Harvesting and Refining Question-Answer Pairs for Unsupervised QA

Zhongli Li[†]*, **Wenhui Wang**[‡], **Li Dong**[‡], **Furu Wei**[‡], **Ke Xu**[†]
[†]Beihang University
[‡]Microsoft Research
{lizhongli@,kexu@nlsde.}buaa.edu.cn
{wenwan,lidong1,fuwei}@microsoft.com

Abstract

Question Answering (QA) has shown great success thanks to the availability of large-scale datasets and the effectiveness of neural models. Recent research works have attempted to extend these successes to the settings with few or no labeled data available. In this work, we introduce two approaches to improve unsupervised QA. First, we harvest lexically and syntactically divergent questions from Wikipedia to automatically construct a corpus of question-answer pairs (named as REFQA). Second, we take advantage of the QA model to extract more appropriate answers, which iteratively refines data over REFQA. We conduct experiments[1] on SQuAD 1.1, and NewsQA by fine-tuning BERT without access to manually annotated data. Our approach outperforms previous unsupervised approaches by a large margin and is competitive with early supervised models. We also show the effectiveness of our approach in the few-shot learning setting.

1 Introduction

Extractive question answering aims to extract a span from the given document to answer the question. Rapid progress has been made because of the release of large-scale annotated datasets (Rajpurkar et al., 2016, 2018; Joshi et al., 2017), and well-designed neural models (Wang and Jiang, 2016; Seo et al., 2016; Yu et al., 2018). Recently, unsupervised pre-training of language models on large corpora, such as BERT (Devlin et al., 2019), has brought further performance gains.

However, the above approaches heavily rely on the availability of large-scale datasets. The collection of high-quality training data is time-consuming and requires significant resources, especially for new domains or languages. In order to tackle the setting in which no training data available, Lewis et al. (2019) leverage unsupervised machine translation to generate synthetic context-question-answer triples. The paragraphs are sampled from Wikipedia. NER and noun chunkers are employed to identify answer candidates. Cloze questions are first extracted from the sentences of the paragraph, and then translated into natural questions. However, there are a lot of lexical overlaps between the generated questions and the paragraph. Similar lexical and syntactic structures render the QA model tend to predict the answer just by word matching. Moreover, the answer category is limited to the named entity or noun phrase, which restricts the coverage of the learnt model.

In this work, we present two approaches to improve the quality of synthetic context-question-answer triples. First, we introduce the REFQA dataset, which harvests lexically and syntactically divergent questions from Wikipedia by using the cited documents. As shown in Figure 1, the sentence (statement) in Wikipedia and its cited documents are semantically consistent, but written with different expressions. More informative context-question-answer triples can be created by using the cited document as the context paragraph and extracting questions from the statement in Wikipedia. Second, we propose to iteratively refine data over REFQA. Given a QA model and some REFQA examples, we first filter its predicted answers with a probability threshold. Then we refine questions based on the predicted answers, and obtain the refined question-answer pairs to continue the model training. Thanks to the pretrained linguistic knowledge in the BERT-based QA model, there are more appropriate and diverse answer candidates in the filtered predictions, some of which do not appear in the candidates extracted by NER tools. We also show

*Contribution during internship at Microsoft Research.
[1]The code and data are available at https://github.com/Neutralzz/RefQA.

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 6719–6728
July 5 - 10, 2020.

that iteratively refining the data further improves model performance.

We conduct experiments on SQuAD 1.1 (Rajpurkar et al., 2016), and NewsQA (Trischler et al., 2017). Our method yields state-of-the-art results against strong baselines in the unsupervised setting. Specifically, the proposed model achieves 71.4 F1 on the SQuAD 1.1 test set and 45.1 F1 on the NewsQA test set without using annotated data. We also evaluate our method in a few-shot learning setting. Our approach achieves 79.4 F1 on the SQuAD 1.1 dev set with only 100 labeled examples, compared to 63.0 F1 using the method of Lewis et al. (2019).

To summarize, the contributions of this paper include: i) REFQA constructing in an unsupervised manner, which contains more informative context-question-answer triples. ii) Using the QA model to iteratively refine and augment the question-answer pairs in REFQA.

2 Related Work

Extractive Question Answering Given a document and question, the task is to predict a continuous sub-span of the document to answer the question. Extractive question answering has garnered a lot of attention over the past few years. Benchmark datasets, such as SQuAD (Rajpurkar et al., 2016, 2018), NewsQA (Trischler et al., 2017) and TriviaQA (Joshi et al., 2017), play an important role in the progress. In order to improve the performance on these benchmarks, several models have been proposed, including BiDAF (Seo et al., 2016), R-NET (Wang et al., 2017), and QANet (Yu et al., 2018). Recently, unsupervised pre-training of language models such as BERT (Devlin et al., 2019), achieves significant improvement. However, these powerful models rely on the availability of human-labeled data. Large annotated corpora for a specific domain or language are limited and expensive to construct.

