

Cooperative Learning of Zero-Shot Machine Reading Comprehension

Hongyin Luo¹ Seunghak Yu^{2*} James Glass¹

¹MIT CSAIL, ²Amazon Alexa AI

hyluo@mit.edu, yuseungh@amazon.com, glass@mit.edu

Abstract

Pretrained language models have significantly improved the performance of down-stream tasks, for example extractive question answering, by providing high-quality contextualized word embeddings. However, learning question answering models still need large-scale data annotation in specific domains. In this work, we propose a cooperative, self-play learning model for question generation and answering. We implemented a masked answer entity extraction task with an interactive learning environment, containing a question generator and a question extractor. Given a passage with a mask, a question generator asks a question about the masked entity, meanwhile the extractor is trained to extract the masked entity with the generated question and raw texts. With this strategy, we can train question generation and answering models on any textual corpora without annotation. To further improve the performances of the question answering model, we propose a reinforcement learning method that rewards generated questions that improves the extraction learning. Experimental results showed that our model outperforms the state-of-the-art pretrained language models on standard question answering benchmarks, and reaches the state-of-the-art performance under the zero-shot learning setting.

1 Introduction

Recent studies have showed that pretrained language models significantly improves the performance of neural networks on a variety of tasks by providing high-quality language representations (Peters et al., 2018). Based the latest state-of-the-art Transformer model (Vaswani et al., 2017), various language models have been proposed recently (Devlin et al., 2018; Liu et al., 2019; Clark et al.,

Preprint. Work in progress. *Work done before the second author joined Amazon

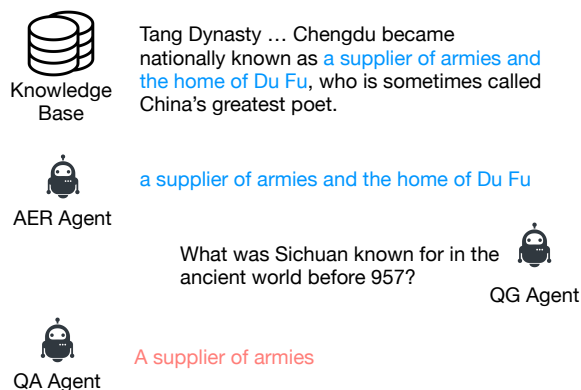


Figure 1: The pipeline of zero-shot machine reading comprehension learning. AER stands for answer entity recognition, QG is question generation, and QA means question answering. The showed answer entity and question are recognized and generated by our model (Chengdu is a city in Sichuan).

2020). These models are pretrained to predict a masked word in a given context on large corpora, and the language representation and knowledge can significantly improve down-stream tasks after finetuning. Although masked language modeling is a powerful self-supervised training strategy, large-scale data annotation is still necessary for finetuning on some difficult down-stream tasks, including extractive question answering, which needs a large amount of annotated question-answer pairs based on training corpora. In this work, we focus on exploring the possibility of learning high-quality question answering while reducing the data annotation efforts.

According to the fact that we can train a high-quality question answering model by providing accurate question-answer pairs with pretrained language models like BERT (Devlin et al., 2018), we solve the semi-supervised and zero-shot question answering problem by reducing the answering task to two subtasks: answer entity recognition

and question generation. Given a textual corpus in any domain, a high-quality question answering model can be trained if (1) answer entities, including phrases, spans, and named entities can be successfully recognized; and (2) a grammatically correct question asking about the target answer entity is annotated. Both answer entity recognizer and question generator can be pretrained on an already annotated seed question answering corpus, and adapt to other unlabeled corpora. Inspired by the masked span selection model proposed by the authors of [Glass et al. \(2019\)](#), we propose the basic automatic question answering corpus building method: (1) recognize possible answer entities in the input passages and replace them with '[MASK]' tokens, and (2) generate question based on masked passages.

In this work, we proposed a method that enables question-answering learning on textual corpora without annotated question-answer pairs. We first pretrain a seed answer entity recognizer and a question generator on a limited number of context-answer-question triples, and apply them on target corpora. Experiment results show that a question answering model trained with the recognized answer entities and generated question can achieve satisfying zero-shot performance.

