A Question Type Driven and Copy Loss Enhanced Framework for Answer-Agnostic Neural Question Generation

Xiuyu Wu[1], Nan Jiang[2], Yunfang Wu[2]*

[1]School of Foreign Language, Peking University
[2]MOE Key Lab of Computational Linguistics, School of EECS, Peking University
textttxiuyu_wu@pku.edu.cn,jnhsyxxy@126.com,wuyf@pku.edu.cn

Abstract

The answer-agnostic question generation is a significant and challenging task, which aims to automatically generate questions for a given sentence but without an answer. In this paper, we propose two new strategies to deal with this task: question type prediction and copy loss mechanism. The question type module is to predict the types of questions that should be asked, which allows our model to generate multiple types of questions for the same source sentence. The new copy loss enhances the original copy mechanism to make sure that every important word in the source sentence has been copied when generating questions. Our integrated model outperforms the state-of-the-art approach in answer-agnostic question generation, achieving a BLEU-4 score of 13.9 on SQuAD. Human evaluation further validates the high quality of our generated questions. We will make our code public available for further research.

1 Introduction

Question Generation (QG) has been investigated for many years because of its huge potential benefits to various fields, especially for education (Mitkov et al., 2006; Rus and Arthur, 2009; Heilman and Smith, 2010). QG can also act as an essential component of other comprehensive task, such as dialogue systems (Piwek et al., 2007; Shum et al., 2018). Besides, it can supplement question answering task by automatically constructing a large question set (Duan et al., 2017; Tang et al., 2017).

Traditional methods for automatic question generation are mostly rule-based, which need a set of complex empirical rules (Beulen and Ney, 1998; Brown et al., 2005; Heilman and Smith, 2010; Mazidi and Nielsen, 2014). Recently, with the flourish of deep learning, especially the sequence-to-sequence (seq2seq) frame (Sutskever et al., 2014) with attention mechanism (Bahdanau et al., 2015), many neural models have been proposed to solve the QG task and achieve rapid progress (Du et al., 2017; Zhou et al., 2017; Tang et al., 2017; Tong et al., 2017; Zhao et al., 2018; Dong et al., 2019; Nema et al., 2019; Zhou et al., 2019).

Source sentence: the notion of style in the arts was not developed until the 16th century , with the writing of vasari : by the 18th century, his lives of the most excellent painters , sculptors, and architects had been translated into italian , french , spanish and english .
Groud-truth:
Q1: **when** were the *styles* of *arts* created ?
Q2: **who** wrote *lives* of the most *excellent painters* , *sculptors* , and *architects* ?
Q3: by the *18th century* **which** languages was *vasaris* book *translated* in ?
Q4: in **what** *century* did " *style* " as an artistic concept arise ?

Table 1: Different types of questions with respect to the same source sentence, where the italics words are keywords occurring both in the source sentence and reference questions.

However, most of the previous works are devoted to deal with answer-aware question generation. That is, given a text and also an answer span, the system is required to generate questions. But in a real application for educational purpose, people or machines are often required to generate questions for natural sentences without explicitly annotated answer. Comparing with the answer-aware QG, the answer-agnostic QG (AG-QG) task is more challenging and attractive. Unfortunately, AG-QG has been much less studied. Du's work (Du et al., 2017) is the first one to tackle this problem, and (Scialom et al., 2019) achieve the state-of-the-art by employing an extended transformer network.

For the AG-QG task, where the input is only a sentence but without any answer, multiple questions might be asked from various perspectives. According to our statistics on SQuAD (Rajpurkar

* Corresponding author.

Proceedings of the 4th Workshop on Neural Generation and Translation (WNGT 2020), pages 69–78

www.aclweb.org/anthology/D19-56%2d

et al., 2016), nearly 34% of the source sentences are offered multiple gold reference questions, and nearly 20% of the source sentences are offered different types of questions. Table 1 gives an example, where one source sentence corresponds with four different types of questions. However, most existing approaches can only generate one question for one input sentence.

To enable the model to ask different types of questions given the same input sentence, we propose a question type driven framework for AG-QG task. Specially, our model firstly predicts the probability of different question types distribution on the input sentence, which allows us to choose the best K question types with higher possibility. Then these different question types will be embedded into different vectors, which will guide the decoder to pay attention to informative parts with respect to different questions.

Meanwhile, according to the statistics on SQuAD, on average there are 3.09 non-stop words copying from the source sentence for each reference question. Those non-stop words appearing in both questions and sentences are regarded as keywords, since they act as the connection of these two parts. To increase the probability of copying keywords from source sentences, we design a new copy loss to enhance the traditional copy mechanism. In our model, by minimizing the new copy loss, the model will be forced to copy these keywords at least once during decoding.

