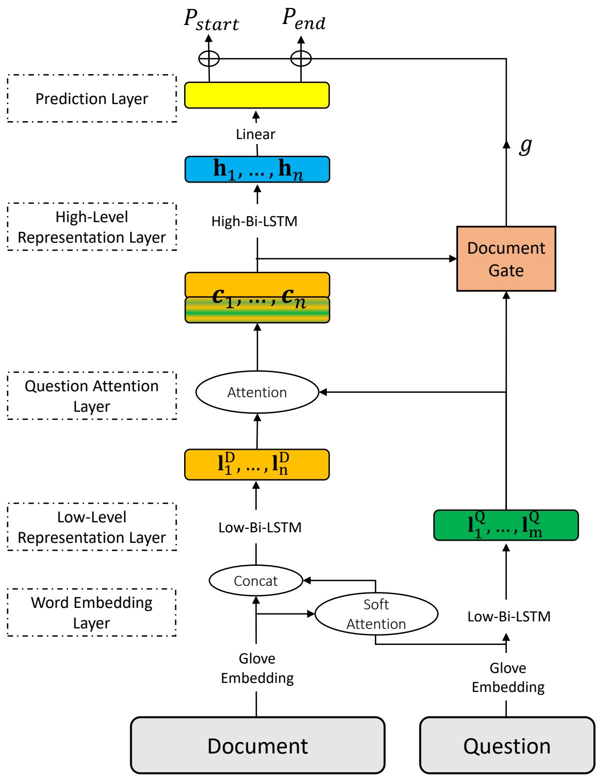


Figure 2: Overview of the proposed Document Gated Reader.

representations for ith word in the document. We concatenate this representation with the low-level document representation to derive the fused document representation $\mathbf{c}_i$ where the question information has been embedded.

3.1.4 ***High Level Representation Layer****.* After the low-level representation and attention layer, we aggregate these information for the answer prediction. Similar with low-level representation, we use a two-layer-bi-directional LSTMs to process the document in the high level:

$$\mathbf{h}_1, \ldots, \mathbf{h}_n = \text{2-Layer-Bi-LSTM}(\mathbf{c}_1, \ldots, \mathbf{c}_n) \tag{4}$$

The high-level representation $\mathbf{h}$ contains both the document and question information and thus could be used for predicting the answer.

3.1.5 ***Prediction Layer****.* The prediction layer is built to predict the span of tokens that is likely to be the correct answer. Similar with previous methods [8, 48, 52], we use two seperate classifiers to predict the start and end position of the answer span independently. Concretely, the start and end score of the ith word is:

$$\begin{aligned} s_{start}(i) &= \text{Relu}[(\mathbf{w}_{start})^T \cdot \mathbf{h}_i] \\ s_{end}(i) &= \text{Relu}[(\mathbf{w}_{end})^T \cdot \mathbf{h}_i] \end{aligned} \tag{5}$$

where $\mathbf{w}_{start}$ and $\mathbf{w}_{end}$ are two weight vectors. We use Relu function to guarantee that the start and end score is positive, so the span score (i.e. the product of the start and end score) is also positive, which is comparable across different spans in different documents.

3.2 Document Gate

One innovation of this paper is that we propose a document gate operation, which is embedded into the machine reading framework, to determine the question-document relevance. Concretely, we use $\mathbf{L} = \{\mathbf{l}_1^Q, \ldots, \mathbf{l}_m^Q\}$ and $\mathbf{C} = \{\mathbf{c}_1, \ldots, \mathbf{c}_n\}$ as input to the document gate. The document gate consists of two components, the first one is a convolutional attention pooling to derive the local document-question relevance vector, and the second one is a LSTM based recurrent neural networks to take the global multi-document information into the relevance score.

3.2.1 ***Convolutional Attention Pooling****.* We use a non-linear layer to calculate the word-level correlation score:

$$\beta_{i,j} = g_1(\mathbf{c}_i)^T \cdot g_2(\mathbf{l}_j^Q) \tag{6}$$

where g_1 and g_2 are two multi-layer perceptrons (MLP) and $\beta_{i,j}$ is the correlation score between ith document word and jth question word. Next, we use these word-by-word relevance to determine the sequence-by-sequence relevance between the document and the question. Traditional methods sometimes resort to max-pooling or mean-pooling directly on top of $\beta_{i,j}$ to get the scalar score. However, when the document is long and the question has many words relevant to the documents, this pooling operation may ignore some useful information [30]. In this paper, we treat the attention score matrix $\mathbf{B} = \{\beta_{i,j} | i = 1, \ldots, m, j = 1, \ldots, n\} \in \mathbb{R}^{n \times m}$ as a feature map, and apply 100 3×3 convolutional filters $\mathbf{f}$ on this feature map[3], which results in a 3-d tensor $\mathbf{A} \in \mathbb{R}^{100 \times (n-2) \times (m-2)}$:

$$\mathbf{A}_{i,j} = \text{Relu}(\mathbf{f} \otimes \mathbf{B}_{i-1:i+1, j-1:j+1} + \mathbf{b}) \tag{7}$$