However, there is always a gap between the pre-training corpus and the target corpus. To better adapt the question generator and answer entity recognizer to the new domain and improve the performance of the question answering model, we propose a reinforced training algorithm. We construct a cooperative environment, where a question generator and a answer extractor work together to solve a masked entity prediction problem. We first replace a tagged named entity in a given passage with a mask token, and the question generator outputs a question based on the masked passage. With the generated question and the original passage without mask tokens, the answer extractor is trained to select the correct span, which should be the previously masked named entity. To successfully extract the masked entity, the question generator needs to provide a good question, and the extractor should select the most likely word or span. We reward the questions that lead to correct answers, and also progressively increase the difficulties of generated question ([Karpukhin et al., 2020](#)) by also rewarding the questions that are not correctly answered but with relatively low extraction losses.

With this method, we can train a question an-

swering model on a seed corpus, and then adapt to any other textual corpus without question-answer annotations. An example of the proposed pipeline is shown in Figure 1. In this work, we make following contributions,

- We solve the zero-shot extractive question answering task by split it into the answer entity recognition and question generation sub-tasks.
- We propose a cooperative, reinforced self-training framework that better adapts the pre-trained models to the target corpus.
- We design a stochastic expectation-maximization algorithm that identifies difficult but answerable questions without supervision.

2 Related Work

Representation learning has been an important topic in the area of natural language processing since neural language models were proposed ([Bengio et al., 2003](#)). Based on word co-occurrence, [Mikolov et al. \(2013\)](#) and [Pennington et al. \(2014\)](#) proposed language embedding algorithms focusing on word-level semantics. Recent studies have focused on pretraining contextualized word representations on large-scale corpora ([Peters et al., 2018](#)). State-of-the-art pretrained representation models are trained with the masked language modeling task ([Devlin et al., 2018](#); [Liu et al., 2019](#); [Clark et al., 2020](#)) applying the Transformer architecture ([Vaswani et al., 2017](#)).

Different variants of masked language models have been proposed to improve finetuning performances from different aspects. [Joshi et al. \(2020\)](#) proposed a masked span generation task instead of word prediction. [Fei et al. \(2020\)](#) and [Shen et al. \(2020\)](#) proposed models that learn better syntax knowledge with syntactic distances ([Shen et al., 2018](#)) and heights ([Luo et al., 2019](#)). [Henderson et al. \(2019\)](#) and [Humeau et al. \(2019\)](#) showed that pretraining language models on dialog corpora performs better on dialog-related finetuning tasks, comparing with pretraining on Wikipedia. To improve the finetuning performance on the question answering task, a span selection task is proposed in [Glass et al. \(2019\)](#) to reduce the gap between the pretraining and finetuning question answering task.

Recently, different training strategies, in addition to maximum likelihood training, for learning language generation has emerged, including reinforce-

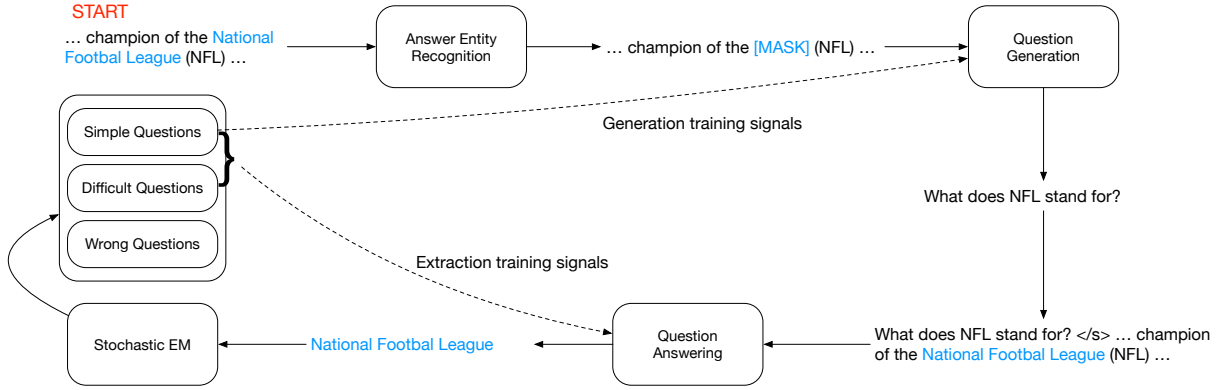


Figure 2: The cooperative learning pipeline for machine reading comprehension. The model starts from a passage and follows the illustrated steps: (1) recognize a potential answer entity, (2) generate a question asking about the answer entity, and (3) answer the question by extracting the answer.

ment learning and self-training. To optimize text generation models directly with non-differentiable objective functions, Rennie et al. (2017) proposed self-critical sequence training (SCST) using policy gradient (Kakade, 2001; Silver et al., 2014). On the other hand, self-training has been proved effective in different tasks, including machine translation (He et al., 2019), image classification (Xie et al., 2020), and structured database-grounded question answering (Xu et al., 2020).

