Consecutive Question Generation with Multitask Joint Reranking and Dynamic Rationale Search

Anonymous ACL submission

Abstract

Automatic question generation (QG) aims to generate a set of questions for a given passage, and can be viewed as a dual task of question answering (QA). However, most current methods of QG tend to generate question by question independently, mainly based on specific extracted answer spans. In this paper, we propose to consecutively generate questions over a whole passage, with a comprehensive consideration of the aspects including accuracy, diversity, informativeness, and coverage. First we exam four key elements in QG, i.e., question, answer, rationale, and context history, and propose a novel multitask framework with one main task generating a question-answer pair, and four auxiliary tasks generating other elements alternately, improving model performance from all aspects through both joint training and reranking. Further, to learn the connection between questions and fully exploit the important information in every sentence, we propose a new consecutive generation strategy, which dynamically selects the rationales and searches for the best question series globally. Extensive experiments on different datasets show that our method can improve question generation significantly and benefit multiple related NLP tasks.

1 Introduction

Question Generation (QG) is an important and promising task in natural language generation (NLG). It has long served as an effective way to construct and enlarge the dataset of question answering (QA) (Duan et al., 2017; Dong et al., 2019). Besides, as more extensive research comes into this area, the applications of synthetic questions have expanded from mere data augmentation to building tutoring or dialogue systems (Lindberg et al., 2013; Bordes and Weston, 2017), self-assessing the ability of language models (Sun et al., 2019), and checking the faithfulness of an abstract summary (Durmus et al., 2020), etc.

Today is Jessica's 80th birthday. Her daughter Mela and Mela's husband Josh is coming over to the birthday party...
Q1: ***Who is coming over?*** –simple
A1: ***Mela and Mela's husband Josh.*** –extractive
Q2: ***Who is Josh?*** –lack of coverage
A2: ***Mela's husband.*** –lack diversity
Q3: ***Who has a birthday party?*** –not connected
A3: ***Mela.*** –QA inconsistent

Table 1: A problematic question generation on a passage using two-step inconsecutive method based on extractive answers.

Traditionally, syntax-based methods such as semantic parsing are commonly adopted to synthesize questions (Berant et al., 2013; Khullar et al., 2018). Recently, with the development of deep learning, transformer-based pre-trained language models (Vaswani et al., 2017; Devlin et al., 2019) have been widely used to generate questions. Most of these QG researches mainly focus on generating questions independently (Sun et al., 2018; Rennie et al., 2020), and rely on the ground-truth or extracted answers to generate the corresponding questions (Wang et al., 2019; Jia et al., 2020), as shown in Table 1. However, in real scenarios such as daily conversations or reading comprehension, we usually raise several questions consecutively to understand the whole story. In such cases, current methods are inadequate and cannot learn the connections between multiple questions, leading to shortage of coverage and diversity. In addition, current pre-extracted non-free-form answers may result in simple questions, and the two-step generation is inclined to QA-inconsistency.

To solve these problems, we propose to generate consecutive question-answer pairs over a whole passage. We first adopt a Seq2Seq model to generate a question-answer pair in one step. To improve the questions from the aspects of accuracy, diversity, and informativeness, we further comprehensively exam four key elements in QG, i.e., question, answer, rationale[1], and context history[2], and introduce four corresponding auxiliary tasks to assist the generation. These tasks also follow the Seq2Seq pattern and synthesize different elements respectively. In training, the five tasks are trained jointly in one model to help it from different views. During inference, the main task generates several question-answer candidates and the auxiliary tasks use their losses to rerank them. This is the multitask joint reranking framework, which helps enhance each question-answer pair all-roundly.

After this, to generate consecutively and cover most of the information in one passage, we let each question depend on previous ones, and employ each sentence of the passage to be a potential rationale. To exploit the information in each sentence as much as possible, we sample the rationales following a proposed probability formula to guide the question generation. We also develop the beam-search method to sentence-level, which always keeps several results dynamically and seeks for the final best output. The rationale sampling and new beam-search compose our dynamic rationale search strategy, which helps our model generate questions for a whole passage.

Our main contributions are three-fold:

- We propose to consecutively generate question-answer pairs for a whole passage, where questions are mutually connected and can fully cover the content of the passage.
- We introduce the multitask joint reranking framework to improve the quality of questions from four aspects, and the dynamic rationale search strategy to exploit the rationales and search for the best result globally.
- We conduct abundant experiments on various tasks and manually evaluate our strategy. Particularly, we promote the performance on multiple QA scenes and prove the expansibility of our model on different NLP tasks.