where $\otimes$ is the element-wise multiplication and $\mathbf{b} \in \mathbb{R}^{100}$ is the bias vector. After this convolution representation, we apply max pooling on the second and third dimension of $\mathbf{A}$ to derive the question-document-relevance vector:

$$\mathbf{a} = \max_{i,j} \mathbf{A}_{i,j} \tag{8}$$

3.2.2 ***LSTM Attention Globalization****.* After deriving the document-question local relevance vector $\mathbf{a}$, we then interact this vector with other documents relevance vector to obtain a global representation. Concretely, denote $\mathbf{a}_k$ is the kth document relevance vector, its global representation could be obtained by:

$$\begin{aligned} \tilde{\mathbf{a}}_1, \ldots, \tilde{\mathbf{a}}_K &= \text{Bi-LSTM}(\mathbf{a}_1, \ldots, \mathbf{a}_K) \\ g_k &= \sigma(\mathbf{w}^T \cdot \tilde{\mathbf{a}}_k) \end{aligned} \tag{9}$$

where K is the number of candidate documents. $\mathbf{w}$ is a 100d weight vector. We use sigmoid function σ to make the score lie in $(0, 1)$. g_k is the output of the document gate which denote the question-document-relevance score of the kth document. As we use the recurrent Bi-LSTM network to incorporate other document information, each document is modeled dependently.

The document gate operation is illustrated in Figure 3.

After the machine reading and document gate layer, we combine them together to form the span score of a candidate answer:

$$\begin{aligned} s'_{start}(i) &= s_{start}(i) * g, \\ s'_{end}(i) &= s_{end}(i) * g \end{aligned} \tag{10}$$

[3] So we presuppose that the document and question length is larger than 3, which is evident in practical settings.

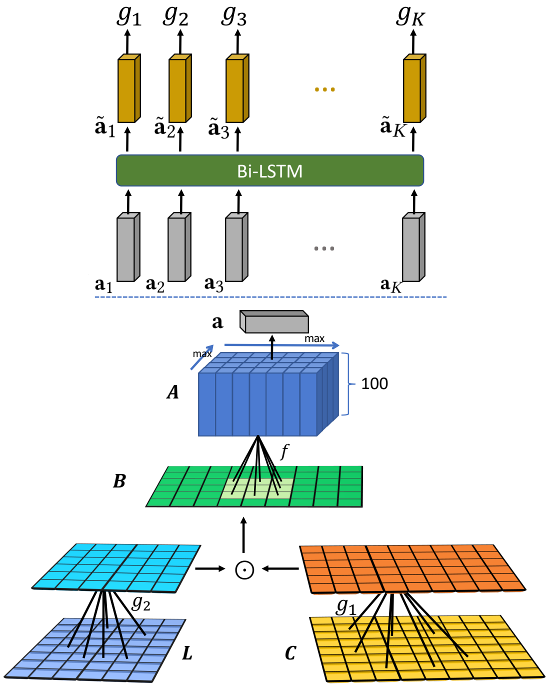


Figure 3: Illustration of document gate operation.

We omit the document index k for brevity. This output score has a straightforward explanation: the score of a candidate answer is the combination of reading comprehension score and document selection score g. Thus, we take both the two parts into account for the final answer prediction.

4 MODEL TRAINING

4.1 Optimization Objective

During training period, we can express the probability of ground-truth answer span $a*$ as:

$$p(a*) = \frac{\exp^{s'_{start}(a^*_{start})\cdot s'_{end}(a^*_{end})}}{\sum^{K} \sum \exp^{s'_{start}(a_{start})\cdot s'_{end}(a_{end})}} \tag{11}$$

where a^*_{start} and a^*_{end} is the start and end position of the ground truth answer. This score is normalized across multiple documents, thus it is global normalized probability. The objective is the negative log likelihood of the ground truth answer:

$$\mathcal{L}_r = -\log p(a*) \tag{12}$$

However, this objective do not directly optimize the document selection, thus the document gate score g may not correspond well with the document rank. In this paper, we add an additional supervision to the document gate:

$$\mathcal{L}_g = -\sum_{g} y \log(g) + (1-y)\log(1-g) \tag{13}$$

where y is the binary label of the document denote whether it is relevant (1) or irrelevant (0) with the question. We incorporate this document selection objective with the answer span objective with weight λ to form the final objective:

$$\mathcal{L} = \mathcal{L}_r + \lambda \cdot \mathcal{L}_g \tag{14}$$

4.2 Data Generation Scheme

As has been demonstrated in the Introduction, the first step of the open domain question answering is to generate high-quality data for the reader. In some datasets such as SQuAD or MARCO, the ground truth documents have already been provided so that we can treat those documents as the seeds positive documents. In some other types of question answering datasets such as WikiMovies [32] or WebQuestions [5], there are no ground truth documents, so we select the one that has the highest rank from the IR model as the positive documents seeds. We find this heuristic is very effective that there are only four false positive instances out of 100 randomly sampled positive (q, d^+, a) triplets.

To generate the negative documents, we first use the traditional IR based model to retrieve some documents according to query q. Then we select those retrieved documents that do not contain the right answer as the negative documents. For example, given a query: *who is the first president of the USA?* we select those documents that do not contain *George Washington* as the negative documents.

