PReGAN: Answer Oriented Passage Ranking with Weakly Supervised GAN

Pan Du
du@youark.com
Thomson Reuters Labs
Toronto, Canada

Jian-Yun Nie
University of Montreal
Montreal, Canada
nie@iro.umontreal.ca

Yutao Zhu
University of Montreal
Montreal, Canada
yutaozhu94@gmail.com

Hao Jiang
Huawei Poisson Lab.
Shenzhen, China
jianghao66@huawei.com

Lixin Zou
Tshinghua Univeristy
Beijing, China
zoulx15@mails.tsinghua.edu.cn

Xiaohui Yan
Huawei Poisson Lab.
Shenzhen, China
yanxiaohui2@huawei.com

ABSTRACT
Beyond topical relevance, passage ranking for open-domain factoid question answering also requires a passage to contain an answer (answerability). While a few recent studies have incorporated some reading capability into a ranker to account for answerability, the ranker is still hindered by the noisy nature of the training data typically available in this area, which considers any passage containing an answer entity as a positive sample. However, the answer entity in a passage is not necessarily mentioned in relation with the given question. To address the problem, we propose an approach called PReGAN for Passage Reranking based on Generative Adversarial Neural networks, which incorporates a discriminator on answerability, in addition to a discriminator on topical relevance. The goal is to force the generator to rank higher a passage that is topically relevant and contains an answer. Experiments on five public datasets show that PReGAN can better rank appropriate passages, which in turn, boosts the effectiveness of QA systems, and outperforms the existing approaches without using external data.

KEYWORDS
Passage Ranking, Question Answering, Generative Adversarial Network

1 INTRODUCTION
A common type of open-domain question answering (OpenQA) aims to find answers from a large collection of texts [5]. It operates generally in two steps: finding a limited number of candidate passages using a retrieval method, and performing machine reading through these texts to extract answers. The state-of-the-art machine reading methods have produced human-level performance [51] when reading the right passage. However, when reading the passages retrieved with the question, the performance drops dramatically [19, 27], showing the critical importance of retrieving good candidate passages. Many studies have been devoted to improving the topical relevance of retrieved candidate passages [19, 22, 25, 41], but few studies have investigated the problem of answerability, i.e., whether a retrieved passage may contain an answer. Both criteria - topical relevance and answerability, are critical in the context of OpenQA. In fact, a passage highly relevant to the question may not necessarily contain an answer, and a passage that contains an answer entity may not be relevant to the question. In both cases, the reader may be misled by these passages to select a wrong answer, as the reader highly relies on the few passages provided to it. It is thus important that the selected passages for the reader should both be relevant and contain an answer.

In this paper, we propose an approach to refine the list of candidate passages according to both criteria through an answer-oriented passage (re-)ranking.

Training an effective reranking process is not trivial due to the problem of noisy training data: in many cases (datasets), we only know the correct answer to a question, but not the correct passages from which it should be extracted. For example, as illustrated in Figure 1 (a sample from the Quasar-T dataset), for the question about the inventor of the helicopter, we only know the right answer “Igor”. We can see that P2, P3 and P4 all contain the answer entity. However, P3 is obviously not a relevant passage to the question. Nevertheless, such a passage has been commonly used as positive training data in previous studies. Another typical example is P1, which is highly relevant to the question and contains entities (two persons) that look like the answer. More tricky cases are P2 and P4: Both passages are topically relevant and contain the answer entity. However, P4 provides support for the answer, while P2 does not. Using all the passages containing the answer as positive examples will obviously confuse the subsequent reader. The existing methods for passage ranking do not perform well on these cases. For example, a recent state-of-the-art QA model DPR [19] ranks P2 before P4. Our goal is to rerank the candidates to promote P4, by enhancing the ranker with some capability of machine reading to detect answerability.

The problem of answer-oriented passage ranking has been addressed in a few recent studies. Two typical approaches have been proposed: incorporating a strong neural mechanism in the first
Q: What Russian emigre to the U.S. is credited with inventing the helicopter?
A: Igor

| R | A | Passage | DPR | PreGAN |
|---|---|---------|-----|--------|
| P1 | 1 | Wright brothers: Orville and Wilbur Wright, were the two Americans who are credited with inventing and building the world’s first successful airplane. | 71.7 | 242.9 |
| P2 | 1 | The fellow Russian emigre, Igor Ivanovich Sikorsky, was an American aviation pioneer in both helicopters and fixed-wing aircraft, who was credited with many other accomplishments... | 75.6 | 286.8 |
| P3 | 0 | Igor Ivanovich Sikorsky was an orthodox christian. | 57.7 | 178.1 |
| P4 | 1 | His paternal grandfather, Leo Shoumatoff, was the business manager of Igor Sikorsky’s aircraft company, where Igor developed the first helicopter and the first passenger airplane. | 74.4 | 292.3 |

Figure 1: Examples of noisy passages for answering a question from Quasar-T. R and A indicate if the passage is topically relevant and if it contains the answer, respectively. Underlined words are related topic words, and green words are the answer text. Red words are correct but unsupported answers in the passage. We also show the ranking scores computed by DPR and our PreGAN model.

(3) Experiments on five public OpenQA datasets demonstrate the utility of combining answerability with relevance in passage ranking, which in turn, boosts the quality of extracted answers.

2 RELATED WORK

We only review the most relevant studies to our work in this section.

Passage ranking is crucial for QA to select better and fewer passages for the reader. The naive approach used in many studies relies on an IR model for it [1, 5, 28, 29, 31, 42, 44, 45]. However, an IR model will only rank the passages according to their topical relevance to the question without taking into account the answerability. For example, previous studies [25] show that a better passage ranking performance on Hits@N may not necessarily lead to a better answer extraction for the whole QA system. More recent studies have applied dense vector representations to rank passages, based on which a more complex ranking model can naturally incorporate query expansion [8, 13, 16, 19, 20, 32]. However, the neural retrieval process still focuses on topical relevance only.

Some recent studies have incorporated an intermediate ranking step on the search results, which considers the feedback from the reader, or relies on some reading capability incorporated into the ranker. Typical examples are R^2 [41], DSQA [25], and Retro-Reader [51]. In these approaches, any passage containing the answer text (entity) will be used as a positive sample for ranker training, i.e., the ranker is not explicitly trained to denoise the training data, and it relies on another component (usually a computationally expensive external reader) to provide indications. The GAN-based approach proposed in this paper specifically addresses the problem of noisy data inside the ranker, so as to simultaneously improve the efficiency and effectiveness of the reader by feeding it with a smaller number of passages with better answerability through ranking.