We conduct experiments on SQuAD. Both the question type module and the new copy loss improve performance over the baseline model, and our full model combining two modules obtains a new state-of-the-art performance with a BLEU-4 of 13.9. Moreover, our model can ask different types of questions for a given sentence.

We conclude the contributions as follows:

- We propose a question type driven framework for AG-QG, which enables the model to generate diverse questions with high quality.
- We design a new copy loss function to enhance the standard copy mechanism, which increases the probability of copying keywords from source sentences.
- Our model achieves a new state-of-the-art performance for the challenging AG-QG. The human evaluation further validates a high quality of our generated questions in fluency, relevance and answer-ability.

2 Related Work

Answer-Aware Question Generation. Most previous works on question generation focus on answer-aware QG task. Yuan (Yuan et al., 2017) proposes three parts of loss to enhance the performance of sequence-to-sequence attention model. Zhou (Zhou et al., 2017) leverage lexical features (part-of-speech and named entity tags) to help the model get better encoder representation. Zhao (Zhao et al., 2018) uses paragraph information to do answer-aware QG task. And to use answer information more efficiently, Song (Song et al., 2018) uses multi-perspective matching and Sun (Sun et al., 2018) proposes position-aware model to pay more attention to the surrounding context of answer span. (Nema et al., 2019) uses a answer encoder to encode answer and fusion it with paragraph representation. (Chen et al., 2019) applies reinforcement learning to increase the performance.

Answer-Agnostic Question Generation. AG-QG is more challenging than answer-aware QG. Du's work (Du et al., 2017) is the first one to tackle this problem, and they achieve better performance than rule-based approaches by employing a sequence-to-sequence attention model. (Du and Cardie, 2017) aim to automatically find question-worthy sentences from a paragraph and then generate questions. (Subramanian et al., 2017) treat QG as a two-stage task: answer phrase extraction and answer-aware question generation. (Wang et al., 2019) propose a multi-agent communication framework, using a local extraction agent to extract question-worthy phrases, and then taking extracted phrases as assistance to generate questions. (Scialom et al., 2019) employ the transformer network (Vaswani et al., 2017) and extend it with the placeholder strategy, copy mechanism and contextualized embedding.

Question Word Prediction. Question word is one of the most important components of a question. (Fan et al., 2018) study multi-types visual question generation, by feeding the encoded representation to a multi-layer perception to calculate the question words distribution. (Sun et al., 2018) propose an answer-focused and position-aware model to generate the first question word. (Kim et al., 2019) propose an answer-separated sequence-to-sequence model to identify the proper question

word. They replace the answer span in the source sentence with a special token to make better use of the context information.

Multi-Types Question Generation. The multi-types QG has been much less researched. In Ma's work (Ma et al., 2018), in order to generate different types of questions, they use question type embedding at the first step of decoding. However, because of the difficulty in automatically predicting question types, their model fails to outperform the previous works. The question type driven framework has also been tried for visual question generation (Fan et al., 2018), where they concatenate the question type embedding with the encoded representation of input.

3 Proposed Framework

3.1 Framework Overview

For each sample in our dataset, we have a source sentence $S = (x_1, x_2, \ldots, x_l)$, which is a word sequence and *l* denotes the number of words. Let $Q = (y_1, y_2, \ldots, y_m)$ to represent the question, which is another word sequence and *m* is the length. The answer-agnostic QG task can be defined as finding the best $\overline{Q}$ that:

$$\begin{aligned}\overline{Q} &= \arg\max_{Q} \log P(Q \mid S) \\ &= \arg\max_{Q} \sum_{i=1}^{m} \log P(y_i \mid S, y_{<i})\end{aligned}$$

Figure 1 shows the framework of our full question type driven QG model. With the development of pointer network (Vinyals et al., 2015), the copy mechanism (Gu et al., 2016) has been increasingly more applied to natural language generation task. Thus, our model is based on the general sequence-to-sequence attention model with copy mechanism, which we regard as our baseline. In the next sections, we firstly describe the baseline model and then separately show our question type module and enhanced copy mechanism.