Semi-Supervised QA Several semi-supervised approaches have been proposed to utilize unlabeled data. Neural question generation (QG) models are used to generate questions from unlabeled passages for training QA models (Yang et al., 2017; Zhu et al., 2019b; Alberti et al., 2019; Dong et al., 2019). However, the methods require labeled data to train the sequence-to-sequence QG model. Dhingra et al. (2018) propose to collect synthetic context-question-answer triples by generating cloze-style questions from the Wikipedia summary paragraphs in an unsupervised manner.

Unsupervised QA Lewis et al. (2019) have explored the unsupervised method for QA. They create synthetic QA data in four steps. i) Sample paragraphs from the English Wikipedia. ii) Use NER or noun chunkers to extract answer candidates from the context. iii) Extract "fill-in-the-blank" cloze-style questions given the candidate answer and context. iv) Translate cloze-style questions into natural questions by an unsupervised translator. Compared with Dhingra et al. (2018), Lewis et al. (2019) attempt to generate natural questions by training an unsupervised neural machine translation (NMT) model. They train the NMT model on non-aligned corpora of natural questions and cloze questions. The unsupervised QA model of Lewis et al. (2019) achieves promising results, even outperforms early supervised models. However, their questions are generated from the sentences or sub-clauses of the same paragraphs, which may lead to a biased learning of word matching since its similar lexicons and syntactic structures. Besides, the category of answer candidates is limited to named entity or noun phrase, which restricts the coverage of the learnt QA model.

3 Harvesting REFQA from Wikipedia

In this section, we introduce REFQA, a question answering dataset constructed in an unsupervised manner. One drawback of Lewis et al. (2019) is that questions are produced from the paragraph sentence that contains the answer candidate. So there are considerable expression overlaps between generated questions and context paragraphs. In contrast, we harvest informative questions by taking advantage of Wikipedia's reference links, where lexical and syntactic differences exist between the article and its cited documents.

As shown in Figure 1, given statements in Wikipedia paragraphs and its cited documents, we use the cited documents as the context paragraphs and generate questions from the sub-clauses of statements. In order to generate question-answer pairs, we first find answer candidates that appear in both sub-clauses and context paragraphs. Next, we convert sub-clauses into the cloze questions based on the candidate answers. We then conduct cloze-to-natural-question translation by a depen-

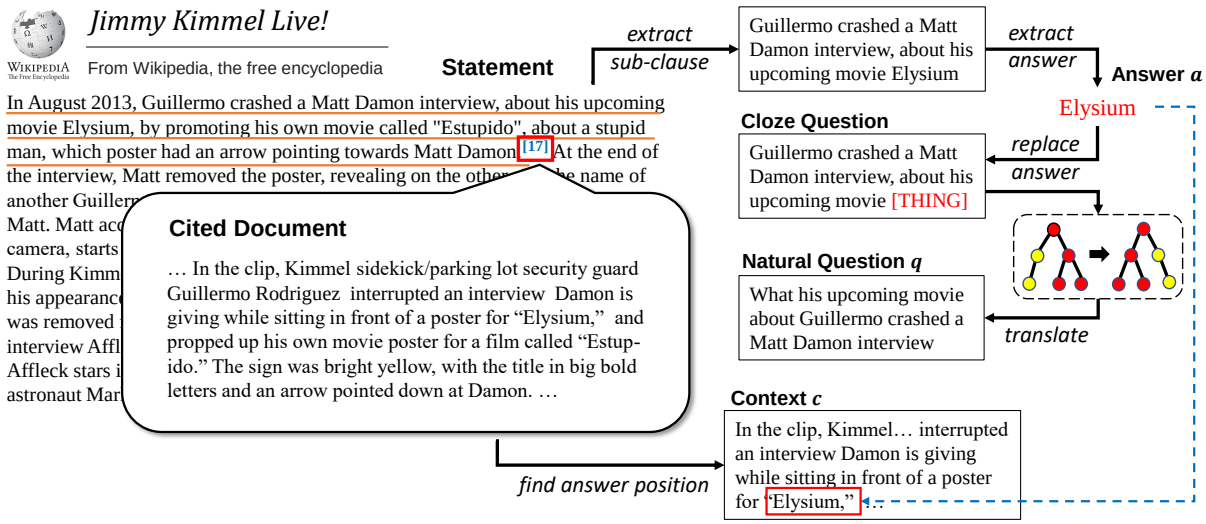

Figure 1: Overview of REFQA construction.

dency tree reconstruction algorithm. We describe the details as follows.