2 Related Work

Question generation is a promising task which is well-studied by a lot of researchers (Heilman and Smith, 2010; Du et al., 2017; Du and Cardie, 2018). Since the appearance of various pre-trained language models (Radford and Narasimhan, 2018; Devlin et al., 2019; Lewis et al., 2020), QG has been a crucial support for data augmentation of QA tasks (Liu et al., 2020; Kannan et al., 2021). Specifically, Zhou et al. (2019) and Ma et al. (2020) employ the multitask structure to generate coherent and fluent questions. Sachan and Xing (2018) and Rennie et al. (2020) adopt self-training strategy to jointly learn to ask and answer questions. Krishna and Iyyer (2019) propose a pipelined system to ask different level of questions from general to specific. Sultan et al. (2020) analyze the importance of diversity in QG and the effect of nucleus sampling (Holtzman et al., 2020). Alberti et al. (2019) use roundtrip consistency to filter out inconsistent results, like our reranking strategy. Durmus et al. (2020) and Wang et al. (2020) check the faithfulness of summaries through answering generated questions. Similar with us, Pan et al. (2021) generate question-answer pairs, but convert them for fact verification. Inspired by Yuan et al. (2021), we use the losses of auxiliary tasks to evaluate the candidates. However, most previous methods generate questions depending on extracted answers and cannot consecutively synthesize a series of high-quality question-answer pairs to cover an entire passage.

3 Method

Here we introduce the consecutive question generation, which synthesize a series of connected questions to totally cover the information of a passage. To achieve this, we propose our strategy with two main components, multitask joint reranking and dynamic rationale search. We jointly train a BART (Lewis et al., 2020) model on five different generation tasks. During inference, in each generating step, the main task produces several question-answer candidates and four auxiliary tasks use their losses to rerank them. From the view of the whole synthesizing procedure on a passage, every step we sample the next rationale following a dynamic probability formula, and keep a few question-answer flows like the beam-search.

3.1 Multitask Joint Reranking Framework

Multitask Joint Training

In our training data, there are several elements in one instance, which are the story, question, answer,

[1]The sentence based on which a question is generated.

[2]The coverage of all previous rationales, representing the information of current question series.

S: *Once upon a time in Greece, there lived a young man called Narcissus. He lived in a small village on the sea and was famous in the land because he was quite handsome. ...*	
Q_1: What was the name of the young man?	A_1: Narcissus.
R_1: Once upon a time in Greece, there lived a young man called Narcissus.	
Q_2: Where did he live?	A_2: Small village on sea.
R_2: He lived in a small village on the sea and was famous in the land because he was quite handsome.	
Q_3: Was he famous in the land?	A_3: Yes.
R_3: He lived in a small village on the sea and was famous in the land because he was quite handsome.	
Q_4: Why?	A_4: Because he was quite handsome.
R_4: He lived in a small village on the sea and was famous in the land because he was quite handsome.	

Task	Input	Output
a	$Q_1A_1 \cdots Q_{n-1}A_{n-1} < sep > \ answer\ this : Q_n < sep > \ S$	A_n
q	$Q_1A_1 \cdots Q_{n-1}A_{n-1} < sep > \ question\ it : A_n < sep > \ S$	Q_n
$main$	$Q_1A_1 \cdots Q_{n-1}A_{n-1} < sep > \ pose\ pair : R_n \ < sep > \ S$	Q_nA_n
r	$Q_1A_1 \cdots Q_{n-1}A_{n-1} < sep > \ find\ rationale : \ Q_nA_n < sep > \ S$	R_n
h	$Q_1A_1 \cdots \ Q_nA_n < sep > \ generate\ history < sep >$	$\bigcup_{i=1}^{n} R_i$

Table 2: An example of data composition of our multitask generation framework, as well as the input and output in the n^{th} generation step. We use "$\bigcup$" to represent coverage. In this example, the output of task h is R_1 when $n = 1$, and is $R_1 + R_2$ when $n \geq 2$.

and rationale. The rationales are the relevant parts in the stories where we can find the answers, and they are all whole sentences in our tasks. To raise connected questions consecutively, every question after the first one depends on the question-answer history, so we define the context of each question as the story plus previous question-answer pairs. In detail, we use the following symbols. S : $story$, Q : $question$, A : $answer$, R : $rationale$, C : $previous\ QA\ pairs + S$.

In traditional methods, to obtain a Q we must use an A as the preliminary, otherwise we have to first extract an answer from the passage. However, we think the extractive answer is too simple and it is indirect to get a question in two steps. This may also lead to inconsistency and ambiguity of the question-answer pairs. Thus, in our strategy, we input the context and rationale to a model and output the question and answer directly. This is the main task of our model. Using the n^{th} turn as an example: $task\ main : C_n + R_n \rightarrow Q_n + A_n$.