3.2 Baseline Model

Our baseline method is a sequence-to-sequence attention model with copy mechanism. Let x_t represent the $t-th$ word in the source sentence and $e(x_t)$ is its corresponding embedded vector. A bi-directional LSTM layer (Hochreiter et al., 1997) is used to encode the embedded vector sequence:

$$\begin{aligned}\overrightarrow{u_t} &= \overrightarrow{LSTM}(\overrightarrow{u_{t-1}}, e(x_t)) \\ \overleftarrow{u_t} &= \overleftarrow{LSTM}(\overleftarrow{u_{t+1}}, e(x_t)) \\ u_t &= [\overrightarrow{u_t}; \overleftarrow{u_t}]\end{aligned} \tag{1}$$

The hidden state at time step t is the concatenation of the forward $\overrightarrow{u_t}$ and backward $\overleftarrow{u_t}$, which can be represented as $\mathbf{U} = [\overrightarrow{u_t}; \overleftarrow{u_t}]_{t=1}^{l}$, where l is the number of words in the source sequence.

The decoder is another LSTM network, which generates a new hidden state h_t, conditioned on the previous state h_{t-1} and previously embedded generated word $e(y_{t-1})$. At the first decoding step, it takes the last encoding hidden state u_l and a special token $[SOS]$, which stands for the start of sequence, as input:

$$\begin{aligned}h_t &= LSTM(h_{t-1}, e(y_{t-1})) \\ h_0 &= LSTM(u_l, e([SOS]))\end{aligned} \tag{2}$$

Given the encoded state $\mathbf{U}$ and decoder state h_t, the baseline model calculates a generating distribution of words at each time step t, which is calculated as follow:

$$\begin{aligned}g_t &= softmax(\mathbf{W^o} tanh(\mathbf{W^t}[h_t; v_t])) \\ v_t &= \sum_{i=1}^{l} a_{it} u_i \\ a_{it} &= \frac{\exp(h_t^\top \mathbf{W^b} u_i)}{\sum_j \exp(h_t^\top \mathbf{W^b} u_j)}\end{aligned} \tag{3}$$

where v_t is the weighted sum of the encoded representation $\mathbf{U}$, and the attention weight a_t is calculated by a bi-linear scoring function and a softmax normalization. $\mathbf{W^t}$ and $\mathbf{W^o}$ are both trainable parameters, which can be regarded as applying a multi-layer perceptron to the concatenation of hidden state h_t and global attention representation v_t. $\mathbf{W^b}$ is also trainable parameter that is applied to calculate attention weights.

Our baseline model also utilizes the copy mechanism. We use the attention score a_t obtained from the decoder attention as the copy distribution. The probability of generating a word, p_gen, is calculated as $p\_gen = sigmoid(\mathbf{W^g}[h_t; v_t])$. The final distribution is the weighted sum of generating distribution and copy distribution:

$$\begin{aligned}\hat{g}_t &= p\_gen \cdot g_t \\ c_t &= (1 - p\_gen) \cdot a_t\end{aligned} \tag{4}$$