3.1 Context and Answer Generation

Statements in Wikipedia and its cited documents often have similar content, but are written with different expressions. Informative questions can be obtained by taking the cited document as the context paragraph, and generate questions from the statement. We crawl statements with reference links from the English Wikipedia. The cited documents are obtained by parsing the contents of reference webpages.

Given a statement and its cited document, we restrict the statement to its sub-clauses, and extract answer candidates (i.e., named entities) that appear in both of them by using a NER toolkit. We then find the answer span positions in the context paragraph. If the candidate answer appears multiple times in the context, we select the position whose surrounding context has the most overlap with the statement.

3.2 Question Generation

We first generate cloze questions (Lewis et al., 2019) from the sub-clauses of Wikipedia statements. Then we introduce a rule-based method to rewrite them to more natural questions, which utilizes the dependency structures.

3.2.1 Cloze Generation

Cloze questions are the statements with the answer replaced to a mask token. Following Lewis et al. (2019), we replace answers in statements with a special mask token, which depends on its answer category[2]. Using the statement and the answer (with a type label `PRODUCT`) from Figure 1, this leaves us with the cloze question "*Guillermo crashed a Matt Damon interview, about his upcoming movie* `[THING]`".

3.2.2 Translate Clozes to Natural Questions

We perform a dependency reconstruction to generate natural questions. We move answer-related words in the dependency tree to the front of the question, since answer-related words are important. The intuition is that natural questions usually start with question words and question focus (Yao and Van Durme, 2014).

As shown in Figure 2, we apply the dependency parsing to the cloze questions, and translate them to natural questions by three steps: i) We keep the right child nodes of the answer and prune its lefts. ii) For each node in the parsing tree, if the subtree of its child node contains the answer node, we move the child node to the first child node. iii) Finally, we obtain the natural question by inorder traversal on the reconstructed tree. We apply the same rule-based mapping as Lewis et al. (2019), which replaces each answer category with the most appropriate wh* word. For example, the `THING` category is mapped to "*What*".

[2]We obtain the answer type labels by a NER toolkit, and group these labels to high-level answer categories, which are used as our mask tokens, e.g., `PRODUCT` corresponding to `THING`, `LOC` corresponding to `PLACE`.

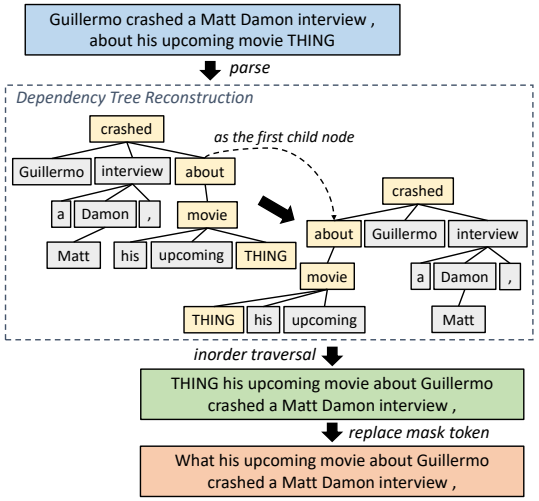

Figure 2: Example of translating cloze questions to natural questions. The node with light yellow color indicates that its subtree contains the answer node.

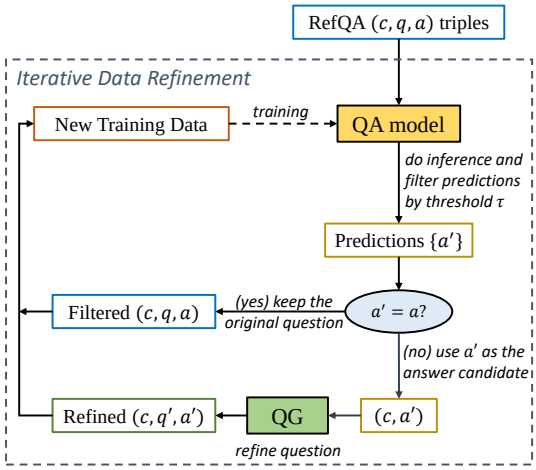

Figure 3: Overview of our iterative data refinement process. "QG" is the process of question generation as described in Section 3.2. We produce new training data and iteratively train the QA model.