To guarantee that the generated question and answer are accurate, we make sure that given the question we can get the answer and given the answer we can get the question. This leads to two of our four auxiliary tasks: $task\ a : C_n + Q_n \rightarrow A_n$. $task\ q : C_n + A_n \rightarrow Q_n$.

Here task a follows traditional QA form. We don't input the rationale in task q because previous QA pairs are included in the context, so if A_n is an accurate candidate answer, the model should recognize the connection between the answer and the previous QA pairs, and restore the question easily.

Moreover, although we input the rationale in task $main$, it doesn't necessarily imply that the question-answer pair is derived from it. So we introduce task r to verify that the model indeed uses the information in input rationale to get the question and answer. $task\ r : C_n + Q_n + A_n \rightarrow R_n$.

Task r helps produce QA pairs asked from the corresponding rationale, and then increase the factual diversity of a QA series, which means more segments of the story will be precisely referred to.

Finally, to generate an informative and useful question, which means the knowledge it asks for doesn't overlap with previous questions, we consider that the more new information included in the question-answer pairs, the better. We introduce the history of the context as the coverage of all previous rationales, which represents the total information till current QA turn. Therewith, we present task h, which uses QA pairs to restore the history. $task\ h : \sum_{i=1}^{n}(Q_i + A_i) \rightarrow \bigcup_{i=1}^{n} R_i$.

We use "$\bigcup$" to represent coverage. Since we don't input the context, the model has to generate the history completely based on QA pairs. Therefore, if a QA pair carries more new information, it will be easier for it to restore the history compared

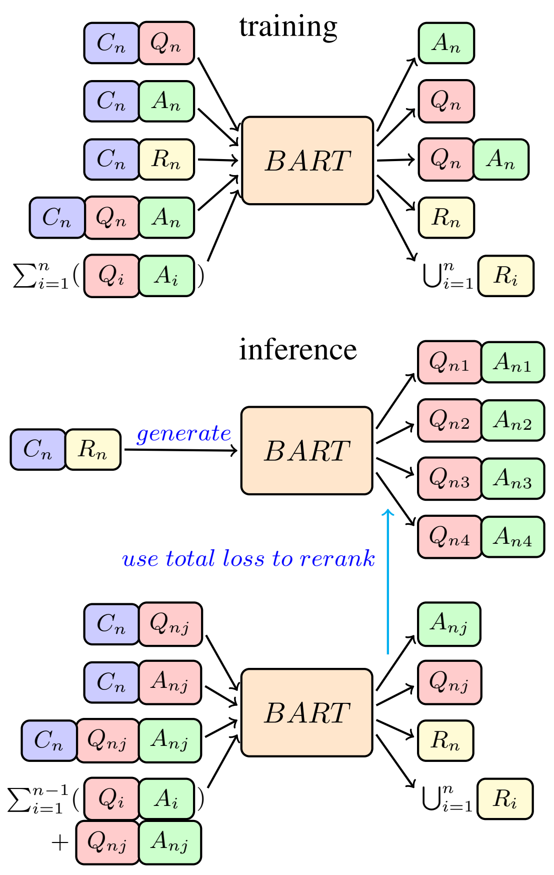

Figure 1: An overview of our multitask joint reranking framework, when generating the n^{th} turn of a series of questions. We generate 4 candidates. $j \in \{1, 2, 3, 4\}$.

with a QA pair with a lot of repetitive information.

Besides, we add some separate tokens $< sep >$ in the input sequences and adopt five hand-made prompts (Liu et al., 2021). Table 2 shows an example of our data structure. We randomly shuffle the five kinds of instances into one dataset and use a BART model to jointly train the five tasks together. Given the Seq2Seq model parameterized by θ, input sequence $\boldsymbol{x}$ with n tokens = $\{x_1, \cdots, x_n\}$ and label $\boldsymbol{y}$ with m tokens = $\{y_1, \cdots, y_m\}$, the generation probability and loss are as follows:

$$p(\boldsymbol{y}|\boldsymbol{x}, \theta) = \prod_{z=1}^{m} p(\boldsymbol{y}_z|\boldsymbol{y}_{<z}, \boldsymbol{x}, \theta) \tag{1}$$

$$loss(\boldsymbol{y}|\boldsymbol{x}, \theta) = -\frac{1}{m}\sum_{z=1}^{m} log\, p(\boldsymbol{y}_z|\boldsymbol{y}_{<z}, \boldsymbol{x}, \theta) \tag{2}$$

Reranking

During the inference stage, through the main task we can obtain many candidate question-answer pairs using a decoding strategy like nucleus sampling (Holtzman et al., 2020). To select the result with the best quality, we rank the candidates using the losses of task a,q,r, and h. The corresponding question and answer of the auxiliary tasks are those which are generated from task $main$. Specifically, we multiply the four losses together as the final criteria, or reranking loss, as Equation 3, where the subscripts i refer to different tasks.