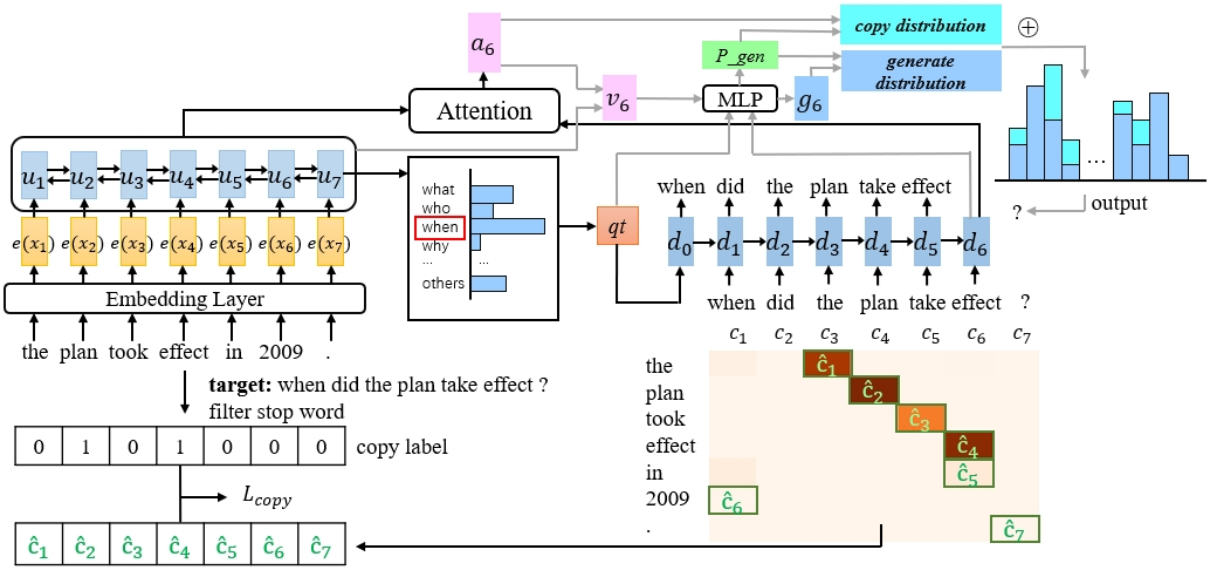

Figure 1: Our copy loss enhanced and question types driven framework

where $\hat{g}_t$, c_t are the weighted generate distribution and copy distribution, respectively. We use f_t to represent the final distribution and f_{ti} is the probability of decoding the $i-th$ word in the vocabulary. Let w_i represent the $i-th$ word in the vocabulary. The final distribution is calculated as follow:

$$f_{ti} = \begin{cases} \hat{g}_{ti} + \sum\limits_{k,where\ x_k=w_i} c_{tk} & w_i \in \mathbf{X} \\ \hat{g}_{ti} & otherwise \end{cases}$$

Given the training corpus D, in which each sample contains a source sentence S and a target question Q, the training objective is to minimize the negative log-likelihood of the target questions L:

$$L = -\sum_{<S,Q>\in D} \log P(Q|S;\theta) \tag{5}$$

where θ represents all the parameters of our model.

3.3 Question Type Prediction

We propose the question type module for two goals. One is to enable our model to generate multiple types of questions for one source sentence, and the other is to improve the generating performance. Our question type module firstly predicts the most proper type and then uses the embedding of it to help the decoding process.

As for question types, we count the distribution of question types in SQuAD and finally category all the questions into 7 types: what, who, how, where, when, yes/no and others.

The question type prediction is a multi-layer perceptron, which takes the last hidden state of encoder as input to predict the probability distribution of question types, denoted by $\mathbf{T}$:

$$\mathbf{T} = softmax(MLP(u_l)) \tag{6}$$

Please note that our model can generate multiple questions for one source sentence. When the number of questions need to be generated is set to K, our model will select K question types with the highest probability as output. Consequently, our decoder will decode K times.

$$ty_1, ty_2, \ldots, ty_K = TopK(\mathbf{T}) \tag{7}$$

At every decoding time, for one of the best K question types, ty, we embed it into a question type vector qt:

$$\begin{aligned} qt &= Embedding(ty) \\ ty &\in ty_1, ty_2, \ldots, ty_K \end{aligned} \tag{8}$$

The embedded question type vector will be used in decoding, and in this way, the question type vector would guide the model to generate questions that follow the pattern of a specific question type. Specially, we use the question type vector qt instead of the embedding of $[SOS]$ token as the input of the decoder at the first decoding step:

$$h_0 = LSTM(u_l, qt) \tag{9}$$

Besides, when calculating the generating distribution p_gen, we concatenate qt with the global

attention representation and the decoder hidden state:

$$g_t = softmax(\mathbf{W^o} tanh(\mathbf{W^t}[h_t; v_t; qt])) \\ p\_gen = sigmoid(\mathbf{W^g}[h_t; v_t; qt]) \quad (10)$$

Finally, we train the question type prediction and question generating simultaneously in multi-task learning framework. For each sample $< S, Q >$, we calculate the ground-truth question type distribution TY and we add an additional factor to the loss function, which is the negative log-likelihood of the target questions' types:

$$L = -\sum_{<S,Q,TY>\in D} [\log P(Q|S;\theta) + \lambda_1 \log P(TY|S;\theta)] \quad (11)$$

where λ_1 is a hyper-parameter to balance two parts.

3.4 Enhanced Copy Mechanism

According to our observation on SQuAD, when creating questions people often (may have to) copy some keywords from the source sentence. So we consider non stop words appearing in both questions and source sentences as keywords, and we assume that these keywords should also occur in the generated sequence. In order to push the model to copy keywords from source sentences, we propose a copy loss to enhance the traditional copy mechanism. For a word x_l in the source sentence, we first define a function $cl(copylabel)$ as follow:

$$cl(x_i) = \begin{cases} 1 & x_i \in Q \; and \; x_i \notin stopword \\ 0 & otherwise \end{cases} \quad (12)$$

We believe that at one decoding step, the copy probability of keywords ($cl(x_i) = 1$) should be close to 1. So we define the copy loss as:

$$\hat{c}_i = max(c_{1i}, c_{2i}, \ldots, c_{mi}) \\ L_{copy} = \sum_{i=1}^{l} cl(x_i)(\hat{c}_i - 1)^2 \quad (13)$$

where c_{ti} is the copy probability of the $i-th$ word in the source sentence at the $t-th$ decoding step, computed by Equation 4. Thus, $\hat{c}_i$ is the highest copy probability of the $i-th$ word in the source sentence among all the m decoding step. l denotes the number of words in the source sentence.