Self-training methods usually apply a teacher-student learning strategy. However, cooperative learning pipelines contains several agents that works together to learn as much knowledge as possible. A typical cooperative learning model is generative adversarial networks (GAN) (Goodfellow, 2016; Goodfellow et al., 2014), in which a generator is trained to confuse a discriminator. Sequence GAN is designed for learning diverse text generation in this manner (Yu et al., 2017). Different from the adversarial learning method where two network works for opposite goals, other studies proposed learning environments where different agents learn the same objective functions for language emergence (Lazaridou et al., 2016; Mordatch and Abbeel, 2018; Havrylov and Titov, 2017), including simple natural language, compositional language, and symbolic language. In the domain of question answering, Shakeri et al. (2020) proposed to generate synthetic question-answer pairs with an end-to-end model simultaneously, and the model introduced in Lee et al. (2020) generates question-answer pairs with VAE, which achieved the previous state-of-the-art performance in zero-shot machine reading comprehension.

3 Zero-Shot Question Answering

In this work, we propose a cooperative, zero-shot question answering learning algorithm that enables training question answering agents on knowledge bases without large-scale data annotation. In this section, we first introduce the zero-shot question answering task with pretrained models on the seed corpora, and then propose a cooperative question answering learning environment for further improving the performance.

3.1 Zero-Shot QA with Pretrained Models

A fully supervised question answering learning task requires large-scale annotation of question-answer pairs. However, the annotation process could be both expensive and biased. To avoid this problem, we explore the possibility for reducing the data annotation efforts. The basic idea is train an answer entity recognition model and a question generator on an annotated seed corpus, and build new question answering corpora on new textual data by generating new question-answer pairs. Then we can train question answering models with state-of-the-art neural networks with the constructed training corpus. The pipeline of our proposed method is illustrated in Figure 2.

3.1.1 Answer Entity Recognition

We first train an answer entity recognition (AER) model on a seed QA corpus. We did this because we found it performs poorly to get potential answers with a named entity recognition (NER) model pretrained on CONLL 2003 shared task (Bender et al., 2003). We will show the experiment results later. Given a passage p , annotated question q , and the annotated answer entity e , we first ran-

domly select a sentence s containing e from p . We train a BERT-based BIO sequence tagging model that takes s as input, with cross entropy losses and the labeled answer e . We use this method as a baseline (AER-Tag).

In our experiments, we found that the quality of AER significantly influence the question answering accuracy. We proposed a extractive AER model, which is very similar as the question answering model. Given a sentence s , we train an extraction model that predicts the start and end position of the answer entity e . With this method, we can easily get potential answer entities by sorting the sum of start and end scores of all candidate spans. We further re-rank the answer entities by selecting those leads to low question generation perplexities (AER-Search).

3.1.2 Question Generation

Assume the input passage is p , and the annotated answer entity is e . As described previously, we replace the answer entity in p with a [MASK] token, and the masked passage is p^* . We represent the question generator as Q . Our goal is training a question generator that outputs answerable questions q in natural language. To improve the quality of the generation model, we concatenate the masked passage and the annotated answer entity e

$$q = Q([p^*, e]) \quad (1)$$

In practice, Q is the BART sequence-to-sequence model (Lewis et al., 2019), and we train Q on question-answer pairs on the Natural Questions corpus (Kwiatkowski et al., 2019) by maximizing the likelihood of annotated questions.

3.1.3 Question Answering

After pretraining the answer entity recognition and question generation models, we apply them on other the target textual corpora. We train a question answering model on an un-annotated corpus with three steps,

- Recognize answer entities with the pretrained AER model
- Generate questions based on the recognized answer entities with the pretrained question generator
- Conduct standard extractive question answering training on the generated question-answer pairs.