Based on the initial (q, D_0, a) seeds triplets, in which the positive document set in D_0 only contains the most confident positive document. We first train DGR on this seeds data. Then we apply the initiated model to the unlabeled documents to select an additional positive document that has the highest relevance score g in Equation 9. To prevent degeneration of the data quality caused by bootstrapping [54], we only select the document that contains the ground truth answers. We add this additional positive document to the original positive documents pool and train our model again. We repeat this process until no positive document could be selected or DGR's performance does not improve on the development sets. The whole process is illustrated in Algorithm 1.

Algorithm 1 Bootstrapping Data Generation.

```
Input: The unlabeled training data $\langle q, a\rangle$, the retrieved documents
       set $D$ for question $q$, $t = 0$.
Using the seeds triplet to train DGR (dgr$_0$).
do
    Apply dgr$_t$ on $D$, select the document $d_t$ that has highest $g$
    score and is not in $D_t$.
    Add $d_t$ to $D_t$ to form $D_{t+1}$.
    Train DGR on $D_{t+1}$ to obtain dgr$_{t+1}$.
    $t = t + 1$.
while $t \leq$ Threshold and the performance of dgr$_{t+1}$ does not
decrease in development data.
Return: The enhanced datasets $\langle q, D_t, a\rangle$
```

5 EXPERIMENTS

5.1 Knowledge Resource

Similar with previous methods on neural ODQA [8, 48, 49], we adopt the Wikipedia as the knowledge for our question answering system, as it contains formal informational documents in various domains

Dataset	#Train	#Dev	#Test
SQuAD	87,599	-	10,570
SearchQA	99,811	13,893	27,247
WebQuestions	3,778	-	2,032
WikiMovies	96,185	-	9,952

Table 1: Statistics of the dataset.

and easy to access. We use the wikidumps[4] with WikiExtractor[5] tools, and we extract only the text passages and ignore lists, tables, and headers. After processing, we keep 30,460,130 documents of English as the knowledge source. We use the tri-gram based Tf-IDF model provided by Elasticsearch [17] as the IR tool. We have also tried other types of retrieval mechanism such as bi-gram TF-IDF adopted in [8] or deep learning based representation methods based on Fasttext[18, 26], but the tri-gram based TF-IDF shows the best compromise between efficiency and performance.

5.2 Commen Setup

For all datasets, we use the spacy [6] tools to segment the documents and sentences. We use the off-the-shelf Glove[7], Cove[8] and Elmo[9] embeddings without fine-tuning its weight. We set all the LSTM hidden size to 128, and apply dropout [41] with 0.2 on the word embedding and CNN outputs. For the LSTM, we use the variational dropout [16] with rate 0.3 to the hidden states which shows the advantage to vanilla dropout in recurrent neural networks. Batch size is set to 16 to avoid memory overflow. The model is optimized by Adamax [28] with a warmup of 10000 steps. We halved the learning rate with patience ten epochs when performance was not improved in the development set. The weight λ in Equation 14 is set to 0.3 based on the development set. During the training period, we use the data generated by bootstrapping, for the documents that contain more than one ground-truth answer spans, we treat each one as positive and maximize their negative log likelihood. During inference, we use IR tools to select 100 most relevant documents and then apply DGR on them to derive the answer. All the models are implemented by Pytorch-0.4.0 [34] on 8 nodes Nvidia-V100 GPU.

5.3 Datasets

There are many datasets that could be used in open-domain question answering evaluation. However, some of them are based on trivia questions, such as TrivaQA [27] or Quasar [13], where some answers could not be answered by a single document or paragraph. So we use four datasets for evaluation:

- **SQuAD** [38]: it is a large scale QA dataset curated by humans. A favorable property of SQuAD is that it has provided the ground truth documents that is extracted from Wikipedia. So we use these ground-truth documents as seeds data for bootstrapping. For each question, we randomly sample 25 documents that do not contain the answer as the negative documents. We use the development set for evaluation, which contains 10447 questions after processing.
- **SearchQA**[14] is a large-scale open domain question answering dataset, which consists of question-answer pairs from Jeopardy! They augment each question-answer pairs with 50 from Google Search API and end up with 140k+ question-answer pairs, and in total 6.9M snippets. As in this datasets, the ground truth documents are not provided, we use the documents that contains the answer span as the seeds positive document.
- **WebQuestions** [5] is a factoid based QA dataset which is curated based on Freebase [7]. This dataset is originally proposed to inference on the knowledge base, so we do not have access to the ground-truth document for a question. We use the top-ranked document that contains the ground truth answer retrieved by the IR tools as the seeds positive document. We retain 3,683 questions for training.
- **WikiMovies** [32]: which contains about 100k question-answer pairs from movie site. The training data generation scheme we use for WikiMovies is same with WebQuestions.

The statistics of these datasets are shown in Table 1.