Finally, we add the copy loss into the total loss:

$$L = -\sum_{<S,Q,TY>\in D} [\log P(Q|S;\theta) + \lambda_1 \log P(TY|S;\theta) + \frac{\lambda_2}{l}\sum_{i=1}^{l} cl(x_i)(\hat{c}_i - 1)^2] \quad (14)$$

where λ_2 is another hyper-parameter to control the impact of penalty loss.

4 Experimental Settings

4.1 Dataset and Pre-processing

We conduct experiments on the SQuAD dataset (Rajpurkar et al., 2016), which contains more than 70k training samples, 10k development samples and 11k test samples. In either training, development or test dataset, multiple samples might share the same source sentence but with different target questions. But a same source sentence will not appear in different datasets, which ensures the confidentiality of test data.

We adopt subword representations (Sennrich et al., 2015) rather than raw words, which can not only reduce the size of vocabulary, increase the training speed, address the problem of out of vocabulary words, but also improve the model performance. By using byte-pair encoding, our vocabulary size is reduced to less than 6k. Due to the vanishing gradient problem in recurrent neural networks (Pascanu et al., 2013; **?**), we choose 256 for the maximum length of inputs and 50 for the maximum length of target questions.

4.2 Implementation Details

We adopt a 2-layers bi-directional LSTM for encoding and a 1-layer LSTM for decoding. The number of hidden units is 600, and the dimension of both word embedding and question type embedding is 300. We do not use pre-trained word embedding since we use subword representations rather than word-level representations. The drop rate (Krizhevsky et al., 2012) between each layer is 0.3. We firstly use Adam (Kingma and Ba, 2014) with learning rate of 0.001 for fast training, and after training 5 epochs, the stochastic gradient descent(SGD) with learning rate of 0.01 is used for fine-tuning. We train our model for 15 epochs with mini-batch size of 64. During training, hyper-parameter K is set to 1 and when decoding, we

do beam search with a beam size of 4. For hyper-parameters λ_1 and λ_2, we try different settings and choose the best one by observing the descending trend of total loss and ascending trend of BLEU-4 score on the valid set. Finally, both λ_1 and λ_2 are set to 0.1.

4.3 Evaluation Metrics

We adopt BLEU (Papineni et al., 2002), METEOR (Denkowski and Lavie, 2014) and ROUGE-L (Flick, 2004) for evaluation, and use the evaluation package released by Chen (Chen et al., 2015). BLEU measures the precision of n-grams on a set of references, with a penalty for over short generation. METEOR calculates the similarity between generations and references by considering synonyms, stemming and paraphrases. ROUGE measures the recall of n-grams on the set of references.

5 Results and Analysis

In this section, we report the automatic evaluation results of our proposed model and do ablation study to prove the effectiveness of different parts of the model. Then we conduct human evaluation and case study to test the quality of generated questions. Furthermore, we give a detailed analysis on multiple questions generation.

5.1 Main Results

We compare our model with the following previous works:

- **Seq2Seq Attention**: It is a traditional sequence-to-sequence attention model.
- **Seq2SeqAtt (Du)**: This is the first work in AG-QG task, which is a sequence-to-sequence attention model (Du et al., 2017).
- **Transformer (Scialom)**: This is the state-of-the-art result for the AG-QG task, which adopts a transformer network with some extension. (Scialom et al., 2019).

We do not take (Wang et al., 2019) into comparison because their evaluation is done on a different test set and is not accessible.

The experimental results are shown in Table 2. The full version of our model which uses both the question type module and copy loss mechanism obtains the best results on all of metrics, achieving a new state-of-the-art result of BLEU-4 13.90 for the challenging AG-QG task. It outperforms the baseline model with 0.73 points and beats the previous best result by 0.67 points.

5.2 Ablation Study

We conduct extensive experiments with different model modules, where k is set to 1 in decoding. The results are reported in Table 3.

- **Baseline**: Our baseline model is a general sequence-to-sequence attention model enhanced with copy mechanism.
- **Baseline+Type**: It adds the question type module to the baseline model.
- **Baseline+CopyLoss**: Based on the baseline model, it calculates and minimizes the additional copy loss.
- **Baseline+CopyLoss+Type**: This is the full version of our proposed model. That is, the question type module is applied to the baseline model and the extra copy loss is also calculated.
- **Upper Bound**: Since our full model incorporates the question type prediction part, the accuracy of question type prediction will undoubtedly affect the final quality of generation. If the right question type is given for every test sample, we get the upper bound of our model.