Generative adversarial networks [14] have been widely used, including in IR [40, 50, 52] and recommendation [3, 4, 7, 12, 49]. The GAN framework aims at generating the positive samples while making large differences with false positives. In so doing, a stronger generation model can be obtained. Recently, supervised GAN frameworks have also been explored in of question answering tasks [18, 23, 24, 26, 33–35, 38, 47], such as community QA, visual QA [36], and multi-choice QA, etc. GANs are utilized to tackle either the class imbalance problem of data [46] or the question clarification problem [37] in QA. However, none of them explored the utilization of GAN for answer-oriented passage ranking. Our study shows that GAN can be effectively used for this purpose.

The traditional GAN framework makes a minimax competition between one generator and one discriminator. For passage ranking, we incorporate two discriminators for relevance and answerability. Multiple discriminators have been used in some previous GAN frameworks. For example, Nguyen et al. [30] have used a similar approach to tackle the mode collapse problem in GAN. Some others applied similar structure to improve image quality [11] or for image-to-image translation [48]. Our study is the first attempt using such an extended GAN to cope with multiple criteria in passage ranking for open-domain question answering.

There are also several studies that exploit external resources and large pre-trained models for QA [15, 22, 43]. The advantages of external resources and pre-trained language models are obvious:
the external resources may contain passages that contain the right answer, or the pre-trained language model may already cover the question, i.e., it is able to generate the right answer for it. In these cases, it is difficult to determine where the improvements observed on QA quality come from: from the external resources/pre-trained language model, or the passage ranking strategy using the same source of information. To avoid this confusing situation, we do not include enhancement by external resources or pre-trained language models in this study, although our approach is orthogonal to them and can be combined with them (this will be investigated later).

3 METHODOLOGY

Our ranking model aims to re-rank the list of passages returned by some efficient retrieval component, such as BM25, so that the passages that can answer the given question will be highly ranked. We can use any reader to read the proposed passages to locate the answers within them. In the following, we only detail the process of reranking. Notice that our ranking model should be efficient so as not to hinder the efficiency of the whole QA process.

3.1 Ranking Framework

The general architecture of the PReGAN model is illustrated in Figure 2. It is a weakly supervised generative adversarial neural network containing three parts: a rank discriminator (for relevance), an answer discriminator (for answerability), and an answer-oriented rank generator.

The rank discriminator distinguishes the generated rank distributions from the ground-truth rank distribution. The answer discriminator tells if a generated sample contains the answer to the given question. The rank generator reads the passages under the guidance of the two discriminators and learns to generate the answer-oriented distribution on those passages for the given question. The discriminators therefore guide the reading-based generator with a REINFORCE process from two different perspectives.

At the end of the training, it is expected that passages that are relevant to the question and contain the answer text can be ranked highly. In other words, optimizing the three objectives modeled in the GAN framework will help discard noise and promote good passages that can truly answer the question. The utilization of the GAN framework as an effective denoising means [39] is not new, but we show for the first time that it is also particularly adapted to the noisy training environment in OpenQA.

3.2 Overall Objective

Taking the answer-oriented ranking as a distribution over passages for a given question, the objective of the generator is to learn this distribution through the minimax game. The rank generator would try to select passages that the answer can be reasoned (by the reader) for the given question. The rank discriminator would try to draw a clear distinction between the ground-truth answer passages and the selected ones made by the generator. Since the positive training passages are noisy, the rank discriminator alone may be misleading for the machine reading-based generator. However, if the answer can be extracted from the passage and the answer discriminator confirms the correctness of the answer, the passage selected by the generator should be a good one. This principle can be formulated as the following overall objective function of the minimax game:

\[
J = \min_{\theta} \max_{\phi, \xi} \frac{1}{N} \sum_{n=1}^{N} \left( \mathbb{E}_{d \sim P_{true}} \left[ \log D_{\phi} (d | q_n) \right] + \lambda_1 \cdot \mathbb{E}_{d \sim P_{true}} \left[ \log \frac{D_{\phi} (d | q_n)}{D_{\phi} (d | q_n)} \right] + \lambda_2 \cdot \mathbb{E}_{d \sim P_{true}} \left[ \log \frac{P_{true} (d | q_n)}{P_\theta (d | q_n)} \right] \right)
\]

where the generative ranking model with parameter \( \theta \) is written as \( P_\theta (d | q_n) \), and the rank discriminator \( D_{\phi} \) parameterized by \( \phi \) estimates the probability that a passage \( d \) can answer the question \( q \). The answer discriminator \( D_{\phi} \) with parameter \( \xi \) works as a regularizer of the generator emphasizing passages in which the answer text appears, no matter it can answer the question or not. The last item in the objective is another commonly used regularizer [25, 41] pushing the overall ranking score distribution produced by the generator towards the ground-truth distribution by minimizing KL-divergence between them.

3.3 Rank Discriminator

As shown in Figure 2, the rank discriminator takes as input the passages generated (selected) by the current optimal generator \( P_\theta \) together with the observed passages with answers, and predicts the source of the passages. Denoting \( D_{\phi} (d | q_n) = \sigma (f_\phi (d, q_n)) \), which is the sigmoid function of the rank discriminator score \( f_\phi \), the loss function of rank discriminator is then represented as:

\[
L_{D_{\phi}} = - \frac{1}{N} \sum_{n=1}^{N} \left( \mathbb{E}_{d \sim P_{true}} \left[ \log (\sigma (f_\phi (d, q_n))) \right] + \mathbb{E}_{d \sim P_{true}} \left[ \log (1 - \sigma (f_\phi (d, q_n))) \right] \right)
\]
3.4 Answer Discriminator

The answer discriminator is designed to cooperate with the generator to mitigate the effects of noise in the training data. The answer discriminator only tells if the answer text appears in a passage for the given question instead of trying to reason out a result. The generator, on the other hand, makes decisions based on a machine reading comprehension process and cares less about the existence of the answer texts in the passages. The idea is that the answer discriminator and generator make judgements from different perspectives, then cross check the judgements and guide the training of the generator towards passages on which they have mutual agreement. The training of the answer discriminator, however, is independent from the generator, and can be trained directly using passage and answer pairs for each question with a binary cross entropy loss as follows:

\[
\mathcal{L}_{D^a} = -\sum_{n=1}^{N} \left( \sum_{d \in A^a} \log \sigma(f_\phi^a((d, q_n))) + \sum_{d \in A^a} \log (1 - \sigma(f_\phi^a(d, q_n))) \right),
\]

where, similar to the rank discriminator, the score function \( f_\phi^a \) is also implemented as a bi-encoder attention network. Passages for question \( q_n \) are divided into \( A^a \) and \( A^a \) depending on whether a passage contains the answer text or not.