5.4 Baselines and Evaluation Metrics

We adopt four recently proposed neural models as baselines:

- **BiDAF** [40] is a popular machine reading model that uses bi-directional attention flow which is built upon LSTM. As the original BiDAF is focused on single document machine comprehension that is not suitable to open-domain question answering setting. In this paper, we use the top-ranked document retrieved by the TF-IDF model and then apply BiDAF to derive the answer.
- **DrQA** [8] is an ODQA model where the document is ranked by off-the-shelf IR tools. It has additional heuristic rules to select relevant documents such as entity matching. The reader is a simply multi-layer LSTM enhanced with attention mechanism.
- $\textbf{DrQA}_{\textbf{MTL}}$ is a multi-task-learning extension to the DrQA model, in which the model is trained in SQuAD, WebQuestions etc.
- $\textbf{SR}^2$ [48] is a pipeline model where the document ranker and reader are trained separately.
- $\textbf{R}^3$ [48] is a joint model where the document ranker is trained under policy gradient and the document reader is trained by maximum likelihood estimation.
- **DS-QA** [29] employs a paragraph selector to filter out those noisy paragraphs and a paragraph reader to extract the correct answer from those denoised paragraphs, and the final answer is obtained by combining each answer in multiple documents.

In addition to the above baselines, we also conduct several ablation studies:

(1) **w/o BD**: to evaluate the quality of the generated data in Section 4.2, we do not adopt the proposed bootstrapped data, but similar with previous ODQA models [8, 48] we train the DGR only on the ground truth or top-ranked IR document.

(2) $\textbf{DGR}_{\textbf{PPL}}$. Instead of joint training the document gate and document reader, it is a pipeline model that trains the document reader and document gate, i.e., $\mathcal{L}_r$ and $\mathcal{L}_g$ in Equation 14, separately.

[4] 2018-06-20 dumps at https://dumps.wikimedia.org/enwiki/

[5] https://github.com/attardi/wikiextractor

[6] https://spacy.io/

[7] https://nlp.stanford.edu/projects/glove/

[8] https://github.com/salesforce/cove

[9] https://allennlp.org/elmo

		BiDAF	DrQA	DrQA$_{\text{MTL}}$	SR^2	R^3	DS-QA	w/o BD	DGR$_{\text{PPL}}$	w/o DG	**B**-Mean	DGR	↑ Δ
SQuAD	EM	23.7	28.4	29.8	27.2	29.1	-	31.5	31.8	31.1	31.9	**33.9**	13.7%
	F1	30.9	-	-	35.8	37.5	-	40.6	39.7	39.0	41.1	**43.6**	16.3%
SearchQA	EM	28.6	-	-	-	49.0	58.8	53.3	53.8	55.2	56.9	**61.4**	4.4%
	F1	34.6	-	-	-	55.3	64.5	60.2	59.9	59.0	61.3	**65.7**	1.9%
WebQuestion	EM	10.9	19.5	20.7	15.6	17.1	18.5	21.7	18.9	18.2	20.9	**22.6**	9.2%
	F1	19.8	-	-	22.5	24.5	25.6	25.3	24.8	21.7	24.2	**27.3**	6.6%
WikiMoviews	EM	30.6	34.3	36.5	38.1	**38.8**	-	32.5	33.4	34.1	35.0	36.2	-
	F1	36.4	-	-	39.3	39.9	-	37.1	39.2	38.3	40.6	**42.5**	6.5%

Table 2: The result of different models on open domain QA. The last column is the relative improvement of DGR over previous state-of-the-arts. All experiments have been gone through the significant test, i.e., one-tailed paired t-test with a default 95% significance level is used here. It needs to mention that for SQuAD some questions are document dependent (i.e. *Who was the university's 5th president?*), so the result is poor compared with the original machine reading settings.

This corresponding to only takes the document gate score, i.e. the document-question-relevance, as a fixed feature in the question answering process.

(3) **w/o DG**. To demonstrate the importance of the document gate, we remove the document gate operation in Equation 10. Thus our model is reduced to original multi-document machine comprehension model [9] without the document level relevance information.

(4) Instead of the CNN attention, we use a mean-pooling on the attention matrix **B** (**B**-Mean) to derive the relevance score.

We use two metrics for evaluation:

(1) **Exact Match (EM)** measures whether the generated answer is strictly matched.

(2) **F1** score measures the weighted average of precision and recall at the token level.

5.5 Results

We can see from the table that our model outperform the previous models in almost all cases. The widely used BiDAF model behaves very bad at this application. Because it is a standard machine comprehension model that requires the oracle document, which is not satisfied in open domain QA. DrQA behaves a little better. It has the similar training objective but uses many heuristic rules, such as entity overlap between paragraph and question, to select the relevant document, which is too specific and hard to apply. Compared with other end-to-end models such as SR^2 or R^3, our model is more simple but achieves better performance. In fact, we find that in R^3 the variance is hard to control, which necessitate meticulous hyperparameter tuning and pre-training by supervised methods.

Our model also achieves consistent improvements compared with the state-of-the-arts model DS-QA [29]. As in DS-QA, although they also aggregate each document answer into the final answer prediction: $p(a|q) = \sum_i p(a|q, d_i) \cdot p(d_i|q)$, but they process each document independently, which would ignore the valuable global information. In this paper, as we use the LSTM based document gate and a global normalization objective to take other document information into account when predicting the answer in a single document, which in turn reflects the importance of evidence aggregation in open-domain question answering.

When trained on the bootstrapped data, our model achieves a better result (DGR better than w/o BD). As the current deep neural networks methods are sometimes data-driven (or data hungry) [4], more abundant training data sometimes result in better results. Our proposed data generation scheme could generate more high-quality positive data, which strengthen the inference ability of our model. The quality of the data will be further evaluated in section 5.9.