The answer extraction model A takes generated question q and the original, un-masked passage p as inputs. Following the standard extractive QA method, we concatenate the question q and passage p , and train the model to locate the answer in the passage, which is expected to be the answer entity e masked for question generation.

$$I_{st}, I_{ed} = A([q, x]) \quad (2)$$

where I_{st} and I_{ed} stand for the start and end positions of the answer in the passage respectively. In optimization, we train the start position and end position predictions separately with cross entropy losses.

3.2 Zero-shot QA with Cooperative Learning

Although the pretrained models might be strong enough to handle corpora from different domains, there is always a gap between the seed QA corpus and the target textual corpus. To efficiently adapt the pretrained models to the new domains, we propose a cooperative learning algorithm that allows finetuning the models on the target corpora without additional annotations. The finetuning is based on a two-agent cooperative environment. The two agents, namely question generator and answer extractor, work together to predict a target answer entity. To achieve this goal, we need a pretrained answer extraction model on the seed QA corpus, and the cooperative system processes its input passages from the target domain with the following steps,

- Produce a masked passage by replacing a randomly selected answer entity with the “[MASK]” token
- Generate a question asking about the masked entity with the pretrained question generator.
- Input the generated question and the original passage into the pretrained QA model, which attempts to extract the masked answer entity from the original passage.
- Optimize the answer extractor with the ground-truth masked named entities.
- Optimize the question generator with self-critical sequence training.

According to the proposed pipeline, all models, including AER, QG, and QA, need pretraining to provide a reasonable start point for the cooperative self-training. However, the pretrained models are

not perfect since there's domain gaps between the pretraining corpus and the target corpus. In order to improve the adaptation of pretrained models on the target domain, we need to distinguish which generated questions are correct. The correctness of a question are defined on two folds. Firstly, it should be fluent and readable, and secondly, it should ask about the masked answer entity instead of the others. One method to identify the quality of questions is human-in-loop training. In other words, when a new question is generated, a human scorer is assigned to the question and decide if the question is both grammatically correct and asking the target answer entity. However, this method is both inefficient and expensive. To solve this problem, we propose a stochastic expectation-maximization (SEM) method that learns the correctness of the generated without supervision. The pipeline of the cooperative learning is illustrated in Figure 2.

3.2.1 Stochastic Expectation-Maximization

In this work, we divide all generated questions into three groups based on their difficulties: simple questions, difficult questions, and wrong questions. In each step of training, simple questions stand for the questions that leads to low-perplexity answers with the pretrained QA model, difficult questions are correct questions that leads to higher uncertainty in question answering. Meanwhile, wrong questions are grammarly incorrect or not asking about the correct answer entity in the given passage. If we train the answering model with all questions, the training signal would be very noisy since there exists wrong questions. On the other hand, we hope the question generator provide as many correct question as possible. As a result, we only train the question answering model with the simple and difficult questions, and reward the question generation model with the all simple questions and correctly answered difficult questions.

In order to conduct the fully-unsupervised question clustering task, we classify a given questions into one of the three types described above by setting thresholds of the answering loss it leads to. It is worth note that simply setting the thresholds as hyper-parameters does not work, since the extraction model is being trained and the overall extraction loss decreases during the training. In order to find the extraction loss margins adaptively, we propose the SEM algorithm.

Similar with the standard EM, the SEM algorithm starts with a set of initial parameters μ_i^0, σ_i^0 ,

and $i \in [0, 2]$ stands for one of the three types of questions. μ_i is the mean value of the extraction loss led by the questions of type i , and σ_i stands for the standard variance of the losses of type- i questions. We keep updating the values of μ_i and σ_i during training. Assume the extraction loss of the question answering model is l_j^t , where j stand for the j -th sample in the current training batch, and t is the current training step. We decide the type of question j by

$$c_j = \underset{i}{\operatorname{argmin}} \frac{|l_j^t - \mu_i|}{\sigma_i} \quad (3)$$

where c_j is the predicted type of the question of training example j . After, we calculate the mean value $\hat{\mu}_i$ and standard variance $\hat{\sigma}_i$ of the extraction losses of all question-answer pairs in the current training for each question types, and update the global mean and standard variance of each type with

$$\mu_i^{t+1} = \mu_i^t * \alpha + \hat{\mu}_i * (1 - \alpha) \quad (4)$$

and

$$\sigma_i^{t+1} = \mu_i^t * \beta + \hat{\sigma}_i * (1 - \beta) \quad (5)$$

where μ_i^{t+1} and σ_i^{t+1} stands for the new mean values and standard variances of extraction losses for each class. The questions from the next training batch will be clustered with the updated parameters. We call this method stochastic EM because similar as stochastic gradient descent, the method updates trainable parameters batch-by-batch instead of observing all training data. In the following sections, we define $i = 0$ for simple questions, $i = 1$ for difficult questions, and $i = 2$ stands for wrong questions. Based on the definition, we have $u_0^t < u_1^t < u_2^t$ for all the time step ts .

