OneStop QAMaker: Extract Question-Answer Pairs from Text in a One-Stop Approach

Shaobo Cui
yuanchun.csb@alibaba-inc.com
DAMO Academy, Alibaba Group

Xintong Bao
xintong.bxt@alibaba-inc.com
DAMO Academy, Alibaba Group

Xinxing Zu
patrick.zxx@alibaba-inc.com
DAMO Academy, Alibaba Group

Yangyang Guo
guoyang.eric@gmail.com
Shandong University

Zhongzhou Zhao
zhongzhou.zhaozz@alibaba-inc.com
DAMO Academy, Alibaba Group

Ji Zhang
zj122146@alibaba-inc.com
DAMO Academy, Alibaba Group

Haiqing Chen
haiqing.chenhq@alibaba-inc.com
DAMO Academy, Alibaba Group

ABSTRACT

Large-scale question-answer (QA) pairs are critical for advancing research areas like machine reading comprehension and question answering. To construct QA pairs from documents requires determining how to ask a question and what is the corresponding answer. Existing methods for QA pair generation usually follow a pipeline approach. Namely, they first choose the most likely candidate answer span and then generate the answer-specific question. This pipeline approach, however, is undesired in mining the most appropriate QA pairs from documents since it ignores the connection between question generation and answer extraction, which may lead to incompatible QA pair generation, i.e., the selected answer span is inappropriate for question generation. However, for human annotators, we take the whole QA pair into account and consider the compatibility between question and answer. Inspired by such motivation, instead of the conventional pipeline approach, we propose a model named OneStop generate QA pairs from documents in a one-stop approach. Specifically, questions and their corresponding answer span is extracted simultaneously and the process of question generation and answer extraction mutually affect each other. Additionally, OneStop is much more efficient to be trained and deployed in industrial scenarios since it involves only one model to solve the complex QA generation task. We conduct comprehensive experiments on three large-scale machine reading comprehension datasets: SQuAD, NewsQA, and DuReader. The experimental results demonstrate that our OneStop model outperforms the baselines significantly regarding the quality of generated questions, quality of generated question-answer pairs, and model efficiency.

CCS CONCEPTS

• **Information systems** → **Question answering**; • **Computing methodologies** → **Natural language generation**.

KEYWORDS

Question generation, Question-Answer pair generation, OneStop approach, Multi-task learning

1 INTRODUCTION

Many tasks in the natural language processing community such as machine reading comprehension and question answering [12, 16, 28] rely heavily on large amounts of human-labeled question-answer (QA) pairs. However, manually annotating QA pairs by human [2, 16, 28] is both costly and time-consuming. Recently, how to automatically extract QA pairs from documents has attracted increasing attention.

The task of QA pair extraction from a document d is to extract the most related QA pair: $\arg\max_{q,a} P(q, a|d)$. Most of existing works [1, 7, 29, 36, 38] adopt a pipeline approach, in which firstly selects candidate answer spans from the document: $\arg\max_a P(a|d)$, and then generate the answer-specific questions: $\arg\max_q P(q|d, a)$. We present the simplified view of the pipeline approach in Figure 1. This type of pipeline approach, however, suffers two major drawbacks. Firstly, There is no explicit correlation between the question generation and the answer extraction process. Namely, the question generation model and the answer extraction model are **isolated** during their training process. This isolation leads the extracted

Table 1: Instances to illustrate QA's incompatibility.

	Type	Utterance
Example 1	Approach	First generate a question and then find the generated questions' answer span in document.
	Incompatibility type	Generate a question that is hard to find its answer in document by the answer extraction model.
	Document	The delta is delimited in the West by the in the East by a modern canalized section.
	Question	What does delta look like?
	Answer	−−
Example 2	Approach	First predict a candidate answer span and then generate an answer-aware question
	Incompatibility type	The predicted answer span is too detailed, incorrect or unsuitable for answer-aware question generation.
	Document	The French crown's refusal to allow non-Catholics to settle in New France may help to explain that colony's slow rate of population growth compared to that of the neighbouring British colonies, which opened settlement to religious dissenters.
	Answer Span	may help to
	Question	−−

QA pairs to be **incompatible**: the question generation model may generate questions that are hard to find their corresponding answers by the answer extraction model(see example 1 in Table 1),

or the answer extraction may extract answer span that are not suitable for question generation(see example 2 in Table 1). This incompatibility can be explained by Figure 1. These two separate steps: { $\arg\max_a P(a|d)$, $\arg\max_q P(q|d, a)$} are not an accurate approximation for $\arg\max_{q,a} P(q, a|d)$. Secondly, this type of pipeline methods are knotty and time-consuming to be trained and deployed in the industrial online application since they involve at least two models and the cumulative error along the pipeline is huge.
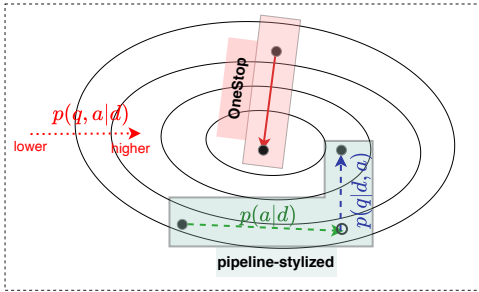

Figure 1: A simplified view of the pipeline approach to QA pair generation. The QA pair under consideration is denoted as a black dot. In the pipeline settings, the expected QA pair is firstly pushed in the direction of maximizing $P(a|d)$ and then in the *conditional* direction of maximizing $P(q|d, a)$. OneStop approach, however, optimizes in the direction of maximizing $P(q, a|d)$ straightforwardly.