We can also see that the DGR achieves a better result than the one without document gate (*w/o DG*), as in DGR we provide the additional document selection information for the reader to generate the answer. In addition, DGR also outperforms the pipeline model DGR$_{\text{PPL}}$, which reveals the importance of the joint training for document selection and answer generation in ODQA.

Furthermore, the proposed CNN based attentive pooling method shows the advantage over traditional mean-pooling methods. We think the reason is that the seq-to-seq relevance is dynamically configured by the word-to-word relevance [6], and the CNN based attention mechanism could somewhat capture this phenomenon.

5.6 Document Selection Evaluation

The first step in ODQA is to select the relevant documents. In this section, we evaluate the performance of the proposed DGR on whether it could retrieve the relevant documents given a question. We use SQuAD as the benchmark because it has already provided the ground-truth documents. The documents without ground truth answer are treated as the negative document. We evaluate the rank performance by **Precision@N**: whether we can retrieve the ground-truth document in top-ranked N results. **AR**: the average rank of the ground-truth documents.

We select four models as baselines:

(1) BM-25 uses the bag-of-words representation for question and document, which is an extension of Tf-IDF model that takes the sequence length into account.

(2) DSSM [24] treats the input sentence as bag-of-words and uses an MLP to embed the question and document into a hidden space and use cosine similarity to measure the relevance.

(3) R^3-Ranker is a reinforcement learning based model proposed by [48]. The objective (reward) of the ranker is based on the reader: if the reader could generate the right answer on the selected document, then the ranker gets positive reward. Otherwise, it gets negative reward.

(4) ATT-LSTM-CNN [43] is a state-of-the-arts attention model based on LSTM and CNN, and it has achieved competitive result on

	P@1	P@5	P@10	AR
BM25	21.32	58.34	69.78	13.8
DSSM	37.26	79.67	81.39	9.95
ATT-LSTM-CNN	44.35	80.27	84.62	7.91
R^3-Ranker	31.29	62.58	75.41	12.05
w/o DG	43.78	76.74	82.30	8.13
B-MAX	45.39	79.25	85.13	4.23
B-MEAN	47.50	79.34	86.21	4.07
DGR	**53.28**	**82.78**	**90.35**	**3.65**

Table 3: Document retrieval result. w/o DG means we do not apply document gate operation and only select the document with highest span score. B-MAX means we do not apply CNN on the attention feature map B but uses max-pooling, B-mean denotes mean-pooling.

several answer selection datasets. The result of document retrieval is shown in Table 3.

We can see from the table that our model achieves the best results. (1): For DSSM, we find that this method is very hard to train. Although we have adopted the word hashing tricks to reduce the vocabulary size, the model is still downgraded to overfitting quickly. (2): For the Reinforcement learning based R^3-Ranker, at the beginning of the training, the agent (i.e. the document ranker) could barely select the right document so it could not obtain the positive reward. After some epochs, the agent gets volatile rewards so the variance becomes high. The high variance of the policy gradient is the main obstacle for the model to converge [42]. In experiments, we observe that in most situation it could not outperform the supervised methods.

Besides, the model without document gate operation behaves not so good. Although the global-normalized answer span score has already embedded the document selection into account, that is, the answer span score in a document can also represent the probability of this document to be selected. Nonetheless, the additional document-question score g in Equation 10 represents the relevance directly, which is excel at selecting the relevant document.

5.7 Attention Globalization Evaluation

In Section 3.2.2, instead of a single document attention vector $\mathbf{a}$, we use the contextualized attention vector $\tilde{\mathbf{a}}$ to derive the document-question relevance score. We denote the $\mathbf{a}$ as a local attention vector because it only takes the current document into account, and $\tilde{\mathbf{a}}$ as the global attention vector because it was built upon the LSTM that have other document information.

To demonstrate the effectiveness of the attention globalization, we evaluate it in two aspects: (1) Whether it could improve the document retrieval result. (2) Whether it could improve the question answering results.

In order to evaluate the first aspect, similar to the previous section we use the Precision@N and AR as the metric. In this setting, we remove the prediction layer of our model and only optimize the document selection objective in Equation 13. The second aspect is evaluated by EM and F1 score.

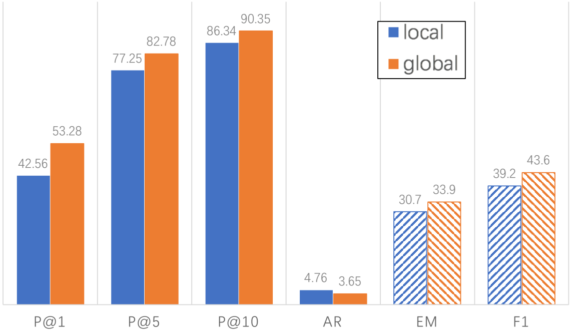


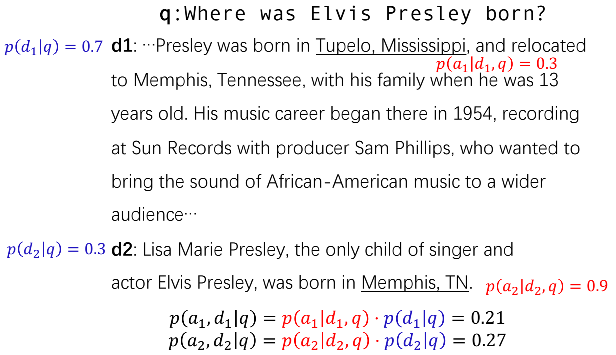
Figure 4: The result with different attention mechanism.