3.5 Generator

The generator \( p_\theta \) aims to minimize the objective, through which it tries to fool the rank discriminator by generating samples faking the ground-truth answer distribution. By keeping the two discriminators \( f_\phi^a \) and \( f_\phi^r \) fixed after their maximization, the generator’s loss function is as follows:

\[
\mathcal{L}_{p_\theta} = \sum_{n=1}^{N} \left( \mathbb{E}_{d \sim p_\theta} [\log (1 - \sigma(f_\phi^a(d, q_n)))] - \lambda_1 \mathbb{E}_{d \sim p_\theta} [\log \sigma(f_\phi^r(d, q_n))] - \lambda_2 \mathbb{E}_{d \sim p_{true}} [\log p_\theta(d|q_n, a)] \right),
\]

where minimizing the first item means to guide the generator \( p_\theta \) toward generating passages which the rank discriminator considers as likely ground-truth passages. Minimizing the second item will drive the generator away from generating passages that the answer discriminator dislikes. The third item imposes a regularization on the overall answer distribution produced by the generator, forcing it to stay close to the ground-truth answer distribution. \( \lambda_1 \) and \( \lambda_2 \) are hyper-parameters controlling the impacts of the answer discriminator and the distribution regularizer on the generator.

The generator is trained under weak supervision using the noisy training data. It predicts the ranking score of each passage using a simple machine reading comprehension (MRC) component. Specifically, the MRC component takes a question and the passage candidates as input, and produces a joint distribution of the start position and end position of the answer in a passage. Then it takes the maximum of the joint probability as the predicted ranking score of the passage with respect to the question. The ranking scores are then normalized into a probabilistic distribution indicating how likely each passage can answer the given question. Similar to \( f_\phi^a \) and \( f_\phi^r \), the ranking scores are also generated by scoring networks.

The details of the scoring networks are given in the next section.

Our scoring networks are inspired by [25]. We opt for it because of its high efficiency, which is required for our ranking. Other scoring schemes, such as transformer-based approaches, could also be adopted in our framework in the future.

From the loss function of the generator, its gradient can be derived as follows (We include the details about the derivation of Equation (4) in the Appendix A.1 for interested readers):

\[
\nabla_\theta \mathcal{L}_{G}(q_n) = \nabla_\theta \mathbb{E}_{d \sim p_\theta} \left[ \log (1 + \exp(f_\phi^r(d, q_n))) \right] + \lambda_1 \nabla_\theta \mathbb{E}_{d \sim p_\theta} \left[ \log (1 + \exp(f_\phi^a(d, q_n))) \right] - \lambda_2 \nabla_\theta \mathbb{E}_{d \sim p_{true}} [\log p_\theta(d|q_n, a)]
\]

\[
= \frac{1}{K} \sum_{k=1}^{K} \nabla_\theta \log p_\theta(d_k|q_n, a) \left( \log (1 + \exp(f_\phi^r(d_k, q_n))) + \lambda_1 \log (1 + \exp(f_\phi^a(d_k, q_n))) \right) - \lambda_2 \sum_{m=1}^{M} \frac{p_{true}(d_m|q_n, a)}{p_\theta(d_m|q_n, a)},
\]

where \( p_{true} = 1/|d^a| \) is the total number of passages containing correct answers if the passage contains a correct answer, otherwise 0. This completes the approximation approach for generator optimization with policy gradient based reinforcement learning. The term \( \log(1 + \exp(f_\phi^a(q_n, a))) + \lambda_1 \log (1 + \exp(f_\phi^r(q_n, a))) \) acts as the balanced rewards from rank discriminator and answer discriminator for the generator \( p_\theta(d|q_n, a) \) to select passage \( d \) for a given question \( q_n \).

3.6 Scoring Networks

In this section, we will introduce the scoring functions we used in details. As can be seen from the Figure 2 and the Equation (1), the three scoring functions lie in the kernel of our proposed GAN framework.

Formally, given a question \( q = (q^1, q^2, \ldots, q^{|q|}) \) and \( m \) passages returned by some retrieval component, which are defined as \( D = (d_1, d_2, \ldots, d_m) \), where \( d_i = (d_i^1, d_i^2, \ldots, d_i^{|d_i|}) \) is the \( i \)-th retrieved passage composed of a sequence of words, the answer-oriented passage ranking aims to assign a score \( P(d_i|q) \) to passage \( d_i \) measuring the probability that it can answer the question \( q \).

For \( f_\phi^a \) in rank discriminator \( D_r \), we use a bi-encoder framework to build the scoring function \( f_\phi^a(d, q_n) \). Each passage \( d_i \) and the question \( q \) are encoded with an RNN network:

\[
< \hat{d}_i^1, \hat{d}_i^2, \ldots, \hat{d}_i^{|d_i|} > = \text{RNN}(< d_i^1, d_i^2, \ldots, d_i^{|d_i|} >), \quad < \hat{q}^1, \hat{q}^2, \ldots, \hat{q}^{|q|} > = \text{RNN}(< q^1, q^2, \ldots, q^{|q|} >),
\]

where \( \hat{d}_i^j \) and \( \hat{q}^j \) are the hidden representations of words in passage \( d_i \) and question \( q \) encoding its context information through RNN. We use a single-layer bidirectional LSTM (biLSTM) as our RNN unit. Following [25], the final representation of the question is obtained through a self-attention operation \( q = \sum_{j=1}^{p} a^j q^j \), where \( a^j = \text{softmax}(w^j q^j) \) is the importance of each word in the question and \( w \) is a learned weight vector. Then the score of each passage is
obtained via a max-pooling layer and a softmax layer:

\[ f_\theta(d_i, q) = p(d_i|q) = \text{softmax}(\max_j (d_i^j W q)), \]

where \( W \) is a weight matrix to be learned.

For \( f_\phi \) in answer discriminator \( D_a \), even though it serves a different purpose with a different objective function from the rank discriminator \( D_r \), the network framework is similar, except that the last layer is a sigmoid function instead of softmax function.