4 Iterative Data Refinement

In this section, we propose to iteratively refine data over REFQA based on the QA model. As shown in Figure 3, we use the QA model to filter REFQA data, find appropriate and diverse answer candidates, and use these answers to refine and augment REFQA examples. Filtering data can get rid of some noisy examples in REFQA, and pretrained linguistic knowledge in the BERT-based QA model finds more appropriate and diverse answers. We produce questions for the refined answers, then continue to train the QA model on the refined and filtered triples.

4.1 Initial QA Model Training

The first step of iterative data refinement is to train an initial QA model. We use the REFQA examples $S_I = \{(c_i, q_i, a_i)\}_{i=1}^{N}$ to train a BERT-based QA model $P(a|c, q)$ by maximizing:

$$\sum_{S_I} \log\ P(a_i|c_i, q_i) \tag{1}$$

where the triple consists of context c_i, question q_i, and answer a_i.

4.2 Refine Question-Answer Pairs

As shown in Figure 3, the QA model $P(a|c, q)$ is used to refine the REFQA examples. We first conduct inference on the unseen data (denoted as S_U), and obtain the predicted answers and their probabilities. Then we filter the predicted answers with a confidence threshold τ:

$$Z_A = \{a_i' | P(a_i'|c_i, q_i) \geq \tau\}_{(c_i,q_i,a_i)\in S_U}$$

where a_i' represents the predicted answer.

For each predicted answer a_i', if it agrees with the gold answer a_i, we keep the original question. For the case that $a_i' \neq a_i$, we treat a_i' as our new answer candidate. Besides, we use the question generator (Section 3.2) to refine the original question q_i to q_i'.

In this step, using the QA model for filtering helps us get rid of some noisy examples. The refined question-answer pairs (q_i', a_i') can also augment the REFQA examples. The pretrained linguistic knowledge in the BERT-based QA model is supposed to find more novel answers, i.e., some candidate answers are not extracted by the NER toolkit. With the refined answer spans, we then use the question generator to produce their corresponding questions.

4.3 Iterative QA Model Training

After refining the dataset, we concatenate them with the filtered examples whose candidate answers agree with the predictions. The new training set is then used to continue to train the QA model. The training objective is defined as:

$$\begin{aligned}\max \sum_{a_i'\in Z_A} & [\mathbb{I}(a_i' = a_i)\log P(a_i|c_i, q_i) \\ & + \mathbb{I}(a_i' \neq a_i)\log P(a_i'|c_i, q_i')],\end{aligned} \tag{2}$$

Algorithm 1: Iterative Data Refinement

Input: synthetic context-question-answer triples $\mathcal{S} = \{(c_i, q_i, a_i)\}_{i=1}^{N}$, a threshold τ and a decay factor γ.

Sample a part of triples $\mathcal{S}_I$ from $\mathcal{S}$

Update the model parameters by maximizing $\sum_{\mathcal{S}_I} \log P(a|c, q)$

Split unseen triples into $\{\mathcal{S}_{U_1}, \mathcal{S}_{U_2}, ..., \mathcal{S}_{U_M}\}$

for $k \leftarrow 1$ ***to*** M **do**

  $\mathcal{D} \leftarrow \phi$

  for (c_i, q_i, a_i) ***in*** $\mathcal{S}_{U_k}$ **do**

    $Z_A \leftarrow \{a_i'$ s.t. $P(a_i'|c_i, q_i) \geq \tau\}$

    for a_i' ***in*** Z_A **do**

      if $a_i' = a_i$ **then**

        $\mathcal{D} \leftarrow \mathcal{D} \cup (c_i, q_i, a_i)$

      else

        Refine question q_i to q_i'

        $\mathcal{D} \leftarrow \mathcal{D} \cup (c_i, q_i', a_i')$

  $\tau \leftarrow \tau \times \gamma$

  Update the model parameters by maximizing $\sum_{\mathcal{D}} \log P(a|c, q)$

Output: the updated QA model $P(a|c, q)$

where $\mathbb{I}(\cdot)$ is an indicator function (i.e., 1 if the condition is true).

Using the resulting QA model, we further refine question-answer pairs and repeat the training procedure. The process is repeated until the performance plateaus, or no new data available. Besides, in order to obtain more diverse answers during iterative training, we apply a decay factor γ for the threshold τ. The pseudo code of iterative data refinement is presented in Algorithm 1.

5 Experiments

We evaluate our proposed method on two widely used extractive QA datasets (Rajpurkar et al., 2016; Trischler et al., 2017). We also demonstrate the effectiveness of our approach in the few-shot learning setting.