Unlike the aforementioned pipeline approach, human annotators usually take the whole QA pair into consideration and pay close attention to the compatibility between the extracted answer and the generated question. More specifically, from human annotators' perspective, a question that is less likely to be answered by referring to the given document should not be generated in the question generation process. Similarly, an answer whose corresponding question is inferior or unsuitable for question generation should be given less attention in the answer extraction process. In a nutshell, human annotators consider the compatibility and overall quality of QA pairs. Inspired by the limitation of existing pipeline methods and the aforementioned motivation for QA's compatibility, we integrate the question generation and the answer extraction into a unified framework to enhance the compatibility of generated question and the extracted answer. We propose OneStop, an architecture which can be easily adapted from existing pre-trained language models to extract QA pairs from documents in a OneStop approach. The OneStop model takes documents as input and outputs questions q and questions' corresponding answer spans a. The answer extraction and the question generation module in the OneStop model collaborate together to find the most compatible QA pairs. Specifically, our OneStop model tackles the objective $\arg\max P(q, a|d)$ directly instead of the decomposed objectives: {$\arg\max P(a|d)$, $\arg\max P(q|d, a)$}. These two tasks in our OneStop model mutually affect each other: (1) the answer span extraction task pushes the question generation model to generate more answerable questions since it is hard to extract the answer span of an unanswerable question; (2) the question generation task could further enhance the answer extraction model by providing the probability of generating a question. Specifically,

the answer extraction model places more attention on questions favored by the question generation model, i.e., the question whose $P(q|d)$ is large. Additionally, by combining the question generation model and answer extraction model in one single model, our OneStop model is much lighter than the existing pipeline approach that involves at least two models.

As for the model structure, OneStop model adopts the conventional transformer-based sequence-to-sequence structure. Our OneStop model can be easily built upon pre-trained model such as BART [21], T5 [27], ProphetNet [26] and so on. The training objective of the OneStop model is to generate a suitable question and predict the right answer span for this question simultaneously. To verify the effectiveness of our OneStop model, we conduct experiment on three large-scale datasets: SQuAD [28], NewsQA [34], and DuReader [10]. We compare the involved baselines in terms of the quality of generated questions, the quality of QA pairs, and model efficiency. Experimental results prove that our OneStop model achieves SOTA performance in a more efficient way. The contributions of this paper are summarized as follows:

(1) We propose a unified framework in which the answer extraction module and the question generation module could mutually enhance each other.
(2) To our best knowledge, OneStop is the first transformer-based model for generating more compatible QA pairs from documents in a one-stop approach.
(3) OneStop can be easily built upon existing pre-trained language models. Compared with previous pipeline approaches, our OneStop model is much more efficient to train and deploy in industrial scenarios and requires much less human effort.
(4) We conduct comprehensive experiments on three large-scale datasets to evaluate our OneStop model in terms of question generation, QA pair generation and model efficiency.

2 RELATED WORK

Question Generation. Question generation [3, 17, 24, 31, 32, 39, 40] is a well-studied natural language processing task in literature. There are mainly two types of approaches for question generation: template-based and model-based. Methods [11, 19] in the first category rely on human efforts to design the template rule and are thus unscalable across datasets. In contrast, the model-based methods [39, 40] employ an end-to-end neural network to generate questions, which takes as inputs selected key phrases and documents. However, these methods are limited as the questions cannot be generated from documents directly. An additional entity extraction model or a sequence labeling model [31, 36] is required to determine which part of the document is worthy of being asked. As a result, it is less practical for this kind of methods in question generation due to the following two facts: (1) the key phrase extraction model demands addition manual labor and elaborated tuning; (2) The most question-worthy phrases in a document are difficult to be identified.

Question-Answer Pair Generation. Most of existing works [1, 8, 15, 18, 20, 23] focusing on the QA pair generation follow a pipeline fashion: (1) determine what points in the document should be asked; (2) Learn to ask based on the selected points; (3) Detect the answer span of the question in the document; Du and Cardie

firstly detected the question-worthy answer (which they dubbed as answer span identification) and then generated the answer-aware question. Similarly, Golub et al. proposed a two-stage SynNet for QA pair generation, which consists of an answer tagging module and a question synthesis module. Alberti et al. proposed to generate QA pairs with models of question generation and answer extraction and then filtered the results with roundtrip consistency.

Joint Models for Question Generation and Question Answering. There have been studies [5, 30, 33, 37] focusing on solving question generation and question answering together. In these methods, the input and output of question generation and question answering are inverse, which makes them dual tasks. In this way, question generation and question answering are implemented with separate models connected by their duality. However, the training objective of question answering poses an adverse effect on the performance of the question generation model due to the enforcement of dual constraint. Our work is different from these work [5, 30, 33, 37], which focus on the duality of question generation and question answering. Firstly, for QA extraction from documents, there is not an explicit duality between question generation and answer extraction. Consequently, the duality between these two tasks no longer exists. Secondly, question generation and answer extraction in OneStop are optimized in a multi-task learning approach. Namely, they are optimized simultaneously to find a compatible and optimal solution for QA pair generation.