Finally, for \( f_\xi \) in rank generator \( G \), similar to \( f_\phi \), we encode the passage \( d_i \) in a sequence of hidden vectors, and apply a self-attention layer to attend to the question vectors to get a question-embedding. Different from the discriminators, we ask the generator to score a passage through predicting the probability of the start and end positions of an answer span in the passage. The probability of a position \( j \) as the start position of the answer span in the passage \( d_i \) is calculated by \( p_s^j = \text{softmax}(d_i^j W q) \) with parameter \( W_a \), and \( p_e^j = \text{softmax}(d_i^j W q) \) for the end position using \( W_e \). The score of passage \( d_i \) is then obtained as the maximum of the joint probability of the start and end positions:

\[ f_\xi(d_i, q) = p_\xi(a(q, d_i)) = \max_{j,k} p_s^j(a(q, d_i)) p_e^k(a(q, d_i)) \]

Our scoring functions are inspired by [25]. We opt for it because of its high efficiency, which is required for our ranking. Other scoring schemes, such as transformer-based approaches, could also be adopted in our framework in the future.

3.7 Ranking Algorithm

With all the components in place, the overall ranking algorithm is summarized in Algorithm 1. Before the adversarial training, we initialize the three component with the pre-trained weights.

The answer discriminator, rank discriminator, and generator are pre-trained using the noisy training data.

From the ranked passages, the top-K (K=50 in our experiments) passages for each question and apply our model to them [25, 27].

We use a BERT-based reader architecture in this paper, which follows [19]. Let \( d_i \in \mathbb{R}^{L \times h} \) be a BERT representation of the \( i \)-th passage, where \( L \) is the maximum length of the passage and \( h \) is the hidden dimension. The probability of an answer span \((s, e)\) in passage \( d_i \) is defined as:

\[
P(s, e, i) = P(d_i) \cdot P(s|d_i) \cdot P(e|d_i),
\]

\[
P(s|d_i) = \text{softmax}(d_i^{start}),
\]

\[
P(e|d_i) = \text{softmax}(d_i^{end}),
\]

\[
P(d_i) = \text{softmax}(\tilde{D}^T w_{doc}),
\]

where \( \tilde{D} = [d_1^{[CLS]}, \ldots, d_k^{[CLS]}] \in \mathbb{R}^{h \times k} \) where \( d_i^{[CLS]} \) is the BERT-based sentence representation (with 12 transformer layers and hidden size of 768), and \( w_{start}, w_{end}, w_{doc} \in \mathbb{R}^h \) are learnable parameters. Notice that the final answer’s probability \( P(s, e, i) \) also uses the passage ranking score \( P(d_i) \), so that the reader will trust more the answers from highly ranked passages.

4 EXPERIMENTS

4.1 Datasets and Evaluation Metrics

We evaluate our ranking model on five commonly-used datasets for OpenQA tasks:

- **Quasar-T** [9] contains 43,012 trivia questions, each with 100 passages retrieved from ClueWeb09 data source using LUCENE.

- **SearchQA** [10] is a large-scale OpenQA dataset with question-answer pairs crawled from J! Archive and passages retrieved by Google. It contains 140k question-answer pairs. On average, each question has 49.6 passages.

- **TriviaQA** [17] includes 95k question answer pairs authored by trivia enthusiasts and independently gathered evidence passages. Each question has 100 webpages retrieved by Bing Search API. We only use the first 50 for training the ranker and the reader.

- **Curated TREC** [2] is based on the benchmark from the TREC QA tasks, which contains 2,180 questions extracted from datasets of TREC 1999, 2000, 2001, and 2002.

- **Natural Questions** [21] consists of real anonymized, aggregated questions issued to the Google search engine. The answers are spans in Wikipedia articles identified by annotators. Each question is paired with up to five reference answers.

For Quasar-T, SearchQA, and TriviaQA, we use the same pre-processed paragraphs provided by [25]. For Curated TREC and Natural Questions, following existing work, we determine a subset of 50 passages for each question and apply our model to them [25, 27].

This is done due to our limited computation resources – training our model on a larger set of long passages (webpages) would require more memory than we have.

---

1. [https://github.com/facebookresearch/DPR](https://github.com/facebookresearch/DPR)
2. We assume that the reader is familiar with the BERT training process, or could find more details in [19].
We tune our model on the development set. Restricted by the computation resources available – one Nvidia GTX Titan X GPU (12G VRAM), only limited parameter sets could be explored. The hidden size of the RNN used in the ranker is 128, the number of RNN layers for the passage and question are both set to 1. The maximum passage length is set according to the passage length distributions – 150 for quasar-T, SearchQA, TriviaQA, and Natural Questions, and 350 for Curated TREC. The batch size varies among [2, 4, 8, 16] across datasets depending on the maximum passage length in order to fit the model into the GPU RAM. We only use top-50 passages retrieved by BM25 to train the ranker and the reader. The hyper-parameters $\lambda_1$ and $\lambda_2$ are set to 0.25 and 1. We follow [25] for the setting of other parameters. The complete optimal parameter settings and code will be made public in our Github repository later.

4.4 Ranking Results

We first test the ability of our PReGAN ranker to better rank passages containing the answers. The baseline models have been used on different datasets. To compare with them directly, we run two sets of experiments: one on Quasar-T and SearchQA on which DSQA has been tested, and another on Curated TREC and TriviaQA on which DPR has been tested. In addition, we also compare DPR with our method on the Natural Question subset.

At the top of Table 2, we report the ranking results by BM25, DSQA, and PReGAN. It shows that PReGAN outperforms both baselines by a large margin. Recall that the main difference between DSQA’s selector and PReGAN lies in the denoising based on GAN. The improvements hence directly reflect the usefulness of the GAN mechanism.

On Curated TREC and TriviaQA, as the candidate passages are not provided in the datasets, but retrieved from Wikipedia, we consider two cases – retrieving 50 and 100 candidates with BM25 and submitting them to the ranker. Notice that DPR directly retrieves passages from the whole Wikipedia dump, and thus is not limited by the 50 or 100 candidate passages. Nevertheless, we can see that our ranker can produce similar ranking results (Hits@20) to DPR despite the more limited candidates. On the Natural Question subset, we run DPR retriever and our ranker on the top-50 results from BM25. We see that PReGAN can better rank passages in top positions than DPR. However, at lower rank positions (from HTS@20) to DPR performs better. This can be attributed to the more sophisticated retrieval model in DPR. In the future, the score networks used in our current model could be replaced by more sophisticated ones as in DPR. Another factor that explains the better ranking for DPR when \( N \geq 20 \) is that HTS@N is only an approximation of the ranking quality. This measure should be considered together with the EM measure of answers.

The most important observation to make is the improvements of PReGAN over BM25. They reflect the capability of our ranker to favor the passages that may contain an answer. We expect that this will benefit the reader, as we will show.

4.5 Final Answer Results

We adopt a BERT-based reader structure as in [19] to extract answers from the selected passages. We test the QA answer for each question produced by the reader with the top-50 passages (together with their ranking scores). The results are shown in Table 3.