3.2.2 Reinforced Question Generation

To improve the question generation model on the target domain, we train the QG model with policy gradient obtained by the SEM algorithm. We reward all simple questions, and difficult questions that are correctly answered. While optimizing the generation model, we compare the likelihoods of the generated question q calculated by the optimized generator and the pretrained generator in order to keep the grammar correctness of the generated questions. For a generated question q to be rewarded, we calculate the rewards by

$$r_q = \max(1 - \alpha \cdot D_{KL}(P(q) || P^*(q)), 0) \quad (6)$$

where D_{KL} stands for KL divergence, P stands for the likelihood of q based on the current generation model, p^* stands for the pretrained question generation model, and α is a positive coefficient. We do not reward any question which gets very high perplexity according to the pretrained model. After getting the rewards, we train the question generation model by maximizing the likelihood of the selected generated questions by

$$l_q = r_q \cdot P(q) \quad (7)$$

To preserve the grammar correctness of the generation model, we only assign positive rewards to the question generator. It is worth noting that we do not always reward the same type of questions, and the choice is based on the seed QA corpus for pretraining. If most questions in the seed corpus are very simple, keep rewarding simple generated questions might lead to a trivial convergence, for example, simple sub-sequence matching.

4 Experiments

4.1 Modules

In this work, we train three modules for building the training environment, an answer entity recognizer, a question generator, and an answer extractor, for our cooperative environment. For the semi-supervised scenario, we do not have to pretrain the answer extractor, but it is necessary for the cooperative zero-shot learning. We used a BERT (Devlin et al., 2018) model for answer recognition, a BART (Lewis et al., 2019) model for question generation, and an ELECTRA (Clark et al., 2020) model for answer extraction.

5 Data

In this work, we used the Natural Question (Kwiatkowski et al., 2019) dataset as the seed corpus for pretraining all modules introduced above. To show the performance of the proposed model on question answering tasks with different difficulty levels, we conduct experiments on Both SQuAD v1.1 (Rajpurkar et al., 2016) and AdversarialQA corpora (Bartolo et al., 2020).

The SQuAD v1.1 is the easiest QA corpus we use in the work, which contains 107,785 question-answer pairs on 536 articles, which are splitted into passages. Each question is labeled with an answer that can be extracted from the given passage.

The Natural Questions dataset is a large-scale corpus designed for open-domain question answering, which is harder than SQuAD. All questions are correct from human search queries and is annotated with long answers. Some of the questions are also annotated with a short answer for learning answer-span selection, or reading comprehension. Focusing on the machine reading comprehension task, we select 106,926 questions that are labeled with both long and short answers from the dataset.

AdversarialQA, the most difficult task evaluated in this work, is constructed by collecting natural-language questions that can be correctly answered by human, but cannot be answered by state-of-the-art pretrained language models, including BERT (Devlin et al., 2018) and RoBERTa (Liu et al., 2019). The corpus contains 30,000 question-answer pairs for training, which is smaller comparing with the SQuAD corpus, but the questions are more difficult for machine reading comprehension models. We compare our model on both SQuAD and AdversarialQA corpus, to analysis the quality of the recognized answer entities and generated questions.

5.1 Implementation Details

In this work, the models are trained with three phases: pretraining on the Natural Question corpus, cooperative adaptation with reinforcement learning on the target corpora, and final finetuning on the target corpora. We mostly applied the hyper-parameters reported in the original BERT (Devlin et al., 2018), BART (Lewis et al., 2019), and ELECTRA (Clark et al., 2020) papers. However, we noticed that the ELECTRA model is not stable with the generated question-answer pairs in final finetuning. As a result, we select the final finetuning learning rates from $\{3e-5, 4e-5, 5e-5\}$ and report the highest performance, but fix all other hyper-parameters as reported in the corresponding papers. For all training phases, we fix $eps = 1e-6$ and $s_w = 2000$, where s_w stands for warm up steps, and we applied no weight decays. In the following sections, we describe the details of different training phases.