3 PROBLEM DEFINITION

Given a document, the objective of QA pair generation is to find the related QA pairs. Mathematically:

$$\bar{q}, \bar{a} = \arg\max_{q,a} P(q, a|d), \tag{1}$$

where document d is a sequence of utterances; answer a should be a sub span from the document and question q is an utterance that is closely associated with a. Based on this formulation, existing methods can be classified into the following two groups:

(a) D2A2Q
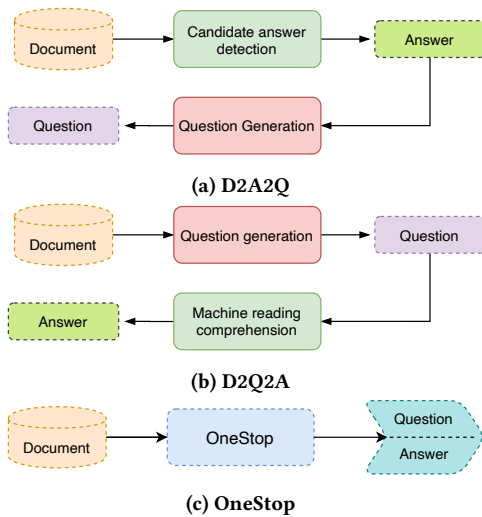
(b) D2Q2A

(c) OneStop

Figure 2: The comparison of D2A2Q, D2Q2A ,and OneStop.

(1) **D2A2Q**: The candidate answer is first extracted from the document: $P(a|d)$, after which the answer-specific question is generated based on the document and the extracted candidate answer: $P(q|d, a)$. It can be summarized as:

$$\arg\max_{q,a} P(q, a|d) \approx \begin{cases} \arg\max_a P(a|d; \theta_{\text{d2a}}), & \text{Step I} \\ \arg\max_q P(q|d, a; \theta_{\text{da2q}}), & \text{Step II} \end{cases} \tag{2}$$

where θ_{d2a} and θ_{da2q} are the parameters of candidate answer extraction model and answer-specific question generation model respectively.

(2) **D2Q2A**: It firstly generates question that is most likely to be asked from the document, i.e., $P(q|d; \theta_{\text{d2q}})$. And then the generated question is utilized to extract its corresponding answer span from the document. Similarly, D2Q2A approach can be summarize as:

$$\arg\max_{q,a} P(q, a|d) \approx \begin{cases} \arg\max_q P(q|d; \theta_{\text{d2q}}) & \text{Step I} \\ \arg\max_a P(a|d, q; \theta_{\text{dq2a}}) & \text{Step II} \end{cases} \tag{3}$$

where θ_{d2q} and θ_{dq2a} are parameters of question generation model and answer extraction (machine reading comprehension) model respectively.

The aforementioned pipeline approaches such as D2Q2A and D2A2Q are all quite rough approximation to the original objective of $\arg\max_{q,a} P(q, a|d)$ [1]. The cumulative error is magnified along these pipelines. Additionally, the training cost and inference efficiency are unfavorable. Motivated by these limits, we propose the OneStop model that models the objective much more precisely. The OneStop framework can be formulated as:

$$\arg\max_{q,a} P(q, a|d) = \arg\max_{q,a} P(q|d; \theta) \cdot P(a|d, q; \theta) \tag{4}$$

where θ is the parameters of the OneStop model. The answer extraction module and the question generation module in the OneStop model share the model parameters θ, which means these two tasks influence each other. As can be observed, our OneStop model is easier to train and more efficient to use during inference since it involves only one model. We present the comparison of these three different approaches in Figure 2. As we can see, both D2A2Q and D2Q2A are pipeline approaches. Nevertheless, our OneStop model tackle the original QA pair generation objective directly.

4 ONESTOP MODEL

In this section, we firstly present the overview of the OneStop model in Section 4.2. Section 4.3 and Section 4.4 are about the question generation and answer span extraction module of the OneStop model respectively, after which we end this section with the training and inference of OneStop model in Section 4.5.

4.1 Preliminary: Self-Attentive Module

Inspired by the superiority of transformer [35] in utterance representation, we adopt the self-attentive unit as the basic unit of encoder and decoder in our OneStop model. As shown in Figure 3, each self-attentive unit consists of a self-attention layer and a position-wise fully connected feed-forward layer. Each of these two layers

---

[1]For D2Q2A, the final output along the pipeline $\arg\max_q P(q|d)$, $\arg\max_q P(a|d, q)$ are unlikely to be the optimal solution for $\arg\max_{q,a} P(q, a|d)$. A similar conclusion can be obtained for D2A2Q.