We can observe that on all the datasets, PReGAN (combined with DPR reader) leads to consistently better answer results than the

| Dataset       | Train | Dev. | Test | #Passages/Question |
|---------------|-------|------|------|-------------------|
| Quasar-T      | 37,012| 3,000| 3,000| 100               |
| Search QA     | 99,811| 13,893| 27,247| ~49.6            |
| Trivia QA     | 87,291| 11,274| 10,790| 100               |
| Curated TREC  | 1,353 | 133  | 694  | Wiki (50)         |
| Natural Question | 79,168| 8,757| 3,610| Wiki (50)         |
Table 2: Ranker performance compared with DSQA on datasets Quasar-T and SearchQA, and compared with DPR on Curated TREC, TriviaQA, and NQ-Sub. The best results are in bold. Results marked with $\diamond$ are cited from [25], which does not provide Hits@20 and Hits@50. Results marked with $\triangle$ are cited from [19]. Results marked with $\heartsuit$ are obtained by running DPR against top-50 passages returned by BM25.

|                | Hits@1 | Hits@3 | Hits@5 | Hits@20 | Hits@50 |
|----------------|--------|--------|--------|---------|---------|
| BM25$^\diamond$ | 6.3    | 10.9   | 15.2   | -       | -       |
| DSQA$^\diamond$ | 27.7   | 36.8   | 42.6   | -       | -       |
| PreGAN         | 35.2   | 52.0   | 59.5   | 72.3    | 74.8    |

|                | Hits@1 | Hits@3 | Hits@5 | Hits@20 | Hits@50 |
|----------------|--------|--------|--------|---------|---------|
| BM25$^\diamond$ | 13.7   | 24.1   | 32.7   | -       | -       |
| DSQA$^\diamond$ | 59.9   | 69.8   | 75.5   | -       | -       |
| PreGAN         | 63.9   | 83.0   | 88.8   | 97.5    | 99.8    |

|                | Hits@1 | Hits@3 | Hits@5 | Hits@20 | Hits@50 |
|----------------|--------|--------|--------|---------|---------|
| BM25           | 23.9   | 45.1   | 54.3   | 73.7    | 82.8    |
| DPR$^*\diamond$ | -      | -      | -      | 79.8    | -       |
| PreGAN (58)    | 30.3   | 51.2   | 60.8   | 78.6    | 82.8    |
| PreGAN (180)   | 33.3   | 54.6   | 64.1   | 81.0    | 84.6    |

|                | Hits@1 | Hits@3 | Hits@5 | Hits@20 | Hits@50 |
|----------------|--------|--------|--------|---------|---------|
| BM25           | 28.1   | 46.7   | 56.6   | 75.6    | 81.3    |
| DPR$^*\diamond$ | -      | -      | -      | 79.4    | -       |
| PreGAN (58)    | 48.9   | 63.8   | 70.0   | 79.2    | 81.5    |
| PreGAN (180)   | 48.5   | 64.1   | 70.0   | 79.2    | 81.7    |

|                | Hits@1 | Hits@3 | Hits@5 | Hits@20 | Hits@50 |
|----------------|--------|--------|--------|---------|---------|
| BM25           | 17.6   | 28.9   | 35.5   | 53.1    | 64.3    |
| DPR (a11)      | 39.3   | 53.6   | 58.8   | 68.9    | 73.7    |
| DPR (58)$^*\diamond$ | 23.6 | 34.8 | 42.4 | 59.7 | 64.3 |
| PreGAN         | 24.0   | 36.7   | 43.3   | 58.2    | 64.3    |

Table 3: Answer results with exact matching (EM) rate. The best results are in bold. Numbers with $^*$ for DSQA and $R^3$ are cited from [25], those for DPR are from [19]. For the Natural Question subset, our model is based on the top-50 passages retrieved by BM25, while DPR uses its own retriever to retrieve top-50 passages.

|                | Quasar-T | SearchQA | Curated TREC | TriviaQA | NQ-Sub |
|----------------|----------|----------|--------------|----------|--------|
| BM25+BERT      | 41.6     | 57.9     | 21.3$^*$     | 47.1$^*$ | 26.7   |
| $R^3$          | 35.3$^*$ | 49.0$^*$ | 28.4$^*$     | 47.3$^*$ | -      |
| DSQA           | 42.2$^*$ | 58.8$^*$ | 29.1$^*$     | 48.7$^*$ | -      |
| DPR            | -        | -        | 28.0$^*$     | 57.0$^*$ | 27.4   |
| PreGAN         | 45.5     | 61.2     | 29.3         | 60.7     | 29.5   |

baselines. As BM25 ranking results are submitted to the same reader as PreGAN, the differences between them are directly attributed to passage ranking. This comparison clearly shows the usefulness of adding a ranker to rerank the BM25 retrieval results.

The comparison with $R^3$ shows the utility of adding discriminators. Recall that $R^3$ also uses the rewards from the reader to rerank the retrieval results, but without contrasting between different passages. We can see that by adding discriminators, PreGAN is forced to better distinguish between truly good passages from noisy ones.

In this experiment, DSQA uses its own selector to rerank the BM25 candidates and its reader to find the answer, and DPR also uses both its retriever to retrieve candidates and its reader. This setting may play in favor of these models because the retriever, selector, and reader have been optimized together. More particularly, DPR is allowed to search for top-ranked candidate passages directly from the document collections in Curated TREC (top-100) and Natural Questions (top-50), instead of using the top-50 retrieved with BM25. The quality of these passages is better than that of BM25. Despite this, compared to DSQA and DPR, PreGAN can still produce better results. This result demonstrates that our QA approach can be competitive against the state-of-the-art approaches that have much more parameters. Taking the experiments on passage ranking and answer identification together, we can see that a better passage ranking generally leads to better answers.

4.6 Further Analysis

In this section, we further analyze the impact of the components of PreGAN and the time efficiency.

4.6.1 Influence of the Number of Retrieved Passages. We test with different numbers of initial retrieval results on Curated TREC, on which we can retrieve the desired number of passages. In the experiment, our ranker is asked to select the top-50 passages from the initial retrieval list of variable sizes.

Figure 3 shows how the HITS measures vary depending on the size of the retrieved passages (from 50 to 300). The general observation is that by increasing the size of initial retrieval results from 50 to 100, all HITS measures are improved. However, when we use a larger set of retrieval results, HITS at small $N$ are hurt, while HITS at larger $N$ keep being improved. This observation tends to suggest that the initial retrieval results could be increased to a reasonable size. However, a too large set of initial retrieval results may lead to a higher likelihood to include noise passages, which may make it more difficult for the ranker to determine the right passages. The right choice of the size of initial passages retrieval is an interesting question that we will investigate in the future.
A good ranker should be able to select a small number of passages on top. We attribute this capability to the GAN mechanism used, which helps to discriminate between good and bad passages.