$$loss_{rank}(\boldsymbol{y}|\boldsymbol{x}, \theta) = \prod_{i \in \{a,q,r,h\}} loss(\boldsymbol{y}_i|\boldsymbol{x}_i, \theta) \tag{3}$$

We consider the candidate with the lowest reranking loss as the one who excels in accuracy, diversity, and informativeness generally. This is inspired by the idea of evaluating generated text as text generation (Yuan et al., 2021). Through this strategy we also unify the form of our main task and four auxiliary tasks and manage to jointly train them in one model. Figure 1 shows the structure of our multitask joint reranking framework.

3.2 Dynamic Rationale Search

Rationale Sampling

The aforementioned framework is useful for generating one question-answer pair. Still, how to effectively generate consecutive questions on a passage remains unsettled. By default, we can set every rationale to the next sentence of previous one. However, one rationale does not necessarily correspond to only one question, because a long informative sentence may be suitable for more QA pairs.

Hence, we propose the rationale sampling strategy, which introduces a probability that the rationale remains in the same sentence as the previous one, as Figure 2 shows. We use the length of each rationale on behalf of its information. Also, we make the probability depend on the max turn number and give it a default value. Therefore, we propose the distribution as follows, where rp means the remaining probability; x means the ratio between the current rationale length and the story length; b is the selected upper bound of rp; d is the value of x when rp reaches the upper bound; t is the set maximum turns number; and μ is the default remaining probability when x equals to $1/t$.

$$rp = \begin{cases} b, \quad d \le x \le 1 \\ \mu + \dfrac{b-\mu}{d-1/t}(x - 1/t), \quad 1/t \le x < d \\ \mu + \dfrac{\mu}{1/t}(x - 1/t). \quad 0 \le x < 1/t \end{cases} \tag{4}$$

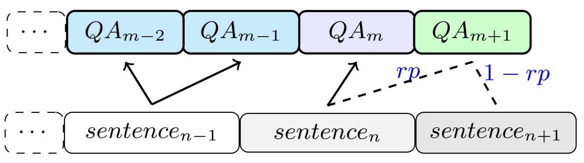

Figure 2: An example of rationale sampling strategy, in which there is a probability of rp that the rationale of QA_{m+1} is $sentence_n$, and $1 - rp$ it is $sentence_{n+1}$.

We can see that rp is piecewise-linear and passes $(0, 0)$, $(1/t, \mu)$ and (d, b), which ensures the longer a rationale is, the more likely it is asked multiple times. By adjusting the parameters we can control the average QA numbers in a passage. If we let $b = 0.75,\ d = 0.25,\ t = 20,\ \mu = 0.15$.

$$rp = \begin{cases} 3x, & 0 \le x \le 0.25 \\ 0.75. & 0.25 < x \le 1 \end{cases} \tag{5}$$

This is the setting we adopt after tuned, and we manage to generate about 15 turns per story, which is nearly 30% more than without the strategy.

Sentence-Level Beam-Search

Although rationale sampling can help catch more information and improves the flexibility, it brings about more uncertainty. When consecutively generating mutually dependent QA pairs, every turn from the first to the last is equally important, so it is also crucial to ensure the quality of whole series.

Naturally, inspired by traditional beam-search (token-level), we propose the sentence-level beam-search. As Figure 3 shows, besides reranking the candidates in each turn, we also keep several possible flows, and instead of generating a token in each search step, we generate a QA pair. We adopt the reranking loss of each QA pair to take the place of the generation probability of each token in traditional beam-search. Every timestep we maintain several candidates with the lowest product of all previous reranking losses, which means

$$L(Q_1A_1 \cdots Q_nA_n|\boldsymbol{x}, \theta) = \prod_{j=1}^{n} loss_{rank_j} \tag{6}$$

where L is the loss of the new beam-search, and multitask reranking plays an important role again.