is employed with residual connection, followed by layer normalization. More specifically, the whole computation process in the self-attentive module can be summarized as:

$$\text{Att}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Softmax}(\frac{\mathbf{Q}\mathbf{K}^{\mathsf{T}}}{\sqrt{d_k}})\mathbf{V}$$

$$\mathbf{X}_1 = f_{\text{norm}}(\mathbf{Q} + \text{Att}(\mathbf{Q}, \mathbf{K}, \mathbf{V}))$$

$$\mathbf{X}_2 = \max(0, \mathbf{X}_1\mathbf{W}_1 + \mathbf{b}_1)\mathbf{W}_2 + \mathbf{b}_2$$

$$f_{\text{att}}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = f_{\text{norm}}(\mathbf{X}_1 + \mathbf{X}_2)$$

where d_k is the model dimension. $f_{\text{att}}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) \in \mathbb{R}^{t \times d_k}$, where t is the input length, $\mathbf{Q} \in \mathbb{R}^{t \times d_k}$, $\mathbf{K} \in \mathbb{R}^{t \times d_k}$ and $\mathbf{V} \in \mathbb{R}^{t \times d_k}$. $\mathbf{W}_1, \mathbf{b}_1, \mathbf{W}_2$ and $\mathbf{b}_2$ are learnable model parameters. The computation process of self-attentive module is shown in Figure 3.
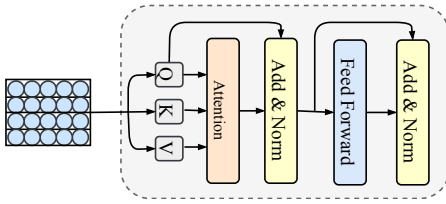

Figure 3: Self-attentive module.

4.2 Model Overview

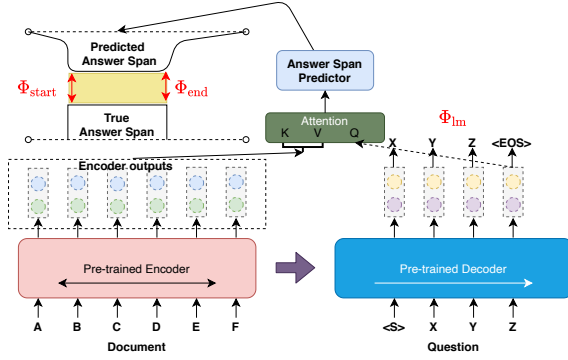The overview of our OneStop model is presented in Figure 4. Our

Figure 4: Overview of our proposed OneStop model.

OneStop model uses the canonical sequence-to-sequence transformer [35] architecture. Here we take BART model as an example to illustrate our model structure. Note that our OneStop approach can be easily modified from other pre-trained language models such as T5 [27] or ProphetNet [26]. OneStop model consists of a bidirectional encoder and an auto-regressive decoder. The encoder in the OneStop model takes the document as input, and each decoder layer performs cross attention over the final hidden layer of the encoder's outputs. The decoder decodes the generated question in an auto-regressive approach. The start and end position of answer span are predicted based on the encoder outputs and the decoder's outputs at <eos> position. This can be explained by the fact that

the corresponding answer for a generated question is determined by document and question together.

4.3 Question Generation

As described above, the input of the encoder is the document d and the output of the decoder is expected to be the question q. The cross entropy loss for question generation is denoted as:

$$\Phi_{\text{lm}} = -\sum_{t=1}^{|q|} \log P(q_t | q_{<t}, d; \theta), \tag{5}$$

where $|q|$ is the length of the question and $P(q_t | q_{<t}, d; \theta)$ is the predicted probability for token q_t. After we obtain the generated question, we use the decoder outputs at <eos> position as the representation of the generated question, which is denoted as $\mathbf{q}_{\text{eos}}$. All the outputs of encoders is denoted as: $\mathbf{D}^e = \{\mathbf{D}_1^e, \mathbf{D}_2^e, \cdots, \mathbf{D}_{|d|}^e\}$, where $\mathbf{D}^e \in \mathbb{R}^{|d| \times m}$ and m is the model dimension. $\mathbf{D}_i^e$ represents the encoder outputs at position i and $|d|$ is the length of the input document. Compared with RNN-based models like GRU [4] or LSTM [13], the self-attentive module endows us the advantage of encoding each token in a given utterance at the same time.

4.4 Answer Span Prediction

Given the generated question and the document, answer span extraction aims to predict the answer span in the document that could answer the generated question properly. More specifically, it should predict the start and end position of the ground-truth answer span. The label prediction network takes $\mathbf{D}^e$ and $\mathbf{q}_{\text{eos}}$ as inputs. The probabilities of each token being the start label and the end label are as:

$$\begin{cases} P_{\text{start}}(i) &= \frac{\exp(\mathbf{D}_i^e \mathbf{W}_s \mathbf{q}_{\text{eos}})}{\sum_i \exp(\mathbf{D}_i^e \mathbf{W}_s \mathbf{q}_{\text{eos}})} \\ P_{\text{end}}(i) &= \frac{\exp(\mathbf{D}_i^e \mathbf{W}_e \mathbf{q}_{\text{eos}})}{\sum_i \exp(\mathbf{D}_i^e \mathbf{W}_e \mathbf{q}_{\text{eos}})} \end{cases}, \tag{6}$$

where $\mathbf{W}_s, \mathbf{W}_e$ are learnable parameters. The cross entropy loss for the start and the end label prediction are:

$$\begin{cases} \Phi_{\text{start}} &= -\log P_{\text{start}}(a_{\text{start}} | d, q; \theta) \\ \Phi_{\text{end}} &= -\log P_{\text{end}}(a_{\text{end}} | d, q; \theta) \end{cases}, \tag{7}$$

where a_{start} and a_{end} are true positions of the answer's start and end.