Figure 4: Reader performance with different numbers of ranked passages on Quasar-T (top) and SearchQA (bottom).

4.6.2 Impact of the Number of Passages to be Read. We select top-K passages from the ranker, and submit them to the reader to find an answer. We want to test the impact of K on the answer results.

Figures 4 shows the Exact Match results of our system compared with the baseline methods DSQA and BM25+DSQA on the datasets Quasar-T and SearchQA. In BM25+DSQA, the passages are ranked by BM25 scores, and we use the reader of DSQA to find the answer.

We can observe that submitting more passages to the reader will generally lead to better answers. However, this is at the cost of higher time complexity due to the complex machine reading process. A good ranker should be able to select a small number of passages for the reader, without penalizing the final results. This is what we observe from PreGAN: by selecting top-20 and top-10 passages for the reader, our model can already obtain the best answer results. This further confirms that PreGAN is able to rank good passages on top. We attribute this capability to the GAN mechanism used, which helps to discriminate between good and bad passages.

Note that BM25+DSQA and DSQA use the same reader and they differ only in an additional selector in DSQA. PreGAN uses a different reader. To see the contribution of the ranker on a fair ground, we will run another experiment using the same reader.

Table 4: EM performance of Bi-LSTM-based reader on our ranking results on Quasar-T.

| Number of Passages (k) | 1   | 3   | 5   | 10  |
|------------------------|-----|-----|-----|-----|
| DSQA                   | 22.6| 28.6| 32.0| 36.5|
| PreGAN+DSQA            | 25.9| 31.2| 34.7| 38.8|
| PreGAN (DPR reader)    | 28.0| 37.1| 40.7| 43.7|

4.6.3 Effectiveness of Ranker. In this experiment, we test the respective contributions of the ranker and the reader. We compare our method with DSQA. To test the effectiveness of the ranker, we replace the ranker in DSQA by that of PreGAN, but still use the same DSQA reader to find the answer (PreGAN+DSQA). Table 4 shows the results.

We can see that when the DSQA ranker is replaced by ours in PreGAN+DSQA, the EM measures are improved (over DSQA). This is a clear demonstration that our ranker PreGAN is more effective than that of DSQA. The comparison between our method and PreGAN+DSQA shows the impact of a better reader – both our method and PreGAN+DSQA use the same list of passages, but PreGAN is followed by a BERT-based reader (similar to DPR), while DSQA uses a biLSTM-based reader. This shows that the quality of both the ranker and the reader contribute to the global effectiveness of QA, and they are complementary.

4.6.4 Time Efficiency. As an intermediate step, a ranker should be efficient. To show the time cost of different steps, we report in Table 5 the time required for different steps on NQ-sub, on which we can perform all the retrieval-ranking-reading processes. The experiment is run on a server equipped with one Nvidia GTX Titan X GPU (12GB VRAM), one Intel Core i7@3.7GHz CPU with 12 cores, and 32G RAM size.

As we can see, the time cost of our ranker is very small compared to the retrieval and reading steps. The reader is the most expensive component. The more we submit passages to the reader, the more time it takes to find the answers. Therefore, it is advantageous to submit only a few passages to the reader without alternating the quality of final answers, which our ranker is able to do (see Figure 4).

4.6.5 Effects of Discriminators. We used two discriminators together for different criteria. We show in Table 6 the performance of the ranker without the answer discriminator component (w/o AD). We can see that the HITS measures generally decrease when the answer discriminator is removed. This confirms the utility of this additional discriminator in the GAN framework.
We use the DPR Table 6: Performance of ranker without answer discrimina-

tor (AD) on Quasar-T.

| Hits@1 | Hits@3 | Hits@5 | Hits@20 | Hits@50 |
|--------|--------|--------|---------|---------|
| DPR    | 35.2   | 52.0   | 59.5    | 72.3    | 74.8    |
| w/o AD | 33.0   | 50.1   | 58.1    | 72.0    | 74.8    |

4.6.6  Case Study. Let us analyze the example we gave at the be-

ginning of the paper (Figure 1) to see the impact of our ranker on

some concrete example. The example is from the Quasar-T dataset.

We use the DPR retriever and our ranker (on top of BM25 results) to

score the candidate passages. In this example, the passage P3 can

be easily ranked low by both approaches due to its low relevance

to the question. The passage P4 is the best passage that contains

the right answer to the question, and the passage provides explicit

support for the answer. P2 is topically relevant and also contains

the answer, but the passage does not provide support for the an-

swer. Between the passages P2 and P4, DPR prefers P2 because

it contains more question-related words, i.e., more topically rele-

vant. Topical relevance is indeed the major criterion used in DPR’s

retriever, as can be reflected in the ranking of the passages. On the

other hand, our ranker can successfully rank the best passage P4

on top, due to its integration of some reading capability. When a

reader goes through the two passages, it is expected that the reader

will find the answer in P4 with a higher probability than in P2. This

example makes the expected impact of our ranker more concrete.

The effect is exactly what we desire.

One could argue that the ranking difference of the desired pas-

sage is not so important as long as the passage is submitted to the

reader, and we could simply rely on the capability of the reader to

find the right answer. This is not true because: 1) ranking a good

passage at a lower rank will make it more likely to be cut off (not

submitted to the reader), especially when the reader can only read

a small number of passages; 2) the final answer from the reader also

takes into account the ranking score – the more the reader is con-

fident about the passage, the more the reader will have confidence

about the answer in the passage. So the correct ranking score will

directly impact the final answer of the reader.

5 CONCLUSION

In this paper, we addressed the problem of passage (re-)ranking

for open-domain QA. The goal is to create a better and reduced

ranking list for the reader. Previous attempts on passage ranking

have focused on improving the relevance of the passages [19], us-

ing explicit interaction between the ranker and the reader [41] or

incorporating some reading capability into the ranker [25]. How-

ever, given the noisy nature of the training data available, these

approaches are hindered by the noise. In this paper, we proposed a

model based on a customized GAN framework to support answer-

oriented passage ranking. We utilized two discriminators – rank-

ning discriminator and answer discriminator, to guide the reading-based

generator to assign high probabilities to passages that are relevant

and contain the answer. Experiments on five datasets demonstrated

the effectiveness of our proposed method.