5.1 Configuration

REFQA Construction We collect the statements with references from English Wikipedia following the procedure in (Zhu et al., 2019a). We only consider the references that are HTML pages, which results in 1.4M statement-document pairs.

In order to make sure the statement is relevant to the cited document, we tokenize the text, remove stop words and discard the examples if more than half of the statement tokens are not in the cited document. The article length is limited to 1,000 words for cited documents. Besides, we compute ROUGE-2 (Lin, 2004) as correlation scores between statements and context. We use the score's median (0.2013) as a threshold, i.e., half of the data with lower scores are discarded. We obtain 303K remaining data to construct our REFQA.

We extract named entities as our answer candidates, using the NER toolkit of Spacy. We split the statements into sub-clauses with Berkeley Neural Parser (Kitaev and Klein, 2018). The questions are generated as in Section 3.2. We also discard sub-clauses that are less than 6 tokens, to prevent losing too much information of original sentences. Finally, we obtain 0.9M REFQA examples.

Question Answering Model We adopt BERT as the backbone of our QA model. Following (Devlin et al., 2019), we represent the question and passage as a single packed sequence. We apply a linear layer to compute the probability of each token being the start or end of an answer span. We use Adam (Kingma and Ba, 2015) as our optimizer with a learning rate of 3e-5 and a batch size of 24. The max sequence length is set to 384. We split the long document into multiple windows with a stride of 128. We use the uncased version of BERT-Large (Whole Word Masking). We evaluate on the dev set every 1000 training steps, and conduct early stopping when the performance plateaus.

Iterative Data Refinement We uniformly sample 300k data from REFQA to train the initial QA model. We split the remaining 600k data into 6 parts for iterative data refinement. For each part, we use the current QA model to refine question-answer pairs. We combine the refined data with filtered data in a 1:1 ratio to continue training the QA model. Specially, we keep the original answer if its prediction is a part of the original answer during inference. The threshold τ is set to 0.15 for filtering the model predictions. The decay factor γ is set to 0.9.

5.2 Results

We conduct evaluation on the SQuAD 1.1 (Rajpurkar et al., 2016), and the NewsQA (Trischler et al., 2017) datasets. We compare our proposed approach with previous unsupervised approaches and several supervised models. Performance is measured via the standard Exact Match (EM) and F1 metrics.

Models	SQuAD 1.1 Dev Set	SQuAD 1.1 Test Set	NewsQA Dev Set	NewsQA Test Set
Supervised Methods				
DCR (Yu et al., 2016)	62.5 / 71.2	62.5 / 71.0	- / -	- / -
mLSTM (Wang and Jiang, 2016)	64.1 / 73.9	64.7 / 73.7	34.4 / 49.6*	34.9 / 50.0*
FastQAExt (Weissenborn et al., 2017)	70.3 / 78.5	70.8 / 78.9	43.7 / 56.1	42.8 / 56.1
R-NET (Wang et al., 2017)	71.1 / 79.5	71.3 / 79.7	- / -	- / -
BERT-Large (Devlin et al., 2019)	84.2 / 91.1	85.1 / 91.8	- / -	- / -
SpanBERT (Joshi et al., 2019)	- / -	88.8 / 94.6	- / -	- / 73.6
Unsupervised Methods				
Dhingra et al. (2018)†	28.4 / 35.8	- / -	- / -	- / -
Lewis et al. (2019)	- / -	44.2 / 54.7	- / -	- / -
Lewis et al. (2019)‡	45.4 / 55.6	- / -	19.6 / 28.5	17.9 / 27.0
Our REFQA	57.1 / 66.8	55.8 / 65.5	29.0 / 42.2	27.6 / 41.0
+ Iterative Data Refinement	**62.5 / 72.6**	**61.1 / 71.4**	**33.6 / 46.3**	**32.1 / 45.1**

Table 1: Results (EM / F1) of our method, various baselines and supervised models on SQuAD 1.1, and NewsQA. "∗" means results taken from Trischler et al. (2017), "†" means results taken from Lewis et al. (2019), and "‡" means our reimplementation on BERT-Large (Whole Word Masking).

Dhingra et al. (2018) propose to train the QA model on the cloze-style questions. Here we take the unsupervised results that re-implemented by Lewis et al. (2019) with BERT-Large. The other unsupervised QA system (Lewis et al., 2019) borrows the idea of unsupervised machine translation (Lample et al., 2017) to convert cloze questions into natural questions. For a fair comparison, we use their published data[3] to re-implement their approach based on BERT-Large (Whole Word Masking) model.