4.5 Training and Inference of OneStop Model

Training of OneStop Model As introduced in Equation 4, we have:

$$P(q, a | d; \theta) = P(q | d; \theta) \cdot P(a | d, q; \theta)$$

$$= \left(\prod_{t=1}^{|q|} P(q_t | q_{<t}, d; \theta) \right) \cdot \left(P_{\text{start}}(a_{\text{start}} | d, q; \theta) \cdot P_{\text{end}}(a_{\text{end}} | d, q; \theta) \right) \tag{8}$$

The negative log-likelihood of OneStop model can be expressed:

$$\begin{aligned} \Phi = & -\log P(q, a | d; \theta) \\ = & -\sum_{t=1}^{|q|} \log P(q_t | q_{<t}, d; \theta) - \log P_{\text{start}}(a_{\text{start}} | d, q; \theta) \\ & -\log P_{\text{end}}(a_{\text{end}} | d, q; \theta) \\ = & \Phi_{\text{lm}} + \Phi_{\text{start}} + \Phi_{\text{end}} \end{aligned} \tag{9}$$

We use a generalization of OneStop objective which introduces a hyperparameter λ that balance question generation and answer extraction:

$$\Phi = \lambda \cdot \Phi_{lm} + (1 - \lambda) \cdot (\Phi_{start} + \Phi_{end}) \qquad (10)$$

Training Algorithm   The training algorithm of the OneStop model is described in Algorithm 1.

Algorithm 1: Training algorithm of OneStop model.

Input   : (d, q, a) triples, a pre-trained BART language model.

Output : OneStop model which takes the document as input and outputs QA pairs.

1 Load the pre-trained BART model as the initial checkpoint of generation parts of the OneStop model;

2 Fine-tune the question generation part of the OneStop model with $\gamma = 1$ in Equation (10), i.e., $\Phi = \Phi_{lm}$;

3 Fine-tune the answer prediction part of the OneStop model with $\gamma = 0$ in Equation (10), i.e., $\Phi = \Phi_{start} + \Phi_{end}$;

4 Determine the value of γ;

5 **while** *not converge* **do**

6 | Fine-tune the OneStop model with
   | $\Phi = \lambda \cdot \Phi_{lm} + (1 - \lambda) \cdot (\Phi_{start} + \Phi_{end})$;

7 **end**

Inference of OneStop Model   In the inference phase, we feed the document into the OneStop's encoder, the question is generated from OneStop's decoder in an auto-regressive approach. The start and end position of the answer span are predicted by the answer span predictor network. With the start and end position, we can obtain the answer span in the document for the generated question. OneStop model also supports for generating multiple QA pairs for long documents, i.e., the long document could be split as multiple sub-documents and OneStop could generate the most related QA pairs for each sub-document.

5 EXPERIMENT SETUP

In this section, we mainly elaborate the datasets, involved baselines, evaluation metrics, and model settings sequentially.

5.1 Datasets

In this paper, we conducted experiments on three large-scale machine reading comprehension datasets to evaluate the performance of our proposed OneStop model.

- SQuAD [28]: SQuAD consists of questions posed by crowdworkers on Wikipedia articles, and the corresponding answer is a subspan of the corresponding articles.
- NewsQA [34]: the documents in NewsQA are articles collected from CNN news. Similar to SQuAD, questions are acquired through crowd-sourcing while the answer is a subspan of documents.
- DuReader [10]: DuReader is an open-domain machine reading comprehension dataset in which questions are collected from real anonymized user queries. The documents and the answers are acquired using the search engine.

Besides, the answer should be subspan of the corresponding document. In this setting, we filtered out the data item in DuReader

whose answer is not part of the document. The QA pair associated with one document should be unique. However, for SQuAD and NewsQA, one long document may have more than one QA pair. For this reason, we split the long document into multiple sub-document to ensure that each sub-document contain only one QA pairs. We list the statistics of the modified datasets in Table 2.

Table 2: The statistics of the filtered datasets.

	SQuAD	NewsQA	DuReader
# Training	59,819	37,688	198,532
# Validation	1,127	1,412	1,145
# Test	3,000	3,000	6,000
Avg. len. of document	27.20	29.63	144.32
Avg. len. of question	10.18	6.57	9.65
Avg. len. of answer	3.28	5.22	86.54

5.2 Baselines

To evaluate our proposed OneStop model's performance, we compare our model with two types of baselines. The first type is the models for question generation, which is to evaluate the quality of generated questions. The second type is about QA pair generation baselines, which is to evaluate the quality of generated QA pairs.

Baselines for Question Generation   We used the following models as the baselines for the evaluation of question generation.

- DeepNQG: the neural question generation model proposed in [8], an end-to-end model implemented with GRU module.
- CRF-DeepNQG: we followed the conventional setting in the D2A2Q approach, which firstly selects the most likely answer span from the document and then utilizes the extracted answer span and the document to generate a question. The answer extraction (AE) is defined as a sequence tagging task implemented with a BiLSTM-CRF model [7, 14]. The embedding of the document and the extracted answer are concatenated together to generate the answer-specific question. If the answer extraction model predicts more than one answer tag, we randomly selected one span from the span set as the answer to generate the question. If no answer tag is predicted, we viewed the whole document as the selected answer span.
- BART-QG: a fine-tuned model from a pretrained BART [21] model on the question generation task, whose input is document and output is question.
- BART-A2QG: a fine-tuned model from a pretrained BART [21] model, whose input is answer and output is question. This model is to explore the utility of answer directly in question generation.