Effect of Question Type Module. Comparing with the baseline model, the question type module brings a slight performance gain. The upper bound shows if the right type is given for each test sample, the model will yield a much better performance with a BLEU-4 score of 15.27, which demonstrates the huge potential of our model. It proves that our model has successfully learned the patterns of different types of questions.

However, our question type predict module cannot achieve a 100% accuracy, and once a wrong question type is offered to the decoder, it will have negative influence on the generating quality. Actually, our question type predict part achieves an overall 69% accuracy, and the prediction results of different question types are shown in Table 4. It shows that without the answer as input, to predict the types of questions that should be asked for a given sentence is non-trivia.

model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	ROUGE-L
Seq2SeqAtt	42.90	25.97	17.68	12.49	16.85	39.59
Seq2SeqAtt (Du)	43.09	25.96	17.50	12.28	16.62	39.75
Transformer (Scialom)	43.33	26.27	18.32	13.23	-	40.22
Our model	**45.08**	**27.98**	**19.38**	**13.90**	**18.12**	**40.77**

Table 2: Experimental results comparing with previous works.

model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	ROUGE-L
Baseline	43.39	26.75	18.48	13.17	17.40	40.57
Baseline+Type	44.59	27.25	18.76	13.41	17.54	40.41
Baseline+CopyLoss	44.45	27.62	19.08	13.65	17.90	40.16
Baseline+CopyLoss+Type	**45.08**	**27.98**	**19.38**	**13.90**	**18.12**	**40.77**
Upper Bound	48.42	30.28	21.12	15.27	19.43	43.82

Table 3: Ablation study of our model.

Type	Total	Predict	Precision	Recall	Fscore
what	6707	8542	0.63	0.80	0.71
who	1421	1298	0.35	0.32	0.34
how	1476	1258	0.59	0.50	0.54
where	455	350	0.19	0.15	0.17
when	647	346	0.27	0.15	0.19
yes/no	868	65	0.09	0.01	0.01
others	303	18	< 0.01	< 0.01	< 0.01

Table 4: Prediction results of different question types.

	Key Words	Percentage
Baseline+Type	1.26	40.78%
Our model	**1.40**	**45.31%**
Golden	3.09	100%

Table 5: Performance of copying key words between two models with and without the enhanced copy mechanism.

Effect of Enhanced Copy Mechanism Our designed copy loss aims to enhance the copy mechanism. Since it tries to make the model ensure that every key word is copied, it directly leads to a higher BLEU-1. To our delight, the experiment shows the copy loss mechanism also contributes to a stable 0.48 increment of BLEU-4.

To make an in-depth analysis on the new copy mechanism, we also conduct experiments with and without the copy loss, counting the average number of keywords in questions generated by different models. The results are shown in Table 5. Our copy loss brings an absolute 4.53% increment on copying words from source sentences, which helps the model generate higher quality questions.

5.3 Human Evaluation

We also conduct human evaluation to judge the quality of questions generated by our model and the baseline model, respectively. We take three metrics into consideration: 1) Fluency: it measures whether the question is grammatical; 2) Relevance: it measures whether the question is asked for and highly related to the source sentence; and 3) Answer-ability: it measures whether the generated question can be answered by the information of the source sentence. We randomly selected 100 sentence-question pairs generated by different models, and asked three annotators to score the questions on a 1–5 scale (5 for the best). We also exploit the Spearman correlation coefficient to measure the inter-annotator agreement. The results are shown in Table 6. It shows that the consistency among three annotators is satisfying, and our generated questions are better from different perspectives.

	Fluency	Relevance	Answer-ability
Baseline	3.78	3.73	3.54
Our model	**4.23**	**4.23**	**4.20**
Golden	4.69	4.44	4.48
Spearman	0.59	0.67	0.65

Table 6: Human evaluation results of different models.

Besides, a further two-tailed t-test proves that our generated questions are better than that of the baseline model significantly, with $p < 0.001$ for every metric.