Table 1 shows the main results on SQuAD 1.1 and NewsQA. Training QA model on our REFQA outperforms the previous methods by a large margin. Combining with iterative data refinement, our approach achieves new state-of-the-art results in the unsupervised setting. Our QA model attains 71.4 F1 on the SQuAD 1.1 test set and 45.1 F1 on the NewsQA test set without using their annotated data, outperforming all of the previous unsupervised methods. In particular, the results are competitive with early supervised models.

5.3 Analysis

We conduct ablation studies on the SQuAD 1.1 dev set, in order to better understand the contributions of different components in our method.

[3] https://github.com/facebookresearch/UnsupervisedQA

	Identity	Noise	UNMT	DRC
WIKI	20.8 / 30.5	36.6 / 45.6	40.5 / 49.1	26.3 / 35.7
REFQA	**42.5 / 51.6**	**45.1 / 53.5**	**43.4 / 52.0**	**49.2 / 58.8**

Table 2: Results (EM / F1) of REFQA and WIKI datasets with different cloze translation methods on the SDuAD 1.1 dev set. "DRC" is short for dependency reconstruction.

5.3.1 Effects of REFQA

We conduct experiments on REFQA and another synthetic dataset (named as WIKI). The WIKI dataset is constructed using the same method as in Lewis et al. (2019), which uses Wikipedia pages as context paragraphs for QA examples. In addition to the dependency reconstruction method (Section 3.2.2), we compare three cloze translation methods proposed in Lewis et al. (2019).

Identity Mapping generates questions by replacing the mask token in cloze questions with a relevant wh* question word.

Noise Cloze first applies a noise model, such as permutation, and word drop, as in Lample et al. (2017), and then applies the "Identity Mapping" translation.

UNMT converts cloze questions into natural questions following unsupervised neural machine translation. Here we directly use the published model of Lewis et al. (2019) for evaluation.

it finished first in the Arbitron ratings in April 1990	he was sold to Colin Murphy's Lincoln City for a fee of 15,000
UNMT: Who finished it first in the ratings in April 1990 ? **DRC:** Who ratings in it finished first in April 1990	**UNMT:** How much do we need Colin Murphy 's Lincoln City for a fee ? **DRC:** How much of a fee for he was sold to Colin Murphy 's Lincoln City

Table 3: Examples of generated questions using UNMT and our method. "DRC" is short for our dependency reconstruction. The blue words indicate extracted answers.

	Iter.	Size	EM / F1
Initial QA Model		300k	57.1 / 66.8
Training on			
Filtered Data	✗	464k	57.4 / 67.1
Refined Data	✗	100k	61.0 / 70.7
Refined + Filtered Data	✗	200k	61.8 / 71.0
Refined Data	✓	6×15k	60.1 / 70.0
Refined + Filtered Data	✓	6×30k	**62.5 / 72.6**

Table 4: Results of using filtered data, refined data, and the combination for data refinement on the SDuAD 1.1 dev set. "Iter." is short for iterative training.

For a fair comparison, we sample 300k training data for each dataset, and fine-tune BERT-Base for 2 epochs. As shown in Table 2, training on our REFQA achieves a consistent gain over all cloze translation methods. Moreover, our dependency reconstruction method is also favorable compared with the "Identity Mapping" method. The improvement of DRC on WIKI is smaller than on REFQA. We argue that it is because WIKI contains too many lexical overlaps, while DRC mainly focuses on providing structural diversity.

We present the generated questions of our method (DRC) and UNMT in Table 3. Most natural questions follow a similar structure: question word (what/who/how), question focus (name/-money/time), question verb (is/play/take) and topic (Yao and Van Durme, 2014). Compared with UNMT, our method adjusts answer-related words in the dependency tree according to the linguistic characteristics of natural questions.

5.3.2 Effects of Data Combination

We validate the effectiveness of combining refined and filtered data for our data refinement. We use only refined or filtered data to train our QA model, comparing with the combining approach.

The results are shown in Table 4. We observe that both data can help the QA model to achieve better performance. Moreover, the combination of refined and filtered data is more useful than only using one of them. Using iterative training, our combination approach further improves the model performance to 72.6 F1 (1.6 absolute improvement). Besides, using our refined data contributes further improvement compared with filtered data.

τ	0.0	0.1	0.15	0.2	0.3	0.5	0.7
EM	54.3	61.2	61.8	61.1	59.7	59.2	58.5
F1	69.6	70.4	71.0	70.9	69.4	68.7	67.7

Table 5: Results of using different confidence thresholds during the construction of the refined data and filtered data.