Baselines for Question-Answer Pair Generation: Pipeline vs. OneStop   For pipeline QA pair generation methods, we utilized the aforementioned question generation model for question generation. To obtain the corresponding answer to the generated question, we chose BERT [6] as the answer extraction model. In this setting, we have the following QA pair generation approaches:

- **Existing Pipeline Approach**
  - DeepNQG + BERT-MRC: the DeepNQG model for question generation and the BERT model for answer extraction.
  - CRF-DeepNQG + BiLSTM-CRF: as described above, the answer extraction model is implemented with a BiLSTM-CRF model.

Table 3: The comparison of baselines on question generation.

Models	SQuAD					NewsQA					DuReader				
	BLEU-1	BLEU-2	Rouge-1	Rouge-2	Rouge-L	BLEU-1	BLEU-2	Rouge-1	Rouge-2	Rouge-L	BLEU-1	BLEU-2	Rouge-1	Rouge-2	Rouge-L
DeepNQG	17.49	8.81	17.54	4.53	17.77	14.30	6.22	14.64	2.81	14.79	3.14	1.72	4.67	1.32	4.72
CRF-DeepNQG	19.61	9.68	19.10	4.74	18.92	17.06	7.93	17.07	3.73	17.11	0.70	0.53	7.66	4.10	7.76
BART-QG	**31.36**	**21.25**	32.65	14.62	29.04	**22.30**	**13.48**	23.32	**8.34**	21.81	45.22	38.33	**47.59**	33.53	43.13
BART-A2QG	20.85	10.51	21.50	5.50	18.81	21.53	11.81	23.29	6.96	21.75	40.61	33.94	42.85	29.32	38.58
OneStop	**31.32**	**21.28**	**32.77**	**14.79**	**29.10**	**22.28**	**13.46**	**23.39**	**8.39**	**21.90**	**45.19**	**38.35**	**47.56**	**33.59**	**43.16**

The BiLSTM-CRF model's tagged phrase is used as the answer a while the answer-aware question is used as the question q corresponding to a.

– BART-A2QG + BERT-MRC: the BART-A2QG for question generation and the BERT model for answer extraction.

– BART-QG + BERT-MRC: BART model is used as the question generation model. The BERT model is used for answer extraction.

- **Our Methods**

  – OneStop: our proposed OneStop model involves only one model, in which the process of question generation and answer extraction is simultaneous and affects each other.

  – OneStop + BERT-MRC: the approach in which we used the question generated by OneStop and the answer extracted by the BERT model.

5.3 Evaluation Metrics

We evaluated the involved baselines from two aspects: (1) the similarity between generated questions and the ground-truth; (2) the quality of the generated QA pairs.

Similarity Between Generated Questions and Ground-Truth We chose BLEU-1, BLEU-2 [25], Rouge-1, Rouge-2, and Rouge-L [22] as the evaluation metrics to evaluate the similarity between the generated questions and the ground-truth questions.

Quality of Generated Question-Answer pairs   Since there is no widely-accepted automatic metrics on the quality of QA pairs, we used the score from two human annotators as the quality of generated QA pairs. The specific scoring criteria of the human raters are given as follows:

- Score 0: if any of the following cases are encountered, the QA pairs is given a score of 0. (1) there are serious grammatical errors in the question; (2) the question is an empty string; (3) the question is totally unrelated to the given document; (4) the question is unanswerable, i.e., the question cannot be answered by referring to the document; (5) The question and the answer are totally unrelated.
- Score 0.5: the question is partly related to the document.
- Score 1: the question is closely related to the document and is grammatically correct, but the answer is not associated with the question.
- Score 1.5: the question is closely related to the document and is grammatically correct, but the answer can only partially answer the question or contains redundant information.
- Score 2: the question is closely related to the document, and the answer can precisely and concisely reply to the question.

These involved baselines are compared based on an average over human raters' scores.

5.4 Model Settings

The encoder and the decoder in all the involved pre-trained language models contain 6 layers and a hidden size of 768. We utilized one well-trained English pre-trained BART model as the initial checkpoint of BART-A2QG, BART-QG and OneStop on SQuAD and NewsQA datasets. For models on the DuReader dataset, we pre-trained the BART language model on a very-large Chinese Baike dataset as the initial checkpoint of BART-A2QG, BART-QG and OneStop. In our experiments, we set $\gamma = 0.2$. The beam size is set to be 3. The batch size is 16 and epoch is set to 4. We chose Adam as our optimizer. The learning rate is set to 1e-4 with a warmup ratio of 0.05. The dropout rate $p = 0.1$. All the experiment is run with P100 GPUs.