5.1.1 Pretraining

We pretrain the answer entity recognition, question generation, and question answering models on the Natural Questions corpus. Since the seed corpus is designed for learning question-answering models and each passage only contains one answer

entities, the training data is very sparse. To solve the problem, we only use the sentence containing the answer entity to balance the token classes. We use the entire long answers for both question generation and question answering, which have been described with Equation 1 and 2.

5.1.2 Cooperative Learning

In the cooperative training phase, we follow the steps described in section 3.2, which including answer entity recognition, question generation, and question answering steps in order. Since the training data of the answer entity recognition model is sparse, we first split input passages into sentences, and then run the AER model on the sentences to predict the answer entities. After recognized answer entities from each sentence, we randomly sample one answer entity to train the question generation and answering models.

5.1.3 Final Finetuning

For finetuning, we apply the similar method for preprocessing as we used in cooperative learning. We first recognize all possible answer entities in each passage in the training set of the target corpus. After the recognition process is finished, we generate questions with all recognized answer entities. We construct a new training corpus. We then finetune the question answering model with the official Huggingface (Wolf et al., 2019) training scripts for question answering ¹.

5.2 Finetuning Performance

To evaluate the performance of the proposed models, we finetune the proposed model on the target corpus with both zero-shot learning and semi-supervised learning task. Note that our semi-supervised learning means that we use annotations of the answer entities, but discard the annotated questions. For the zero-shot learning task, we use no annotated data, but re-construct the corpus with recognized answer entities and generated question instead.

5.2.1 Semi-supervised QA

We first evaluate the proposed model on the semi-supervised reading comprehension task. Under this setting, we only use the annotated answer entities for cooperative training, and use both annotated answer entities and generated questions for finetuning.

¹<https://github.com/huggingface/transformers/tree/master/examples/question-answering>

The performances of the models on each corpus and the performance of the state-of-the-art model, under the fully supervised setting is shown in Table 1 on both SQuAD and AdversarialQA corpora.

Models	EM	F1
SQuAD 1.1		
PT	81.2	89.1
CT	83.1	90.7
Supervised ELECTRA-large	89.7	94.9
AdversarialQA		
PT	27.9	41.4
CT	30.3	43.9
Supervised ELECTRA-large	50.1	62.9

Table 1: The performance of the models on the semi-supervised task. The supervised SOTA results are trained with the Electra-large-discriminator model. PT means the pretrained model, and CT stands for the cooperatively trained model.

Table 1 showed that our both models achieve satisfying performance, and the cooperative learning method reaches over 90% F1 score, approaching the performance of the state-of-the-art (SOTA) performance, which is achieved by training an ELECTRA-large-discriminator model on the entire annotated training set. On the AdversarialQA corpus, the cooperative learning model also outperformed the pretrained models, but the performance decreases more after switching to semi-supervised settings. The experiment results show that the cooperative learning strategy improves the question generation model, but it does not generate human-level difficult question, since the performance drop on AdversarialQA is higher than SQuAD.

5.2.2 Zero-Shot QA

We also evaluate the models under the zero-shot learning settings, in which we do not use any annotated data, including questions and answer entities. We first run the AER model to recognize all answer entities, and generate corresponding questions with the QG model. We train the question-answering model based on the reconstructed training set and evaluate the model on the original test set. We also compare the zero-shot experiment results and the state-of-the-art performance in Table 2.

The experiment results showed above indicate the the models trained on SQuAD still achieves over 80% F1 score, although no annotation is used in training. However, the performance of the Ad-

Models	EM	F1
SQuAD 1.1		
Info-HCVAE (Lee et al., 2020)	71.2	81.5
PT + NER	18.4	25.7
CT + NER	27.4	35.4
PT + AER-T	60.8	75.7
CT + AER-T	71.4	82.4
PT + AER-S	70.2	83.8
CT + AER-S	72.7	85.9
Supervised ELECTRA-large	89.7	94.9
AdversarialQA		
PT + AER-T	17.7	29.5
CT + AER-T	21.8	34.4
PT + AER-S	22.6	37.1
CT + AER-S	24.1	38.7
Supervised ELECTRA-large	50.1	62.9

Table 2: The performance of the models on the zero-shot question answering task. NER means using the pretrained BERT named entity recognition model on the CONLL 2003 shared task, AER-T stands for recognizing answer entities with Tagging models, and AER-S stands for the extractive AER-Search method.

versarialQA is worse, and the cooperative learning model still significantly better than the pretrained models. It also showed that recognizing answer entities with the NER model pretrained on CONLL corpus does not lead to reasonable extraction performance, proving the importance of our AER module.