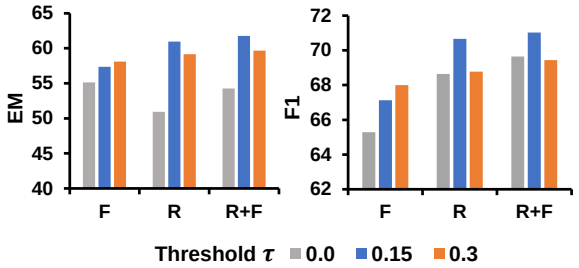

Figure 4: Comparison on filtered data and refined data with different confidence thresholds. "F" is short for using filtered data, "R" is short for using refined data. "R+F" is short for the combination of refined and filtered data.

5.3.3 Effects of Confidence Threshold

We experiment with several thresholds (0.0, 0.1, 0.15, 0.2, 0.3, 0.5 and 0.7) to filter the predicted answers. Their QA results on SQuAD 1.1 dev set are presented in Table 5. Using threshold of 0.15 achieves better performance.

We also analyze the effects of threshold on refined data and filtered data. As shown in Figure 4, for the filtered data, using a higher confidence threshold achieves better performance, suggesting that using the QA model for filtering makes our examples more credible. For the refined data and the combination, we observe that the threshold 0.15 achieves a better performance than the threshold 0.3, but the EM is greatly reduced when the threshold is set to 0.0. Besides, there are 26,257 answers that do not appear in named entities using the threshold 0.15, compared to 15,004 for the threshold 0.3. Thus, an appropriate threshold can help us improve the answer diversity and get rid of some noisy examples.

	Refined	Size	EM / F1
REFQA	-	300k	57.1 / 66.8
OA⊃PA	✗	90k	59.4 / 69.0
OA⊃PA	✓	90k	50.9 / 64.6
OA⊂PA	✗	35k	47.5 / 61.2
OA⊂PA	✓	35k	60.3 / 69.9
Others	✗	75k	52.2 / 62.3
Others	✓	75k	58.8 / 69.7

Table 6: Comparison between different types of data refinement on the SQuAD 1.1 dev set.

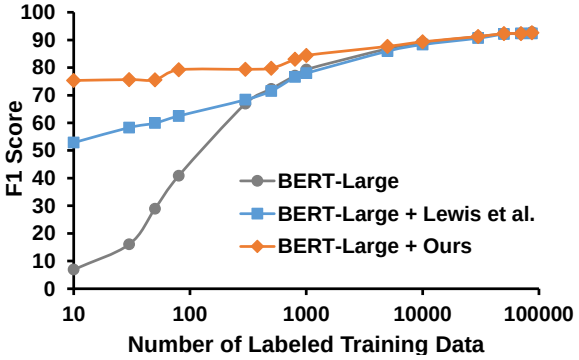

Figure 5: F1 score on the SQuAD 1.1 dev set with various training dataset sizes.

5.3.4 Effects of Refinement Types

For brevity, we denote the original answer and predicted answer by "OA" and "PA", respectively. In order to analyze the contribution of our refined data, we categorize the data refinements into the following three types:

OA⊃PA The original answer contains the predicted answer.

OA⊂PA The predicted answer contains the original answer.

Others The remaining data except for the above two types of refinement.

For each type, we keep the original data or use refined data to train our QA model. We conduct experiments on the non-iterative setting with the data combination.

As shown in Table 6, our refined data improves the QA model in most types of refinement except "OA⊃PA". The results indicate that the QA model favors longer phrases as answer spans. Moreover, for the "OA⊂PA" and "Others" types, there are 47.8% answers that are not extracted by the NER toolkit. The iterative refinement extends the category of answer candidates, which in turn produces novel question-answer pairs.

We show a few examples of our generated data in Table 7. We list one example for each type. For the "OA⊃PA" refinement, the predicted answer is a sub-span of the extracted named entity, but the complete named entity is more appropriate as an answer. For the "OA⊂PA" refinement, the QA model can help us extend the original answer to be a longer span, which is more complete and appropriate. Besides, for the "Others" refinement, its prediction can be a new answer, and not appear in named entities extracted by the NER toolkit.

5.4 Few-Shot Learning

Following the evaluation of (Yang et al., 2017; Dhingra et al., 2018), we conduct experiments in a few-shot learning setting. We use the best configuration of our approach to train the unsupervised QA model based on BERT-Large (Whole Word Masking). Then we fine-tune the model with limited SQuAD training examples.

As shown in Figure 5, our method obtains the best performance in the restricted setting, compared with the previous state of the art (Lewis et al., 2019) and directly fine-tuning BERT. Moreover, our approach achieves 79.4 F1 (16.4 absolute gains than other models) with only 100 labeled examples. The results illustrate that our method can greatly reduce the demand of in-domain annotated data. In addition, we observe that the results of different methods become comparable when the labeled data size is greater than 10,000.