5.4 Case Study

In order to show the effectiveness of our model, we offer two real samples in the test set, as shown in Table 7. In both samples, the baseline model generates a wrong type of question while our model predicts the right type of question. At the same time, our model successfully copies more key words from the source sentence, which are shown in italics, while baseline model fails. In both samples,

Sample 1
Source sentence: the ex-president of def jam l.a. reid has described beyonc as the greatest entertainer alive .
Ground-truth: **who** has said that *beyonc* is the best entertainer alive ? **Baseline**: what is the greatest entertainer alive ? **Our model(K=1)**: **who** described *beyonc* as the greatest entertainer alive ?
Sample 2
Source sentence: tibetan sources say deshin shekpa also persuaded the yongle emperor not to impose his military might on tibet as the mongols had previously done .
Ground-truth: **who** convinced the *yongle emperor* not to send *military* forces into *tibet* ? **Baseline**: what did tibetan sources say deshin ? **Our model(K=1)**: **who** persuaded the *yongle emperor* not to impose his *military* might on *tibet* ?

Table 7: Examples of generated questions by different models.

Sample 1
Source sentence: by 1790 , new york had surpassed philadelphia as the largest city in the united states .
Groud-truth: by **which** year , did new york city become the largest city in the united states ? **what** was the second largest city in the united states in 1790 ?
Our Model (K=2): 1st: in **what** year did new york city surpassed philadelphia ? 2nd: **which** city had surpassed philadelphia in the us ?
Sample 2
Source sentence: buddhist architecture , in particular , showed great regional diversity .
Groud-truth: **which** cultures architecture showed a lot of diversity ? **what** type of architectural is especially known for its regional differences ?
Our Model (K=2): 1st: **which** buddhist architecture has showed great regional diversity? 2nd: **what** is buddhist architecture ?

Table 8: Different types of questions generated from the same input sentence.

our generated questions are more fluent and coherent.

5.5 Asking Different Types of Questions

For a given sentence, our question type driven framework offers the model the ability to generate different types of questions. In this case, the parameter K is set to more than 1, and the question type predictor will give K question types with the highest possibility. Then the model automatically decodes K times to generate the best K types of questions. We list a sample in Table 8 with $K = 2$ to show the generating diversity of our model, where two types of questions (what and which) are generated from the same input sentence.

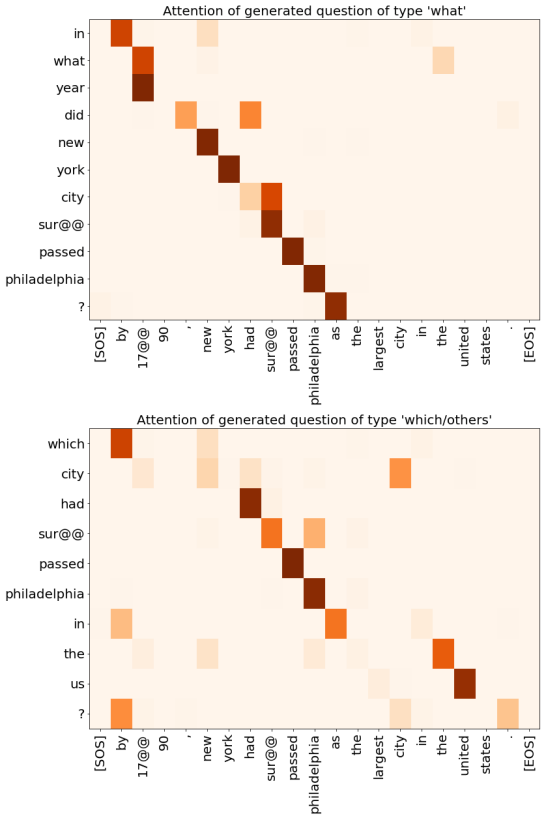

Figure 2: Heat map of attention for different types of generated questions.

Besides, to identify the effect of our model, we visualize the decoder attention, as shown in Figure 2. The two attention maps show the attention points when our model is generating different types (which and what) of questions with respect to the same input sentence, where x-axis is the source sentence and y-axis is the generated question. Differences between these attention maps prove that our model can attend on different information when generating different types of questions.

From the table, we prove that our model has the ability to generate multiple questions. However, the limitation is also obvious. First, if K is too large, the generated questions of some low probable types are of low quality. Second, since the probability distribution of question types are automatically calculated, the types of generated questions cannot be known beforehand.

6 Conclusion

In this paper, we propose two new strategies to deal with the answer-agnostic QG: question type module and copy loss mechanism. These proposed modules improve the performance over the base-

line model, achieving the state-of-the-art. Moreover, our model has the ability and flexibility to generate multiple questions for one source sentence. Hopefully, the idea of question type module and copy loss mechanism can also be used to do answer-aware QG task or other similar text generation tasks.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (61773026) and the Key Project of Natural Science Foundation of China (61936012).

References

Dzmitry Bahdanau, KyungHyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to alignand translate. In *3rd International Conference on Learning Representations, ICLR 2015*.