4 Experiments

4.1 Experimental Setup

We employ CoQA (Reddy et al., 2019) training set as our training data. The questions are con-

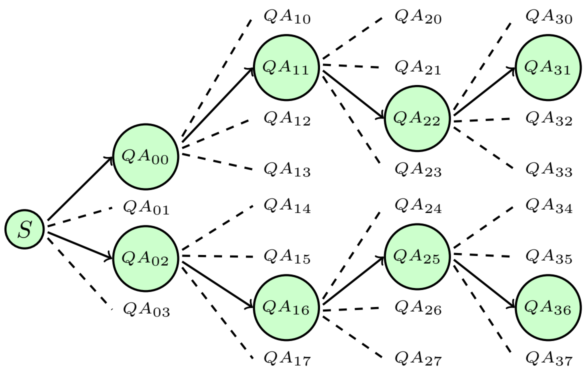

Figure 3: An overview of the dynamic sentence-level beam-search strategy. In this example each step the model generates 4 question-answer candidates and the sentence-level beam size is 2.

versational, and thus, every question after the first is dependent on the conversation history. The answers are free-form text with their corresponding rationales in the story. We expand the rationales to whole sentences and remove the questions with an unknown answer. Finally, we get 7199 stories and each story has 15 turns of question-answer pair on average. After jointly training a BART model θ on CoQA, we test its question generation ability using two main tasks, data augmentation for QA and document-level NLI.

Implementation Details

We use beam-search with beam size 4 to generate answers for QA. Following Sultan et al. (2020), we use nucleus sampling with top-k(k=50) and top-p(p=0.95) to generate question-answer pairs. We averagely return 4 candidates each step and set sentence-level beam size to 4 in sentence-level beam-search. The models we use are all *base* size.

We use PyTorch to implement our models. We acquire the BART model pre-trained by Facebook[3] on the Transformers library (Wolf et al., 2020). The hyperparameters after tuned and other training details are shown in Table 3. We use 1 NVIDIA GeForce RTX 3090 GPU for training and inference. It costs about 10 hours per epoch to jointly train five tasks together.

Metrics

Most natural language generation tasks adopt BLEU score (Papineni et al., 2002) or ROUGE score (Lin, 2004) as important metrics. However, both Sultan et al. (2020) and Schlichtkrull and

[3]`https://huggingface.co/facebook/bart-base`

Batch size	Learning rate	Epochs	Grad norm
64	1e-5	16	1.0
Dropout	Weight decay	Optimizer	Warm up
0.1	0.1	AdamW	0.1

Table 3: Implementation details.

Cheng (2020) suggest that standard QG evaluation metrics such as BLEU, are inversely correlated with diversity. Thus, inspired by (Yuan et al., 2021), we also use the losses to measure the performance, and adopt BLEU only to check the answers. In QA tasks, we additionally employ the F1$_{qa}$ score (lexical overlap) as the CoQA leaderboard does. In document-level NLI, we meanwhile use the F1$_{nli}$ score (harmonic mean of the precision and recall on the classification task).

4.2 Experiments of Training

Firstly, we evaluate the losses of five tasks on CoQA dev set after training. We also calculate the F1$_{qa}$ scores using task a. Table 4 shows the results of models with different training settings. We can see that joint training improves the performance on four out of five tasks, suggesting that different tasks benefit each other effectively. Prompts also improve the QA ability and lead to decreases of losses on three out of five tasks.

Metric	Ours	w/o Prompts	w/o Joint
$Loss$ a	**0.767**	0.771	0.777
$Loss$ q	**1.364**	1.370	1.377
$Loss$ m	**1.372**	1.378	1.388
$Loss$ r	0.062	**0.058**	0.068
$Loss$ h	2.554	2.543	**2.536**
$F1_{qa}$ a	**80.60**	80.07	78.54

Table 4: Losses and F1$_{qa}$ scores on CoQA dev set using different training method.

4.3 Experiments to Augment QA Data

Data augmentation is one common way to employ generated questions and verify QG models. To augment QA dataset D, we (1) use θ to synthesize QA pairs D' on the training set of D; (2) train another BART model θ' on D' or $D + D'$ to answer questions; (3) test θ' on the dev set of D.

Results on CoQA

First we test our strategy to augment CoQA dataset. The setting $Origin$ means the model θ' is trained on original CoQA training set, and $Synth$ means it is trained with synthetic QA pairs. There are also two synthesizing conditions, relay and auto. Relay means the previous QA pairs of every synthetic instance are from CoQA training set, and auto means they are the QA pairs generated in previous steps. In other words, in relay the model inherits the QA flow from authentic CoQA's context, and in auto the model automatically synthesizes the whole series. Additionally, we merge original training set with synthetic data to create the merging setting (D+D'). We don't implement dynamic rationale search in relay.