The work can be further improved on several aspects. First, due

to the computation resources, we were unable to do experiments

on large datasets (such as the full set of Natural Questions). We

are looking for more powerful computation servers to run larger

experiments. Second, our design choice has been inspired by the

previous studies (namely, DQA), which uses a simpler architecture

(biLSTM) than the one commonly used now (based on BERT). More

recent approaches are typically based on BERT. We expect that

similar improvements will obtain from the idea of incorporating

both relevance and answerability into a ranker, even though another

scoring network is used. This will be tested in the future.

A APPENDIX

A.1 Derivation of the Reward Function

The detailed derivation process of the reward functions and regu-

larizers are as follows:

\[
\nabla_\theta L^G(q_n) = \nabla_\theta \mathbb{E}_{d-p_\theta} [\log (1 + \exp(f_\phi(d, q_n))] + \nabla_\theta \mathbb{E}_{d-p_\theta} [\log (1 + \exp(f_\phi(d, q_n))] - \nabla_\theta \mathbb{E}_{d-p_{true}} [\log p_\theta(d|q_n, a)] \\
= \sum_{i=1}^{M} \nabla_\theta p_\theta(d_i|q_n, a) \log (1 + \exp(f_\phi(d_i, q_n)) + \nabla_\theta p_\theta(d_i|q_n, a) \log (1 + \exp(f_\phi(d_i, q_n)) - \nabla_\theta p_{true}(d_i|q_n, a) \log p_\theta(d_i|q_n, a) \\
= \sum_{i=1}^{M} p_\theta(d_i|q_n, a) \nabla_\theta \log p_\theta(d_i|q_n, a) \log (1 + \exp(f_\phi(d_i, q_n)) + \nabla_\theta p_{true}(d_i|q_n, a) \log p_\theta(d_i|q_n, a) \\
= \mathbb{E}_{d-p_\theta} \left( \nabla_\theta \log p_\theta(d|q_n, a) \right) \log (1 + \exp(f_\phi(d, q_n))) + \nabla_\theta \mathbb{E}_{d-p_{true}} \left[ \nabla_\theta \log p_\theta(d|q_n, a) \right] \\
= \frac{1}{K} \sum_{k=1}^{K} \nabla_\theta \log p_\theta(d_k|q_n, a) \log (1 + \exp(f_\phi(d_k, q_n))) + \nabla_\theta \mathbb{E}_{d-p_{true}} \left[ \nabla_\theta \log p_\theta(d|q_n, a) \right] \\
= \frac{1}{K} \sum_{k=1}^{K} \frac{p_{true}(d_m|q_n, a)}{p_\theta(d_m|q_n, a)} \log (1 + \exp(f_\phi(d_k, q_n)))
\]

(5)

where the term \( \log (1 + \exp(f_\phi(q_n, a))) \) acts as the balanced rewards from both of the rank discriminator \( D^r_\phi \) and the answer discriminator \( D^a_\phi \).

To reduce the variance during the reinforcement learning pro-

cess, the reward function \( f_{true} \) is usually modified in policy gradient
in practice as below:

\[
f_{\omega} = log \left[ 1 + \exp \left( f_\phi (q_\theta(a)) \right) \right] + \lambda_1 log \left[ 1 + \exp \left( f_\phi (q_\theta(a)) \right) \right] - \mathbb{E}_{p_\theta} \left[ log \left[ 1 + \exp \left( f_\phi (q_\theta(a)) \right) \right] \right] + \lambda_2 log \left[ 1 + \exp \left( f_\phi (q_\theta(a)) \right) \right].
\]

REFERENCES

[1] Akari Asai, Karuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. 2019. Learning to retrieve reasoning paths over wikipedia graph for question answering. arXiv preprint arXiv:1911.10470 (2019).

[2] Pete Baudisch and Jan Sedivy. 2015. Modeling of the question answering task in the yodaqa system. In International Conference of the Cross-Language Evaluation Forum for European Languages. Springer, 222–228.

[3] Homanga Bhargavdhwaj, Homay Namin, and Brian Y Lim. 2018. RecGAN: recurrent generative adversarial networks for recommendation systems. In Proceedings of the 12th ACM Conference on Recommender Systems. 372–376.

[4] Xiaoyan Cai, Junwei Han, and Libin Yang. 2018. Generative adversarial network-based heterogeneous bibliographic network representation for personalized citation recommendation. In Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 32.

[5] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In 35th Annual Meeting of the Association for Computational Linguistics, ACL 2017. Association for Computational Linguistics (ACL), 1870–1879.

[6] Eunsol Choi, Daniel Hewlett, Jakob Uszkoreit, Illia Polosukhin, Alexandre Laconte, and Jonathan Berant. 2017. Coarse-to-fine question answering for long documents. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 209–220.

[7] Xiaoyan Cai, Junwei Han, and Libin Yang. 2018. Generative adversarial network-based heterogeneous bibliographic network representation for personalized citation recommendation. In Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 32.

[8] Rajarshi Das, Shehzad Dhulikawala, Manzil Zaheer, and Andrew McCallum. 2018. Multi-step Retriever-Reader Interaction for Scalable Open-domain Question Answering. In International Conference on Learning Representations.

[9] Bhuwan Dhingra, Kathryn Mazaitis, and William W Cohen. 2017. Quasar: Datasets for question answering by search and reading. arXiv preprint arXiv:1707.03904 (2017).

[10] Matthew Dean, Levent Sagun, Mike Higgins, V Ugur Gunez, Volkan Cirkic, and Kyunghyun Cho. 2017. Searchqa: A new qa dataset augmented with context from a search engine. arXiv preprint arXiv:1704.05179 (2017).

[11] Ishan Durdikar, Ian Gemp, and Sriradh Mahadevan. 2016. Generative multi-adversarial networks. arXiv preprint arXiv:1611.01673 (2016).

[12] Wenqi Fan, Tyler Derr, Yao Ma, Jianping Wang, Jiliang Tang, and Qing Li. 2019. Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. Multi-sentence Scoring. In International Conference on Learning Representations.

[13] Vladimir Karpukhin, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Ilia Polosukhin, Jakob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. Transactions of the Association for Computational Linguistics 7 (2019), 453–466.

[14] Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent Retrieval Answering. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. 6086–6096.

[15] Seanie Lee, Dongyuu Kim, and Jangwon Park. 2019. Domain-agnostic Question-Answering with Adversarial Training. In Proceedings of the 2nd Workshop on Machine Reading for Question Answering. 196–202.

[16] Mike Lewis and Angela Fan. 2018. Generative question answering: Learning to answer the whole question. In International Conference on Learning Representations.