We use the proposed architecture without any bootstrapped data for evaluation. The performance of the different model in SQuAD is shown in Figure 4:

We can see that when taken other document information into account (via LSTM), the global model behaves good at both document selection and final answer generation. This result has also been verified by previous conclusion on answer selection [1], information retrieval [2, 11] and question answering [49]. Where they found that global information is sometimes useful for dealing with the current document or sentence. In the ODQA setting, a single document retrieved by the IR tools sometimes not sufficient to answer the question, so take other documents into account is useful for determining the current document relevance to the question. Furthermore, the global information in this work is imported via an LSTM in Equation 9, it brings not much burden to the whole architecture, which is efficient to implement in practice.

5.8 Global Normalization Evaluation

Next we focus on the score un-normalization problem. In a word, the answer score un-normalization problem lie in the fact that the question-document-answer score is not in accordance with question-answer score. Lets take a concrete example in Figure 5:



q:Where was Elvis Presley born?

$p(d_1|q) = 0.7$ **d1**: ···Presley was born in Tupelo, Mississippi, and relocated $p(a_1|d_1,q) = 0.3$ to Memphis, Tennessee, with his family when he was 13 years old. His music career began there in 1954, recording at Sun Records with producer Sam Phillips, who wanted to bring the sound of African-American music to a wider audience···

$p(d_2|q) = 0.3$ **d2**: Lisa Marie Presley, the only child of singer and actor Elvis Presley, was born in Memphis, TN. $p(a_2|d_2,q) = 0.9$

$$p(a_1, d_1|q) = p(a_1|d_1, q) \cdot p(d_1|q) = 0.21$$
$$p(a_2, d_2|q) = p(a_2|d_2, q) \cdot p(d_2|q) = 0.27$$

Figure 5: An example of answer score un-normalization problem. Candidate answers have been underlined. The right answer should lie in doc1, but this document is comparative long so the system makes the wrong prediction.

Suppose the doc2 contains only 17 words and doc1 contains more than 200 words. The question-document-answer probability

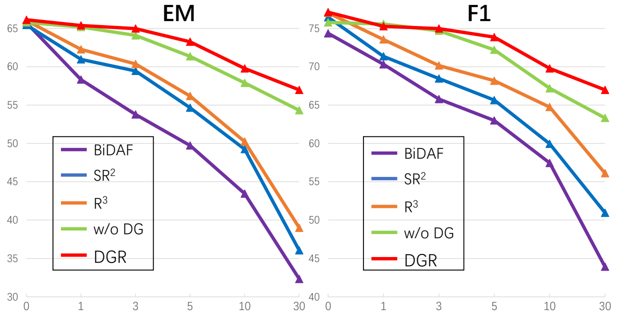


Figure 6: The result with different number of negative documents being added to the ground-truth document in SQuAD. X-axis is the number of negative documents. BiDAF, SR^2 and R^3 are normalized upon a single document.

of doc1-$p(a_1|d_1, q)$ may be much smaller than doc2-$p(a_2|d_2, q)$ due to the much larger dominator item in probability normalization, and the document selection probability for two documents is 0.7 and 0.3 respectively, although the document selection has made the right prediction $p(d_1|q) > p(d_2|q)$. Due to the answer score is merely normalized in a single document (not all documents), the system made a wrong prediction.

To analysis this problem quantitatively, we randomly add several negative documents, which are retrieved by IR tools and do not contain the ground-truth answer, to the ground-truth positive document in SQuAD, and evaluate the model performance on this mixed dataset. The EM and F1 score are shown in Figure 6.

We can see from the figure that with the number of negative document increasing, the performance of the model without global normalization drops drastically. Previous models never exploit the global optimum: during training, they normalize the answer span score $p(a|q, d)$ only under its corresponding document, so it is not exposed to the negative documents. However, during inference, the answer is predicted from multiple documents, which is very vulnerable to distractions. This phenomenon was also discovered by Clark and Gardner [9]. When they concatenate several irrelevant documents to the ground-truth document, the behavior of traditional model drops quickly.

In our DGR, we train the model with a global view that has been formulated in Equation 11: we normalize the result among all documents. Thus the incorrect answer span score is low no matter how likely it is to be an answer in this document, so it is more robust to the negative adversaries.

5.9 Data Quality Evaluation

The success of DGR is partially due to the superior quality of the generated data. In this subsection, we evaluate the quality of the data in an implicit way: if the QA model trained on one dataset has better performance, then we can declare that this data is superior. To eliminate the influence of the model, we use two kinds of models, namely BiDAF and the proposed DGR, and ensemble their results together for evaluation.