6 EXPERIMENTAL RESULTS AND ANALYSIS

6.1 Results on Question Generation

We list the baselines' performance on question generation in Table 3. As can be observed, OneStop model achieves better or comparable performance than the baselines on question generation. The performance of DeepNQG is quite poor since the RNN-based model cannot handle the long document well. The comparison between BART-QG and OneStop proves that the answer extraction does not degenerate the quality of generated questions. It even improves the performance of generated question models. This phenomenon can be explained by the fact that the probability of answer extraction $p(a|d, \hat{q})$ can further enhance the question generation model $p(q|d; \theta_{d2q})$.

6.2 Results on QA Pair Generation

We list the result of QA evaluation in Table 4. From the results, we have the following observations:

(1) OneStop outperforms pipeline methods: CRF-DeepNQG + BiLSTM-CRF, DeepNQG + BERT-MRC, and BART-A2QG + BERT-MRC significantly and achieves a human rater's score 1.41, which prove the effectiveness of the OneStop model on question-answer pair generation.

(2) Compared with OneStop, OneStop + BERT-MRC sees an additional performance improvement. The difference between OneStop (1.41) and OneStop + BERT-MRC (1.67) proves that the answer extraction module in OneStop model is not as good as the answer extraction model implemented with BERT. This can be explained by the fact that the BERT model for answer extraction has 12 transformer layers, which has better representation

capacity on the document and question encoding. However, both the encoder and decoder in the OneStop model have only 6 transformer layers, which has a less satisfying representation ability.

(3) The comparison between BART-QG + BERT-MRC and OneStop + BERT-MRC is to verify the effectiveness of answer extraction module on OneStop's question generation module. The improvement (from 1.61 to 1.67) demonstrates that the answer extraction module in the OneStop model enhances the quality of QA pairs.

Table 4: Result of generated question-answer pairs.

Approach	Score	Approach	Score
CRF-DeepNQG + BiLSTM-CRF	0.22	BART-QG + BERT-MRC	1.61
DeepNQG + BERT-MRC	0.20	OneStop	1.41
BART-A2QG + BERT-MRC	0.24	OneStop + BERT-MRC	**1.67**

6.3 Model Efficiency

Another significant advantage of our OneStop model is efficiency:

(1) We list the number of parameters of each QA pair generation approach in Table 5. As we can see, OneStop model is one of the lightest model for QA pair generation.

(2) Different from the pipeline approaches that involve more than one model, which requires additional efforts and computational resource to train and deploy these models. OneStop, nevertheless, involves only one model, which is much more efficient for both training and deployment.

(3) The pipeline baselines require additional human efforts when deployed online. For instance, for the answer extraction (AE) model in D2A2Q, it may select none or more than one answer span, which requires well-designed rules to select from these answer span for question generation.

Table 5: The number of parameters (millions) of each QA pair generation approach.

Approach	SQuAD	NewsQA	DuReader
CRF-DeepNQG + BiLSTM-CRF	109	71	163
DeepNQG + BERT-MRC	151	146	323
BART-A2QG + BERT-MRC	248	248	423
BART-QG + BERT-MRC	248	248	423
OneStop	142	142	121
OneStop + BERT-MRC	253	253	427

6.4 Case Study

We present several QA pairs generated by our OneStop model in Table 6. We also include the answer span predicted by the BERT model. Based on our observation, in most cases, the answer predicted by the OneStop model is same as that of BERT. But for certain cases, the answer extraction in OneStop may tend to include the related information besides the precise answer span. Most of the extracted QA pairs from our OneStop model can be applied in downstream tasks like question answering.

Table 6: Question-answer pairs generated by OneStop.

	text
document	The French crown's refusal to allow non-Catholics to settle in New France may help to explain that colony's slow rate of population growth compared to that of the neighbouring British colonies, which opened settlement to religious dissenters.
OneStop question	What did the French government refusal to allow?
BERT-MRC answer	non-Catholics
OneStop answer	non-Catholics to settle in New France
document	The delta is delimited in the West by the Alter Rhein ("Old Rhine") and in the East by a modern canalized section.
OneStop question	What is the delta delimited by?
BERT-MRC answer	Alter Rhein
OneStop answer	Old Rhine

7 CONCLUSION

Existing pipeline QA pair generation approaches suffer problems like incompatible and sub-optimal solutions, inefficiency, and heavy human effort. This paper proposes a transformer-based sequence-to-sequence model to generate QA pairs in a one-stop fashion. Our model achieves state-of-the-art performance on question generation and QA pair generation on three large-scale machine reading comprehension datasets in a more efficient way. Our work sheds light on a novel One-Stop approach to QA pair extraction. We will explore more effective techniques of generating QA pairs such as the soft approach of answer extraction and copy mechanism.

REFERENCES

[1] 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*. 6168–6173.

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

[3] Ying-Hong Chan and Yao-Chung Fan. 2019. A Recurrent BERT-based Model for Question Generation. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*. 154–162.

[4] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 1724–1734.

[5] Shaobo Cui, Rongzhong Lian, Di Jiang, Yuanfeng Song, Siqi Bao, and Yong Jiang. 2019. DAL: Dual Adversarial Learning for Dialogue Generation. In *Proceedings of the Workshop on Methods for Optimizing and Evaluating Neural Language Generation*. 11–20.

[6] 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)*. 4171–4186.

[7] Xinya Du and Claire Cardie. 2018. Harvesting Paragraph-level Question-Answer Pairs from Wikipedia. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 1907–1917.