Model	Bleu	Loss	F1$_{qa}$
$Origin$			
Baseline[4]	-	-	79.80
Bart	38.52	0.777	78.54
$Synth$			
Bart relay	35.11/45.24	5.477/0.781	75.90/81.79
Bart auto	24.88/38.26	5.674/0.768	65.05/80.52
+RS	31.75/45.07	5.253/0.762	72.25/81.78
+SBS	31.75/**47.85**	5.461/**0.762**	72.21/**81.88**
$rerank\ by$			
a	26.11/42.01	5.652/0.810	68.84/81.37
q	35.70/42.71	5.341/0.769	73.53/81.31
r	33.02/41.85	5.437/0.776	71.73/81.08
h	27.97/41.22	5.509/0.794	70.47/81.28
w/o rerank	34.24/41.32	5.348/0.755	72.78/81.16

Table 5: Results on CoQA dev set of different models. Baseline is the result on CoQA in-domain test set of an extractive BERT-base model by Tsinghua University CoAI Lab. In $Synth$, results without and with merging are separated by "/". Below are five ablation experiments of auxiliary tasks with Bart auto+RS+SBS. RS: rationale sampling. SBS: sentence-level beam-search.

Table 5 shows the results. We can see from bart relay that the generated data indeed help answer questions, improving the F1$_{qa}$ scores by 3.25 points. In auto generation, both rational sampling and sentence-level beam-search improve the performance and result in higher F1$_{qa}$ scores, and finally it outperforms relay generation by 0.09 F1$_{qa}$ scores. We also conduct other five experiments where synthesized questions are reranked by different tasks. It is not hard to see that the multitask reranking strategy measure the results effectively, leading to 0.72 higher F1$_{qa}$ scores.

[4]https://stanfordnlp.github.io/coqa/

Results on SQuAD

To check the ability to generate questions on out-of-domain passages, we augment SQuAD (Rajpurkar et al., 2018) dataset using our model trained on CoQA. We select the instances without unknown answers and with a story longer than 128 words. Since the questions in SQuAD are mutually independent, there are no previous QA pairs. However, the questions in the dataset are well-organized, so we manually add previous QAs to check their effects and align with CoQA.

Model	**Bleu**	**Loss**	**F1$_{qa}$**
Origin			
Baseline[5]	-	-	83.06
Bart	65.52	0.675	84.26
+preQA	**68.67**	**0.625**	85.32
Synth			
Ours	36.92/64.37	4.579/0.636	62.57/84.35
+preQA	41.60/67.57	4.654/0.651	67.41/**85.61**

Table 6: Results on SQuAD dev set of different models. Baseline is the result on SQuAD test set of an extractive BERT-base model by Google AI. In $Synth$, results without and with merging are separated by "/".

Table 6 shows the results. We can see that the QA series indeed enhances question answering. It also indicates that even if our model is trained on another dataset, its synthesized questions still help a QA model gain 0.09 and 0.29 more F1$_{qa}$ points on SQuAD, with and without previous QA pairs. It shows that our consecutive questions performs well when transferring to a different dataset, too.

Results with more unlabeled Data

To truly reveal the ability of our model, we employ it to synthesize more questions on a large number of unlabeled passages. We randomly collect 10000 Wikipedia passages whose numbers of words are from 100 to 500. Then we use our model trained on CoQA to generate questions on them, with full strategies, resulting in about 0.15 million QA pairs. We use the Wikipedia questions as data augmentation, too, to evaluate their quality.

Tabel 7 shows the results. In both CoQA and SQuAD datasets, with more Wikipedia questions, we manage to further improve F1$_{qa}$ by 0.25 and 0.19 scores. Our model can augment the QA training sets with large-scale unlabeled data.

[5]https://rajpurkar.github.io/SQuAD-explorer/

Data	**Bleu**	**Loss**	**F1$_{qa}$**
CoQA			
Ours	31.75/47.85	5.461/0.762	72.21/81.88
+Wiki	33.06/47.34	5.433/0.763	72.73/**82.13**
SQuAD			
Ours	41.60/67.57	4.654/0.651	67.41/85.61
+Wiki	50.48/65.61	4.017/0.604	74.98/**85.80**

Table 7: Results with more Wikipedia Data. Results without and with merging are separated by "/".

4.4 Experiments to Cover a Passage

To prove that our generated questions can really cover most information in an entire passage, we adopt our model on document-level NLI (DocNLI) (Yin et al., 2021) task. Models are required to predict the relation (entailment or not) between a premise and a hypothesis. Traditionally, a model predicts the relation in a sequence classification way. However, given our ability to synthesize consecutive questions to explore a story, we propose a novel zero-shot method to predict the relation based on question generating and answering, using our model θ trained on CoQA. In detail, we (1) use θ to synthesize a series of Q and A on the hypothesis; (2) use θ to answer Q on the premise, obtaining A'; (3) check the overlap (F1$_{qa}$) between A and A'. The more same answers in A and A', the more chance the relation is entailment.

We select the instances whose premise and hypothesis are 200 to 1000 words from all train, dev, and test set of DocNLI, to be our evaluation set. It is a total of 1677 instances, and we averagely generate 15 turns of QA each instance with rationale sampling. We calculate F1$_{qa}$ per instance and use 60 points of F1$_{qa}$ as the boundary of entailment.