For comparison, we use the data generation strategy proposed by previous methods [8, 48]: the retrieved documents that contain the answer are selected as the positive documents, which we refer as **Greedy**. The EM and F1 results are shown in Figure 7. We can see from the figure that compared with the traditional greedy search, the proposed data generation method has consistently better performance. With the bootstrapping proceed, the advantage is more significant which means the model trained on the bootstrapping data becomes more accurate and could select higher quality question-document pairs.

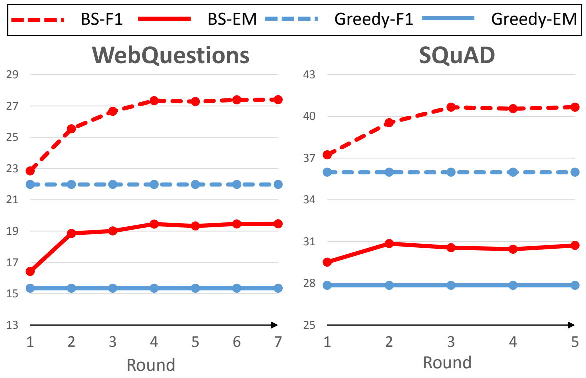


Figure 7: The result of data generation. BS stands for our proposed bootstrapping based data generation scheme. The X-axis is the training step. In Greedy the data is static, so we train the model until convergence. For BS, we terminate the generation process at seven rounds for WebQuestions and five rounds for SQuAD.

Another discovery in this experiment is that the low-quality data selected by traditional greedy search method can harm the behavior of the ODQA model. We can see that at the beginning of the training (i.e. round 1), our bootstrapping strategy only has one positive document (which is the ground-truth document in SQuAD or top-ranked document in Webquestions), but its performance is a little better. This means that the other positive documents selected by greedy search introduce much irrelevant information, which have a negative influence on the model. We conjecture that the QA system is very vulnerable to the context of the answer [25], so these false positive patterns around the answer make an adversarial attack on the model. Briefly, bootstrapping is more effective for data generation in ODQA than greedy search used by previous work.

6 CONCLUSION

In this paper, to tackle the large-scale open-domain question answering problem, we proposed a document gated reader, which combines the CNN based document selection and LSTM based answer generation in a single model. The model processes each document dependently with other documents which shows the advantage in document selection and final answer generation. To alleviate the influence of false positive data in distant supervised ODQA, we propose a bootstrapping based data generation scheme to generate high-quality data. Experimental results show the advantage of our model both on question answering precision and the quality of the generated data.

7 ACKNOWLEDGEMENTS

This research work is supported by the Natural Key R&D Program of China (No.2018YFC0830101), the National Science Foundation of China (No. 61533018, 61532011) and the independent research project of National Laboratory of Pattern Recognition.

REFERENCES

[1] Yoram Bachrach, Andrej Zukov-Gregoric, Sam Coope, Ed Tovell, Bogdan Maksak, Jose Rodriguez, Conan McMurtie, and Mahyar Bordbar. 2017. An Attention Mechanism for Neural Answer Selection Using a Combined Global and Local View. In *Tools with Artificial Intelligence (ICTAI)*. IEEE, 425–432.

[2] Ricardo Baeza-Yates, Berthier de Araújo Neto Ribeiro, et al. 2011. *Modern information retrieval.* New York: ACM Press; Harlow, England: Addison-Wesley,.

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. *CoRR* abs/1409.0473 (2014).

[4] Yoshua Bengio et al. 2009. Learning deep architectures for AI. *Foundations and trends® in Machine Learning* 2, 1 (2009), 1–127.

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

[6] Weijie Bian, Si Li, Zhao Yang, Guang Chen, and Zhiqing Lin. 2017. A Compare-Aggregate Model with Dynamic-Clip Attention for Answer Selection. In *CIKM*.

[7] 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, 1247–1250.

[8] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to Answer Open-Domain Questions. In *ACL*.

[9] Christopher Clark and Matthew Gardner. 2018. Simple and Effective Multi-Paragraph Reading Comprehension. *ACL* (2018).

[10] Hang Cui, Renxu Sun, Keya Li, Min-Yen Kan, and Tat-Seng Chua. 2005. Question answering passage retrieval using dependency relations. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 400–407.

[11] Ronan Cummins and Colm O'Riordan. 2006. Evolving local and global weighting schemes in information retrieval. *Information Retrieval* 9, 3 (2006), 311–330.

[12] Bhuwan Dhingra, Hanxiao Liu, Ruslan Salakhutdinov, and William W Cohen. 2017. A Comparative Study of Word Embeddings for Reading Comprehension. *arXiv preprint arXiv:1703.00993* (2017).

[13] Bhuwan Dhingra, Kathryn Mazaitis, and William W. Cohen. 2017. Quasar: Datasets for Question Answering by Search and Reading. *CoRR* abs/1707.03904 (2017).

[14] Matthew Dunn, Levent Sagun, Mike Higgins, Ugur Guney, Volkan Cirik, and Kyunghyun Cho. 2017. Searchqa: A new q&a dataset augmented with context from a search engine. *arXiv preprint arXiv:1704.05179* (2017).

[15] Minwei Feng, Bing Xiang, Michael R. Glass, Lidan Wang, and Bowen Zhou. 2015. Applying deep learning to answer selection: A study and an open task. *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)* (2015), 813–820.