Klaus Beulen and Hermann Ney. 1998. Automatic question generation for decision tree based state tying. In *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'98 (Cat. No. 98CH36181)*, volume 2, pages 805–808. IEEE.

Jonathan Brown, Gwen A. Frishkoff, and Maxine Eskenazi. 2005. Automatic question generation for vocabulary assessment. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*.

Xinlei Chen, Hao Fang, Tsung Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollar, and C. Lawrence Zitnick. 2015. Microsoft coco captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*.

Yu Chen, Lingfei Wu, and Mohammed J. Zaki. 2019. Natural question generation with reinforcement learning based graph-to-sequence model. *arXiv preprint arXiv:1908.11813*.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the ninth workshop on statistical machine translation*, pages 376–380.

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 *Advances in Neural Information Processing Systems*, pages 13042–13054.

Xinya Du and Claire Cardie. 2017. Identifying where to focus in reading comprehension for neural question generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*.

Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. *arXiv preprint arXiv:1705.00106*.

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.

Zhihao Fan, Zhongyu Wei, Piji Li, Yanyan Lan, and Xuanjing Huang. 2018. A question type driven framework to diversify visual question generation. In *IJCAI*, pages 4048–4054.

Carlos Flick. 2004. Rouge: A package for automatic evaluation of summaries. In *Workshop on Text Summarization Branches Out*.

Jiatao Gu, Zhengdong Lu, Li Hang, and Victor O. K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. *arXiv preprint arXiv:1603.06393*.

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*.

S Hochreiter, Schmidhuber, and Jrgen. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

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*, volume 33, pages 6602–6609.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

Jinwen Ma, Wenpeng Hu, Bing Liu, Dongyan Zhao, and Rui Yan. 2018. Aspect-based question generation. *International Conference on Learning Representations 2018*.

Karen Mazidi and Rodney Nielsen. 2014. Linguistic considerations in automatic question generation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 321–326.

Ruslan Mitkov, Le An Ha, and Nikiforos Karamanis. 2006. A computer-aided environment for generating multiple-choice test items. *Natural Language Engineering*, 12(2):177–194.

Preksha Nema, Akash Kumar Mohankumar, Mitesh M. Khapra, Balaji Vasan Srinivasan, and Balaraman Ravindran. 2019. Lets ask again: Refine network for automatic question generation. *arXiv preprint arXiv:1909.05355*.

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 on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318.

Paul Piwek, Hugo Hernault, Helmut Prendinger, and Mitsuru Ishizuka. 2007. T2d: Generating dialogues between virtual agents automatically from text. In *International Conference on Intelligent Virtual Agents*.

Pranav Rajpurkar, Zhang Jian, 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*.

Vasile Rus and C Graesser Arthur. 2009. The question generation shared task and evaluation challenge. In *The University of Memphis. National Science Foundation*. Citeseer.

Thomas Scialom, Benjamin Piwowarski, and Jacopo Staiano. 2019. Self-attention architectures for answer-agnostic neural question generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Heung-Yeung Shum, Xiao-dong He, and Di Li. 2018. From eliza to xiaoice: challenges and opportunities with social chatbots. *Frontiers of Information Technology & Electronic Engineering*, 19(1):10–26.

Linfeng Song, Zhiguo Wang, Wael Hamza, Yue Zhang, and Daniel Gildea. 2018. Leveraging context information for natural question generation. 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 569–574.

Sandeep Subramanian, Wang Tong, Xingdi Yuan, and Adam Trischler. 2017. Neural models for key phrase detection and question generation. *arXiv preprint arXiv:1706.04560*.

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.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks.

Duyu Tang, Duan Nan, Qin Tao, and Zhou Ming. 2017. Question answering and question generation as dual tasks. *arXiv preprint arXiv:1706.02027*.

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

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

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in neural information processing systems*, pages 2692–2700.

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*, volume 33, pages 7168–7175.

Xingdi Yuan, Wang Tong, Caglar Gulcehre, Alessandro Sordoni, Philip Bachman, Sandeep Subramanian, Saizheng Zhang, Adam Trischler, Xingdi Yuan, and Wang Tong. 2017. Machine comprehension by text-to-text neural question generation. *arXiv preprint arXiv:1705.02012*.

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*, pages 3901–3910.

Qingyu Zhou, Nan Yang, Furu Wei, Chuanqi Tan, Hangbo Bao, and Ming Zhou. 2017. Neural question generation from text: A preliminary study. In *National CCF Conference on Natural Language Processing and Chinese Computing*, pages 662–671. Springer.

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 (EMNLP)*.