Model	**Loss**	**F1$_{qa}$**	**F1$_{nli}$**
Baseline [6]	-	-	46.18
Finetune	-	-	48.56
Ours	2.375/3.348	66.07/51.21	**49.88**
-SBS	2.631/3.612	65.93/51.01	49.85
-RS	3.217/4.149	63.04/49.68	47.91

Table 8: Results of DocNLI task. Baseline is the result on DocNLI dev set of an officially fine-tuned Longformer-base. Finetune is a BERT-base model fine-tuned on about 0.8 million other DocNLI instances different from our selected data. When using our zero-shot method, QA results of entailment and not entailment are separated by "/".

Tabel 8 shows the results. Impressively, without fine-tuning, our model using F1_{qa} as the criterion surpasses the fine-tuned BERT model by 1.32 points of F1_{nli} score. Also, we can see clearly that rational sampling and sentence-level beam-search improve the result significantly by 1.97 F1_{nli} scores and enlarge the discrimination between entailment and not entailment greatly. It suggests that our consecutive generation strategy really produces question-answer pairs with high coverage of a story, which can involve most of the information in one passage.

4.5 Analyses

Accuracy and Diversity

Here we conduct two human evaluations, to prove that our strategy improves question-answer accuracy and rationale dependency (factual diversity), which are the effects of tasks a, q and task r. We randomly collect 10% stories from CoQA dev set and use different methods to generate question-answer pairs. We then manually measure whether every question is correctly asked and answered and whether the question-answer pair is derived from their corresponding rationale.

Model	Acc of QA pair	Acc of rationale
Ours	**93.58**	**96.15**
-SBS	90.83	91.67
-Rerank	86.58	91.67

Table 9: Human evaluations of accuracy of QA and rationale.

Table 9 clearly shows that multitask reranking strategy and sentence-level beam-search increase the accuracy of QA by 7.00 % and rationale by 4.48 %. Thus, we can say that our strategy, especially tasks a, q and task r, help our model generate questions more correctly and locate the rationale more precisely, leading to higher QA accuracy and factual diversity in a series of questions.

Informativeness

To evaluate the ability to utilize information in a rationale, we present the repeat-pose experiment. It is adapted from relay setting, and requires the model to pose another question based on the same rationale and same context as the original question. In other words, the model has to "squeeze" more information from the same rationale, so the key is whether task h can rank the informativeness of each candidate precisely.

[6]Yin et al. (2021)

Model	Bleu	Loss	F1_{qa}
Bart relay w/o rerank	41.99	0.758	81.28
Bart repeat w/o rerank	43.67	0.757	81.37
Bart repeat w/ rerank	**43.00**	**0.748**	**81.69**

Table 10: Results of repeat-pose experiment. Synthetic data are merged with the original training set.

Table 10 shows the results, which indicates that repeat-pose indeed brings more information. Moreover, the reranking strategy further improves the F1_{qa} scores by 0.32 points, demonstrating that task h indeed helps select the more informative question-answer pairs.

One-Step generation or Two-Step

As argued in the introduction, two-step generation is inclined to inferior quality. To verify that, we trained two sets of models in augmenting CoQA with relay and auto+RS settings, without reranking, rationale sampling, or sentence-level beam-search. The two-step model first generates an answer given the rationale and context, and then generates a question based on the answer. Table 11 shows that directly one-step generation truly gains better results than separately two-step generation.

Model	Bleu	Loss	F1_{qa}
One-step	40.99/**43.53**	**0.788/0.806**	**81.40/81.09**
Two-step	**41.86**/42.66	0.821/0.808	80.93/80.88

Table 11: Results of augmenting CoQA dataset without reranking and sentence-level beam search strategy. Results of relay or auto+RS setting are separated by "/".

5 Conclusion

In this paper, we propose to generate consecutive question-answer pairs to improve the quality of questions and explore the information in a passage. By joint training and reranking with four auxiliary tasks, we help a model generate questions more accurately, diversely, and informatively. During consecutive generation, we sample the rationales dynamically and search for the best results globally, helping the questions cover a story more entirely. With extensive experiments, we prove that our model is able to generate high-quality questions for various NLP tasks and has the power to exploit large-scale unlabeled data.

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.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544, Seattle, Washington, USA. Association for Computational Linguistics.

Antoine Bordes and Jason Weston. 2017. Learning end-to-end goal-oriented dialog. *ArXiv*, abs/1605.07683.

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.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, M. Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. *ArXiv*, abs/1905.03197.

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)*, pages 1907–1917, Melbourne, Australia. Association for Computational Linguistics.

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)*, pages 1342–1352, Vancouver, Canada. Association for Computational Linguistics.