[16] Yarin Gal and Zoubin Ghahramani. 2016. A Theoretically Grounded Application of Dropout in Recurrent Neural Networks. In *NIPS*.

[17] Clinton Gormley and Zachary Tong. 2015. *Elasticsearch: The Definitive Guide: A Distributed Real-Time Search and Analytics Engine.* " O'Reilly Media, Inc.".

[18] Edouard Grave, Tomas Mikolov, Armand Joulin, and Piotr Bojanowski. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL*. 3–7.

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

[20] Mark Andrew Greenwood. 2005. *Open-domain question answering*. Ph.D. Dissertation. University of Sheffield, UK.

[21] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *NIPS*. 1684–1692.

[22] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural computation* 9 8 (1997), 1735–80.

[23] Hsin-Yuan Huang, Chenguang Zhu, Yelong Shen, and Weizhu Chen. 2017. FusionNet: Fusing via Fully-Aware Attention with Application to Machine Comprehension. *arXiv preprint arXiv:1711.07341* (2017).

[24] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*. ACM, 2333–2338.

[25] Robin Jia and Percy Liang. 2017. Adversarial Examples for Evaluating Reading Comprehension Systems. In *EMNLP*.

[26] Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with GPUs. *arXiv preprint arXiv:1702.08734* (2017).

[27] Mandar S. Joshi, Eunsol Choi, Daniel S. Weld, and Luke S. Zettlemoyer. 2017. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. In *ACL*.

[28] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2014).

[29] Yankai Lin, Haozhe Ji, Zhiyuan Liu, and Maosong Sun. 2018. Denoising distantly supervised open-domain question answering. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 1736–1745.

[30] Jeffrey Ling and Alexander M. Rush. 2017. Coarse-to-Fine Attention Models for Document Summarization. In *NFiS@EMNLP*.

[31] Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: Contextualized word vectors. In *Advances in Neural Information Processing Systems*. 6294–6305.

[32] Alexander H. Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-Value Memory Networks for Directly Reading Documents. In *EMNLP*.

[33] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. *arXiv preprint arXiv:1611.09268* (2016).

[34] Adam Paszke, Sam Gross, Soumith Chintala, and Gregory Chanan. 2017. PyTorch.

[35] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global Vectors for Word Representation. In *EMNLP*.

[36] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365* (2018).

[37] Anthony Valiant Phillips. 1960. Artificial Intelligence Project—RLE and MIT Computation Center Memo 16—A Question-Answering Routine'. (1960).

[38] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In *EMNLP*.

[39] Matthew Richardson, Christopher JC Burges, and Erin Renshaw. 2013. MCTest: A Challenge Dataset for the Open-Domain Machine Comprehension of Text.. In *EMNLP*, Vol. 1. 2.

[40] Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional Attention Flow for Machine Comprehension. *CoRR* abs/1611.01603 (2016).

[41] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research* 15, 1 (2014), 1929–1958.

[42] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. 2000. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*. 1057–1063.

[43] Ming Tan, Cicero dos Santos, Bing Xiang, and Bowen Zhou. 2015. Lstm-based deep learning models for non-factoid answer selection. *arXiv preprint arXiv:1511.04108* (2015).

[44] Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2016. NewsQA: A Machine Comprehension Dataset. *arXiv preprint arXiv:1611.09830* (2016).

[45] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. 5998–6008.

[46] Suzan Verberne, H van Halteren, Stephan Raaijmakers, DL Theijssen, and LWJ Boves. 2009. Learning to Rank QA Data: Evaluating Machine Learning Techniques for Ranking Answers to Why-Questions. (2009).

[47] Bingning Wang, Kang Liu, and Jun Zhao. 2016. Inner attention based recurrent neural networks for answer selection. In *ACL*, Vol. 1. 1288–1297.

[48] Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerald Tesauro, Bowen Zhou, and Jing Jiang. 2017. R3: Reinforced Reader-Ranker for Open-Domain Question Answering. (2017).

[49] Shuohang Wang, Mo Yu, Jing Jiang, Wei Zhang, Xiaoxiao Guo, Shiyu Chang, Zhiguo Wang, Tim Klinger, Gerald Tesauro, and Murray Campbell. 2017. Evidence Aggregation for Answer Re-Ranking in Open-Domain Question Answering.

[50] Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. 2017. Gated self-matching networks for reading comprehension and question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 189–198.

[51] Zhiguo Wang, Haitao Mi, Wael Hamza, and Radu Florian. 2016. Multi-perspective context matching for machine comprehension. *arXiv preprint arXiv:1612.04211* (2016).

[52] Yusuke Watanabe, Bhuwan Dhingra, and Ruslan Salakhutdinov. 2017. Question Answering from Unstructured Text by Retrieval and Comprehension. *CoRR* abs/1703.08885 (2017).

[53] Yi Yang, Wen tau Yih, and Christopher Meek. 2015. WikiQA: A Challenge Dataset for Open-Domain Question Answering. In *EMNLP*.

[54] Roman Yangarber, Winston Lin, and Ralph Grishman. 2002. Unsupervised Learning of Generalized Names. In *COLING*.