We also noticed that the improvement gained by the cooperative learning over the pretrained models is higher in the zero-shot learning than the semi-supervised learning task on both SQuAD and AdversarialQA tasks. This indicates that the model trained with the cooperative environment is more robust against the automatically recognized answer entities, which are more noisy than the annotated answer entities.

5.3 Analysis

5.3.1 Question Generation

In this section, we evaluate the quality of the generated questions on all the three corpora. Since we do not have human-annotated questions on the recognized answer entities, we generate questions using the annotated answers. We compare the question lengths and vocabulary sizes of the generated questions. The statistics are shown in Table 3.

The statistics showed that the vocabulary sizes of the ground-truth questions are larger than generated questions in both corpora, and the questions lengths are also more diverse. We found that after cooperative learning, the standard deviation of questions lengths are smaller. Under the zero-shot learning setting, the vocabulary size of the generated questions is significantly smaller than the ground-truth questions, indicating that the diversity of the recognized answers is also lower than the annotated data. Also, the vocabulary size of the generated questions is much smaller on a more difficult corpus.

Models	Mean Len.	Std Len.	Vocab
SQuAD			
Ground-truth	11.29	3.72	988703
Semi-sup. PT	10.49	2.48	919105
Semi-sup. CT	10.54	1.91	923191
Zero-shot PT	10.57	2.63	789924
Zero-shot CT	10.53	1.94	873300
AdversarialQA			
Ground-truth	10.83	4.71	324919
Semi-sup. PT	10.64	2.57	319144
Semi-sup. CT	10.63	1.92	318978
Zero-shot PT	10.56	2.65	108620
Zero-shot CT	10.53	1.9	108270

Table 3: The statistics of the generated questions in terms of the vocabulary size and lengths on different corpora and experiment settings, compared with the ground-truth, human-annotated questions.

Models	Mean Len.	Ans./Psg.	Vocab
SQuAD (hit rate: 15.35%)			
Ground-truth	3.38	4.64	295653
AER	2.63	4.20	208994
AdversarialQA (hit rate: 6.19%)			
Ground-truth	4.22	11.32	126731
AER	2.64	4.13	28954

Table 4: Statistics of annotated and recognized answer entities. The vocabulary size of annotated answers is significantly larger than the recognized answer set. Ans/Psg means the average number of answers in a passage, and the hit rate stands for the ratio of annotated answers that are successfully recognized by the pretrained AER model.

5.3.2 Answer Entity Recognition

We analysis the answer entity recognize model by the overlap rate of recognized answers and annotated, or ground-truth answers in both corpora in Table 4. We also show the statistics of the recognized answer entities with the number of answers in each passage, their lengths, and the vocabulary sizes.

Similar with the generated question, the recognized answer entities are less diverse than the annotated answers, in terms of both vocabulary size and answer lengths. The vocabulary size of the recognized answers is significantly smaller on the AdversarialQA task than SQuAD, leading to less diverse question generation. We also noticed that the hit rate, which stands for the ratio of annotated answers that are successfully recognized by the AER model is not high on both corpus, but the zero-shot question-answering model still achieves reasonable performances.

6 Conclusion

In this work, we boost the performance of zero and few-shot machine reading comprehension by developing a cooperative self-play training pipeline. We also proposed a novel stochastic expectation-maximization method to distinguish the correctness of the generated questions. Experiments show that the proposed model significantly outperforms the pretrained models on both zero-shot and semi-supervised settings. Further more, we also find that the question answering model achieves almost state-of-the-art performance with the generated question-answer pairs and very small subset of annotated data. Experiments and analysis showed that cooperative self-play training is a promising method to make machine learning methods more scalable and less depending on data annotation.

References

Max Bartolo, Alastair Roberts, Johannes Welbl, Sebastian Riedel, and Pontus Stenetorp. 2020. Beat the ai: Investigating adversarial human annotation for reading comprehension. *Transactions of the Association for Computational Linguistics*, 8:662–678.