[8] Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to Ask: Neural Question Generation for Reading Comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 1342–1352.

[9] David Golub, Po-Sen Huang, Xiaodong He, and Li Deng. 2017. Two-Stage Synthesis Networks for Transfer Learning in Machine Comprehension. In *Proceedings*

of the 2017 Conference on Empirical Methods in Natural Language Processing. 835–844.

[10] Wei He, Kai Liu, Jing Liu, Yajuan Lyu, Shiqi Zhao, Xinyan Xiao, Yuan Liu, Yizhong Wang, Hua Wu, Qiaoqiao She, et al. 2018. DuReader: a Chinese Machine Reading Comprehension Dataset from Real-world Applications. In *Proceedings of the Workshop on Machine Reading for Question Answering.* 37–46.

[11] Michael Heilman and Noah A Smith. 2010. Good question! statistical ranking for question generation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics.* 609–617.

[12] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in neural information processing systems.* 1693–1701.

[13] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[14] Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *arXiv preprint arXiv:1508.01991* (2015).

[15] Sathish Reddy Indurthi, Dinesh Raghu, Mitesh M Khapra, and Sachindra Joshi. 2017. Generating natural language question-answer pairs from a knowledge graph using a RNN based question generation model. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers.* 376–385.

[16] Mandar Joshi, Eunsol Choi, Daniel S 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).* 1601–1611.

[17] Yanghoon Kim, Hwanhee Lee, Joongbo Shin, and Kyomin Jung. 2019. Improving neural question generation using answer separation. In *Proceedings of the AAAI Conference on Artificial Intelligence,* Vol. 33. 6602–6609.

[18] Kalpesh Krishna and Mohit Iyyer. 2019. Generating Question-Answer Hierarchies. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics.* 2321–2334.

[19] Igor Labutov, Sumit Basu, and Lucy Vanderwende. 2015. Deep questions without deep understanding. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers).* 889–898.

[20] 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* (2020).

[21] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics.* 7871–7880.

[22] Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out.* 74–81.

[23] Bang Liu, Haojie Wei, Di Niu, Haolan Chen, and Yancheng He. 2020. Asking Questions the Human Way: Scalable Question-Answer Generation from Text Corpus. In *Proceedings of The Web Conference 2020.* 2032–2043.

[24] Liangming Pan, Wenqiang Lei, Tat-Seng Chua, and Min-Yen Kan. 2019. Recent advances in neural question generation. *arXiv preprint arXiv:1905.08949* (2019).

[25] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics.* 311–318.

[26] Weizhen Qi, Yu Yan, Yeyun Gong, Dayiheng Liu, Nan Duan, Jiusheng Chen, Ruofei Zhang, and Ming Zhou. 2020. ProphetNet: Predicting Future N-gram for Sequence-to-Sequence Pre-training. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings.* 2401–2410.

[27] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research* 21 (2020), 1–67.

[28] 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.* 2383–2392.

[29] Kazutoshi Shinoda and Akiko Aizawa. 2020. Variational Question-Answer Pair Generation for Machine Reading Comprehension. *arXiv preprint arXiv:2004.03238* (2020).

[30] Linfeng Song, Zhiguo Wang, and Wael Hamza. 2017. A unified query-based generative model for question generation and question answering. *arXiv preprint arXiv:1709.01058* (2017).

[31] Sandeep Subramanian, Tong Wang, Xingdi Yuan, Saizheng Zhang, Adam Trischler, and Yoshua Bengio. 2018. Neural Models for Key Phrase Extraction and Question Generation. In *Proceedings of the Workshop on Machine Reading for Question Answering.* 78–88.

[32] Xingwu Sun, Jing Liu, Yajuan Lyu, Wei He, Yanjun Ma, and Shi Wang. 2018. Answer-focused and position-aware neural question generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing.* 3930–3939.

[33] Duyu Tang, Nan Duan, Tao Qin, Zhao Yan, and Ming Zhou. 2017. Question answering and question generation as dual tasks. *arXiv preprint arXiv:1706.02027* (2017).

[34] Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2016. Newsqa: A machine comprehension dataset. *arXiv preprint arXiv:1611.09830* (2016).

[35] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems.* 5998–6008.

[36] Siyuan Wang, Zhongyu Wei, Zhihao Fan, Yang Liu, and Xuanjing Huang. 2019. A multi-agent communication framework for question-worthy phrase extraction and question generation. In *Proceedings of the AAAI Conference on Artificial Intelligence,* Vol. 33. 7168–7175.

[37] Tong Wang, Xingdi Yuan, and Adam Trischler. 2017. A joint model for question answering and question generation. *arXiv preprint arXiv:1706.01450* (2017).

[38] 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).* 1040–1050.

[39] Xingdi Yuan, Tong Wang, Caglar Gulcehre, Alessandro Sordoni, Philip Bachman, Saizheng Zhang, Sandeep Subramanian, and Adam Trischler. 2017. Machine Comprehension by Text-to-Text Neural Question Generation. In *Proceedings of the 2nd Workshop on Representation Learning for NLP.* 15–25.

[40] Yao Zhao, Xiaochuan Ni, Yuanyuan Ding, and Qifa Ke. 2018. Paragraph-level neural question generation with maxout pointer and gated self-attention networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing.* 3901–3910.