6 Conclusion

In this paper, we present two approaches to improve the quality of synthetic QA data for unsupervised question answering. We first use the Wikipedia paragraphs and its references to construct a synthetic QA data REFQA and then use the QA model to iteratively refine data over REFQA. Our method outperforms the previous unsupervised state-of-the-art models on SQuAD 1.1, and NewsQA, and achieves the best performance in the few-shot learning setting.

Acknowledgements

The work was partially supported by National Natural Science Foundation of China (NSFC) [Grant No. 61421003].

OA⊃PA	**S:** In 1938, E. Allen Petersen escaped the advancing Japanese armies by sailing a junk, "Hummel Hummel", from Shanghai to California with his wife Tani and two White Russians (Tsar loyalists). **Q:** Who escaped the advancing Japanese armies by sailing a junk **OA:** E. Allen Petersen **PA:** Petersen **RQ:** Who escaped the advancing Japanese armies by sailing a junk
OA⊂PA	**S:** Hyundai announced they would be revealing their future rally plans at the 2011 Chicago Auto Show on February 9 . **Q:** What at they would be revealing their future rally plans on February 9 **OA:** Chicago Auto Show **PA:** the 2011 Chicago Auto Show **RQ:** What at their future rally plans they would be revealing on February 9
Others	**S:** In January 2017, she released the track "That's What's Up" that re-imagines the spoken word segment on the Kanye West song "Low Lights". **Q:** What the Kanye West song on the spoken word segment re-imagines **OA:** Low Lights **PA:** That's What's Up **RQ:** What the track she released that re-imagines the spoken word segment on the Kanye West song "Low Lights" .

Table 7: The generated and refined question-answer pairs. "S" and "Q" are short for statement and question. "OA", "PA" and "RQ" are short for the original answer, predicted answer and the refined question.

References

Chris Alberti, Daniel Andor, Emily Pitler, Jacob Devlin, and Michael Collins. 2019. Synthetic QA corpora generation with roundtrip consistency. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6168–6173, Florence, Italy. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Bhuwan Dhingra, Danish Danish, and Dheeraj Rajagopal. 2018. Simple and effective semi-supervised question answering. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 582–587, New Orleans, Louisiana. Association for Computational Linguistics.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2019. SpanBERT: Improving pre-training by representing and predicting spans. *arXiv preprint arXiv:1907.10529*.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations*, San Diego, CA.

Nikita Kitaev and Dan Klein. 2018. Constituency parsing with a self-attentive encoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2676–2686, Melbourne, Australia. Association for Computational Linguistics.

Guillaume Lample, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2017. Unsupervised machine translation using monolingual corpora only. *ArXiv*, abs/1711.00043.

Patrick Lewis, Ludovic Denoyer, and Sebastian Riedel. 2019. Unsupervised question answering by cloze translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4896–4910, Florence, Italy. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable ques-

tions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. *ArXiv*, abs/1611.01603.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. NewsQA: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 191–200, Vancouver, Canada. Association for Computational Linguistics.

Shuohang Wang and Jing Jiang. 2016. Machine comprehension using match-lstm and answer pointer. *ArXiv*, abs/1608.07905.

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

Dirk Weissenborn, Georg Wiese, and Laura Seiffe. 2017. Making neural QA as simple as possible but not simpler. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 271–280, Vancouver, Canada. Association for Computational Linguistics.

Zhilin 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)*, pages 1040–1050, Vancouver, Canada. Association for Computational Linguistics.

Xuchen Yao and Benjamin Van Durme. 2014. Information extraction over structured data: Question answering with Freebase. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 956–966, Baltimore, Maryland. Association for Computational Linguistics.

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V. Le. 2018. QANet: Combining local convolution with global self-attention for reading comprehension. *ArXiv*, abs/1804.09541.

Yang Yu, Wei Zhang, Kazi Saidul Hasan, Mo Yu, Bing Xiang, and Bowen Zhou. 2016. End-to-end reading comprehension with dynamic answer chunk ranking. *ArXiv*, abs/1610.09996.

Haichao Zhu, Li Dong, Furu Wei, Bing Qin, and Ting Liu. 2019a. Transforming wikipedia into augmented data for query-focused summarization. *ArXiv*, abs/1911.03324.

Haichao Zhu, Li Dong, Furu Wei, Wenhui Wang, Bing Qin, and Ting Liu. 2019b. Learning to ask unanswerable questions for machine reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4238–4248, Florence, Italy. Association for Computational Linguistics.