Oliver Bender, Franz Josef Och, and Hermann Ney. 2003. Maximum entropy models for named entity recognition. In *Proceedings of CoNLL-2003*, pages 148–151. Edmonton, Canada.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic lan-

guage model. *Journal of machine learning research*, 3(Feb):1137–1155.

- Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.
- Hao Fei, Yafeng Ren, and Donghong Ji. 2020. Retrofitting structure-aware transformer language model for end tasks. *arXiv preprint arXiv:2009.07408*.
- Michael Glass, Alfio Gliozzo, Rishav Chakravarti, Anthony Ferritto, Lin Pan, GP Bhargav, Dinesh Garg, and Avirup Sil. 2019. Span selection pre-training for question answering. *arXiv preprint arXiv:1909.04120*.
- Ian Goodfellow. 2016. Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*.
- Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. *Advances in neural information processing systems*, 27:2672–2680.
- Serhii Havrylov and Ivan Titov. 2017. Emergence of language with multi-agent games: Learning to communicate with sequences of symbols. In *Advances in neural information processing systems*, pages 2149–2159.
- Junxian He, Jiatao Gu, Jiajun Shen, and Marc’Aurelio Ranzato. 2019. Revisiting self-training for neural sequence generation. *arXiv preprint arXiv:1909.13788*.
- Matthew Henderson, Iñigo Casanueva, Nikola Mrkšić, Pei-Hao Su, Tsung-Hsien Wen, and Ivan Vulić. 2019. Convert: Efficient and accurate conversational representations from transformers. *arXiv preprint arXiv:1911.03688*.
- Samuel Humeau, Kurt Shuster, Marie-Anne Lachaux, and Jason Weston. 2019. Poly-encoders: Transformer architectures and pre-training strategies for fast and accurate multi-sentence scoring. *arXiv preprint arXiv:1905.01969*.
- Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.
- Sham M Kakade. 2001. A natural policy gradient. *Advances in neural information processing systems*, 14:1531–1538.

- Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*.
- Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.
- Angeliki Lazaridou, Alexander Peysakhovich, and Marco Baroni. 2016. Multi-agent cooperation and the emergence of (natural) language. *arXiv preprint arXiv:1612.07182*.
- Dong Bok Lee, Seanie Lee, Woo Tae Jeong, Donghwan Kim, and Sung Ju Hwang. 2020. Generating diverse and consistent qa pairs from contexts with information-maximizing hierarchical conditional vaes. *arXiv preprint arXiv:2005.13837*.
- Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.
- Hongyin Luo, Lan Jiang, Yonatan Belinkov, and James Glass. 2019. Improving neural language models by segmenting, attending, and predicting the future. *arXiv preprint arXiv:1906.01702*.
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositional-ity. *Advances in neural information processing systems*, 26:3111–3119.
- Igor Mordatch and Pieter Abbeel. 2018. Emergence of grounded compositional language in multi-agent populations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.
- Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.
- 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*.
- Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.
- Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. 2017. Self-critical sequence training for image captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7008–7024.
- Siamak Shakeri, Cicero Nogueira dos Santos, Henry Zhu, Patrick Ng, Feng Nan, Zhiguo Wang, Ramesh Nallapati, and Bing Xiang. 2020. End-to-end synthetic data generation for domain adaptation of question answering systems. *arXiv preprint arXiv:2010.06028*.
- Yikang Shen, Zhouhan Lin, Athul Paul Jacob, Alessandro Sordani, Aaron Courville, and Yoshua Bengio. 2018. Straight to the tree: Constituency parsing with neural syntactic distance. *arXiv preprint arXiv:1806.04168*.
- Yikang Shen, Yi Tay, Che Zheng, Dara Bahri, Donald Metzler, and Aaron Courville. 2020. Structformer: Joint unsupervised induction of dependency and constituency structure from masked language modeling. *arXiv preprint arXiv:2012.00857*.
- David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. 2014. Deterministic policy gradient algorithms.
- 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*, pages 5998–6008.
- Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface’s transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.
- Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. 2020. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10687–10698.
- Silei Xu, Sina J Semnani, Giovanni Campagna, and Monica S Lam. 2020. Autoqa: From databases to qa semantic parsers with only synthetic training data. *arXiv preprint arXiv:2010.04806*.
- Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. Seqgan: Sequence generative adversarial nets with policy gradient. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31.