Nan Duan, Duyu Tang, Peng Chen, and Ming Zhou. 2017. Question generation for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 866–874, Copenhagen, Denmark. Association for Computational Linguistics.

Esin Durmus, He He, and Mona Diab. 2020. FEQA: A question answering evaluation framework for faithfulness assessment in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5055–5070, Online. Association for Computational Linguistics.

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*, pages 609–617, Los Angeles, California. Association for Computational Linguistics.

Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. *ArXiv*, abs/1904.09751.

Xin Jia, Wenjie Zhou, Xu Sun, and Yunfang Wu. 2020. How to ask good questions? try to leverage paraphrases. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6130–6140, Online. Association for Computational Linguistics.

Madeeswaran Kannan, Haemanth Santhi Ponnusamy, Kordula De Kuthy, Lukas Stein, and Detmar Meurers. 2021. Exploring input representation granularity for generating questions satisfying question-answer congruence. In *Proceedings of the 14th International Conference on Natural Language Generation*, pages 24–34, Aberdeen, Scotland, UK. Association for Computational Linguistics.

Payal Khullar, Konigari Rachna, Mukul Hase, and Manish Shrivastava. 2018. Automatic question generation using relative pronouns and adverbs. In *Proceedings of ACL 2018, Student Research Workshop*, pages 153–158, Melbourne, Australia. Association for Computational Linguistics.

Kalpesh Krishna and Mohit Iyyer. 2019. Generating question-answer hierarchies. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2321–2334, Florence, Italy. Association for Computational Linguistics.

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*, pages 7871–7880, Online. Association for Computational Linguistics.

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

David Lindberg, Fred Popowich, John Nesbit, and Phil Winne. 2013. Generating natural language questions to support learning on-line. In *Proceedings of the 14th European Workshop on Natural Language*

Generation, pages 105–114, Sofia, Bulgaria. Association for Computational Linguistics.

Dayiheng Liu, Yeyun Gong, Jie Fu, Yu Yan, Jiusheng Chen, Jiancheng Lv, Nan Duan, and Ming Zhou. 2020. Tell me how to ask again: Question data augmentation with controllable rewriting in continuous space. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5798–5810, Online. Association for Computational Linguistics.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ArXiv*, abs/2107.13586.

Xiyao Ma, Qile Zhu, Yanlin Zhou, Xiaolin Li, and Dapeng Oliver Wu. 2020. Improving question generation with sentence-level semantic matching and answer position inferring. *ArXiv*, abs/1912.00879.

Liangming Pan, Wenhu Chen, Wenhan Xiong, Min-Yen Kan, and William Yang Wang. 2021. Zero-shot fact verification by claim generation. In *ACL/IJCNLP*.

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*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Alec Radford and Karthik Narasimhan. 2018. Improving language understanding by generative pre-training.

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

Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Steven Rennie, Etienne Marcheret, Neil Mallinar, David Nahamoo, and Vaibhava Goel. 2020. Unsupervised adaptation of question answering systems via generative self-training. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1148–1157, Online. Association for Computational Linguistics.

Mrinmaya Sachan and Eric Xing. 2018. Self-training for jointly learning to ask and answer questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 629–640, New Orleans, Louisiana. Association for Computational Linguistics.

M. Schlichtkrull and Weiwei Cheng. 2020. Evaluating for diversity in question generation over text. *ArXiv*, abs/2008.07291.

Md Arafat Sultan, Shubham Chandel, Ramón Fernandez Astudillo, and Vittorio Castelli. 2020. On the importance of diversity in question generation for QA. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5651–5656, Online. Association for Computational Linguistics.

Kai Sun, Dian Yu, Dong Yu, and Claire Cardie. 2019. Improving machine reading comprehension with general reading strategies. 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 2633–2643, Minneapolis, Minnesota. Association for Computational Linguistics.

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*, pages 3930–3939, Brussels, Belgium. Association for Computational Linguistics.

Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *ArXiv*, abs/1706.03762.

Alex Wang, Kyunghyun Cho, and Mike Lewis. 2020. Asking and answering questions to evaluate the factual consistency of summaries. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5008–5020, Online. Association for Computational Linguistics.

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 *AAAI*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2020. Transformers: State-of-the-art natural language processing. In *EMNLP*.

Wenpeng Yin, Dragomir Radev, and Caiming Xiong. 2021. DocNLI: A large-scale dataset for document-level natural language inference. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4913–4922, Online. Association for Computational Linguistics.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. *ArXiv*, abs/2106.11520.

Wenjie Zhou, Minghua Zhang, and Yunfang Wu. 2019. Multi-task learning with language modeling for question generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3394–3399, Hong Kong, China. Association for Computational Linguistics.