[17] 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). Association for Computational Linguistics, Melbourne, Australia, 1736–1745. https://doi.org/10.18653/v1/P18-1161

[18] Zhuang Liu, Keli Xiao, Bo Jin, Kaity Huang, Degen Huang, and Yunxia Zhang. 2020. Unified generative adversarial networks for multiple-choice oriented machine comprehension. ACM Transactions on Intelligent Systems and Technology (TIST) 11, 3 (2020), 1–20.

[19] Sewon Min, Jordan Boyd-Graber, Chris Alberti, Danqi Chen, Eunsol Choi, Michael Collins, Kelvin Gou, Hannaneh Hajishirzi, Kenton Lee, Jennimaria Palomaki, et al. 2020. NeurIPS 2020 EfficientQA Competition: Systems, Analyses and Lessons Learned. arXiv preprint arXiv:2012.00133 (2020).

[20] Sewon Min, Danqi Chen, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019. A Discrete Hard EM Approach for Weakly Supervised Question Answering. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). 2844–2857.

[21] Sewon Min, Danqi Chen, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019. Knowledge guided text retrieval and reading for open-domain question answering. arXiv preprint arXiv:1911.03868 (2019).

[22] Dinh Nguyen, Trung Le, Hung Vu, and Dinh Phung. 2017. Dual discriminator generative adversarial nets. In Proceedings of the 51st International Conference on Neural Information Processing Systems. 2667–2677.

[23] Yixin Nie, Songhe Wang, and Mohit Bansal. 2019. Revealing the Importance of Semantic Retrieval for Machine Reading at Scale. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). 2553–2566.

[24] Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage Re-ranking with BERT. arXiv preprint arXiv:1901.04083 (2019).

[25] Joel Hoon Oh, Kanuma Kadowaki, Julien Kloetzer, Ryu Iida, and Kentaro Torisawa. 2019. Open-Domain Why-Question Answering with Adversarial Learning to Encode Answer Texts. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. 4227–4237.

[26] Badri Patro, Shivansh Patel, and Vinay Namboodiri. 2020. Robust explanations for visual question answering. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP-CoNLL). 6086–6096.

[27] Sudha Rao and Hal Daumé III. 2019. Answer-based Adversarial Training for Generating Clarification Questions. CoRR abs/1904.02281 (2019). arXiv:1904.02281 http://arxiv.org/abs/1904.02281

[28] Duyu Tang, Nan Duan, Zhao Yan, Zhirui Zhang, Yiho Sun, Shijue Liu, Yuanhua Lv, and Ming Zhou. 2018. Learning to collaborate for question answering and asking. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers). 1544–1547.

[29] Linh Duy Tran, Son Minh Nguyen, and Masayuki Arai. 2020. GAN-based noise model for denoising real images. In Proceedings of the 2020 Conference on Computer Vision.

[30] Jun Wang, Lantoao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. 2017. Irgan: A minimax game for unifying generative adversarial nets. In Proceedings of the 3rd Workshop on Advancing Neural Information Processing Systems 31 (2018).

[31] Sainandam Ramakrishnan, Ashwarya Agrawal, and Stefan Lee. 2018. Overcoming Language Priors in Visual Question Answering with Adversarial Regularization. Advances in Neural Information Processing Systems 31 (2018).

[32] Sainandam Ramakrishnan, Ashwarya Agrawal, and Stefan Lee. 2018. Overcoming language priors in visual question answering with adversarial regularization. arXiv preprint arXiv:1810.03649 (2018).

[33] Sridhar Rao and Dal Daumé III. 2019. Answer-based Adversarial Training for Generating Clarification Questions. CoRR abs/1904.02281 (2019). arXiv:1904.02281 http://arxiv.org/abs/1904.02281

[34] Duyu Tang, Nan Duan, Zhao Yan, Zhirui Zhang, Yiho Sun, Shijue Liu, Yuanhua Lv, and Ming Zhou. 2018. Learning to collaborate for question answering and asking. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers). 1544–1547.

[35] Linh Duy Tran, Son Minh Nguyen, and Masayuki Arai. 2020. GAN-based noise model for denoising real images. In Proceedings of the 2020 Conference on Computer Vision.
and discriminative information retrieval models. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. 515–524.

[41] Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gery Tesuro, Bowen Zhou, and Jing Jiang. 2018. R 3: Reinforced ranker-reader for open-domain question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.

[42] Tomer Wolfson, Mor Geva, Ankit Gupta, Matt Gardner, Yoav Goldberg, Daniel Deutch, and Jonathan Berant. 2020. Break it down: A question understanding benchmark. *Transactions of the Association for Computational Linguistics* 8 (2020), 183–198.

[43] Wenhan Xiong, Hong Wang, and William Yang Wang. 2020. Progressively pretrained dense corpus index for open-domain question answering. *arXiv preprint arXiv:2005.00038* (2020).

[44] Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. 2019. End-to-End Open-Domain Question Answering with BERTserini. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*. 72–77.

[45] Wei Yang, Yuqing Xie, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. 2019. Data augmentation for bert fine-tuning in open-domain question answering. *arXiv preprint arXiv:1904.06652* (2019).

[46] Xiao Yang, Madian Khabsa, Miaosen Wang, Wei Wang, Ahmed Hassan Awadallah, Daniel Kifer, and C Lee Giles. 2019. Adversarial training for community question answer selection based on multi-scale matching. In *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33. 395–402.

[47] Zhihui Yang, Junjie Hu, Ruslan Salakhutdinov, and William Cohen. 2017. Semi-Supervised QA with Generative Domain-Adaptive Nets. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 1040–1050.

[48] Zhiyi Hao, Zhang, Ping Tan, and Minglun Gong. 2017. Dualgan: Unsupervised dual learning for image-to-image translation. In *Proceedings of the IEEE international conference on computer vision*. 2849–2857.

[49] Feng Yuan, Lina Yao, and Boualem Benatallah. 2020. Exploring Missing Interactions: A Convolutional Generative Adversarial Network for Collaborative Filtering. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 1773–1782.

[50] Weinan Zhang. 2018. Generative adversarial nets for information retrieval: Fundamentals and advances. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. 1375–1378.

[51] Zhuosheng Zhang, Junjie Yang, and Hai Zhao. 2020. Retrospective reader for machine reading comprehension. *arXiv preprint arXiv:2001.09694* (2020).

[52] Shihao Zou, Guanyu Tao, Jun Wang, Weinan Zhang, and Dell Zhang. 2018. On the equilibrium of query reformulation and document retrieval. In *Proceedings of the 2018 ACM SIGIR International Conference on Theory of Information Retrieval*. 43–50.