Enhancing Pre-trained Models with Text Structure Knowledge for Question Generation

Zichen Wu, Xin Jia, Fanyi Qu, Yunfang Wu[*]
Key Laboratory of Computational Linguistics, Ministry of Education, China
School of Computer Science, Peking University, China
{wuzichen,jemmryx,fanyiqu,wuyf}@pku.edu.cn

Abstract

Today the pre-trained language models achieve great success for question generation (QG) task and significantly outperform traditional sequence-to-sequence approaches. However, the pre-trained models treat the input passage as a flat sequence and are thus not aware of the text structure of input passage. For QG task, we model text structure as answer position and syntactic dependency, and propose answer localness modeling and syntactic mask attention to address these limitations. Specially, we present localness modeling with a Gaussian bias to enable the model to focus on answer-surrounded context, and propose a mask attention mechanism to make the syntactic structure of input passage accessible in question generation process. Experiments on SQuAD dataset show that our proposed two modules improve performance over the strong pre-trained model ProphetNet, and combing them together achieves very competitive results with the state-of-the-art pre-trained model.

1 Introduction

Question generation (QG) aims to generate questions for a given passage, which is a challenging task and shows great value in many practical applications. For instance, QG helps in building reading comprehension tests for course assessments (Kurdi et al., 2020); QG can be an important skill for chatbots to start a conversation with human users (Mostafazadeh et al., 2016); QG can help reduce the human labor for collecting large-scale question-answer datasets to improve question answering systems (Duan et al., 2017).

Existing QG methods mostly employ neural network models by treating it as a sequence-to-sequence generative task, where researchers have tried to incorporate text structure to improve the performance. For answer-aware QG, the answer position is widely explored. Song et al. (2018) propose three strategies to match the answer with passage; Sun et al. (2018) model the relative distance between context words and answer to help generate answer-related questions; Ma et al. (2020) enhance model's ability in capturing semantic and positional information via multi-task learning. Some other neural network models dig out deep text structure for generating better questions. For instance, Chen et al. (2020) construct a passage graph to utilize text structure; Pan et al. (2020) construct a semantic-level graph to learn the global structure of a document for deep QG. However, most of these methods adopt standard neural network, and few of them are based on the strong pre-trained models.

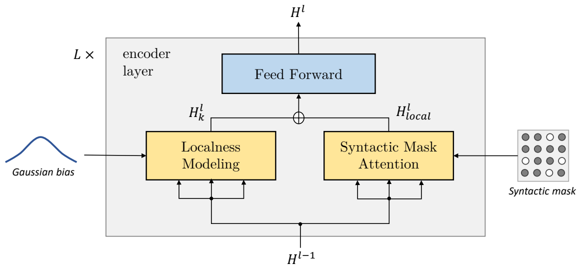

Figure 1: An overall structure of our proposed methods

Nowadays, pre-trained models largely boost the performance of QG systems (Dong et al., 2019; Bao et al., 2020; Qi et al., 2020; Xiao et al., 2020), which utilize huge amounts of unlabeled text corpora in pre-training stage to learn general language representations, and then be fine-tuned in supervised QG tasks. However, in most cases, the input passage and answer are formatted as "answer [sep] passage" to feed into the pre-trained model during the fine-tuning stage, as a result that the neural network cannot explicitly capture the syntactic structure of sentence and is not aware of the answer position (Zhou et al., 2017), which is vital to generate high-quality answer-aware questions. In this end, how to incorporate text structure into

[*]Corresponding author.

pre-trained models for QG remains an open issue.

Fortunately, several previous works have tried to utilize external knowledge and position information for NLP tasks. For instance, Yang et al. (2018) introduce token localness in attention mechanism for machine translation tasks, and Liu et al. (2020) propose a knowledge-enabled language model (K-BERT) to incorporate domain knowledge for question answering systems.

Motivated by these works, we propose a novel strategy to explicitly incorporate syntactic knowledge and answer information to pre-trained models for QG task. The overall architecture is illustrated in Figure 1. We present a localness modeling by designing a Gaussian bias to regulate the attention calculation, and propose a new attention mechanism to incorporate syntactic structure. Specially, in localness modeling, we adopt answer position as the center of Gaussian distribution and predict window size automatically, in order that the content around answer will be more focused and the range can be adjusted dynamically according to different token length. For syntactic knowledge, we first compute a syntactic vector through syntactic mask attention, which we design to blind the tokens outside of dependency triples, and then constitute a syntactic-aware representation by adding syntactic vector to the original context vector.

We base our method on the strong pre-trained model ProphetNet (Qi et al., 2020), which achieves state-of-the-art results on many generation tasks including QG. We conduct experiments on SQuAD dataset (Du et al., 2017). The automatic evaluation shows that our model boosts the performance of pre-trained models and achieves comparable state-of-the-art performance. Further human judgement demonstrates that our model produces high-quality questions by capturing syntactic knowledge and answer-surrounded context.

Since our method is only a modification to the self-attention module, and there is little increase in parameter number, one can easily apply our method to other Transformer-based pre-trained models while almost keeping the original speed without any much computation resources in the training stage. Our codes and data will be publicly available at the Github repository.

Contributions: To summarize, we (1) propose a modified attention mechanism to incorporate syntactic structure and position information, which can be easily applied to any Transformer-based pre-trained language models for QG task; and (2) with selection of syntactic knowledge, our model is able to reach comparable state-of-the-art performance for question generation.

2 Related Work

2.1 Question Generation

Recently, neural question generation approaches have become popular and achieved great success. Serban et al. (2016) first introduce an encoder-decoder framework with attention mechanism to generate questions for facts from FreeBase. Du et al. (2017) employ a seq2seq architecture on QG. Zhao et al. (2018) tackle the challenge of processing long text inputs with a gated self-attention encoder. Song et al. (2018) and Kim et al. (2019) strength model's ability by leveraging answer information. Meng et al. (2020) propose a two-stage QG model by creating a draft first and then doing refinement. These works adopt sequence-to-sequence generative approaches and yield many good results on QG. Benefiting from the development of pre-trained model, the performance of QG has been significantly improved. Varanasi et al. (2020) utilize the information from a self-attention module in BERT (Devlin et al., 2019) to generate questions. Qi et al. (2020) propose a pre-training model with n-stream self-attention mechanism and achieve notable results in QG. Xiao et al. (2020) design a span-by-span generation flow to predict semantically-complete spans consecutively. So far, Bao et al. (2020) achieve the best result on SQuAD dataset, which employ autoencoding and partially autoregressive modeling as the pre-training task.

2.2 Knowledge-Enhanced Text Generation

Many knowledge-enhanced text generation systems have achieved great performance on generating informative texts (Yu et al., 2020). In dialogue systems, topic-aware seq2seq model helps to generate more informative responses (Mou et al., 2016; Xing et al., 2017; Zhou et al., 2018). In story generation, commonsense knowledge helps to capture the story-line and narrates the plots (Guan et al., 2019). Liu et al. (2020) and Huang et al. (2021) incorporate external knowledge base into the Transformer-like architecture.

For question generation, Zhou et al. (2017) and Sun et al. (2018) leverage answer embedding and position information to help generate more answer-related questions; Du and Cardie (2018) incorpo-

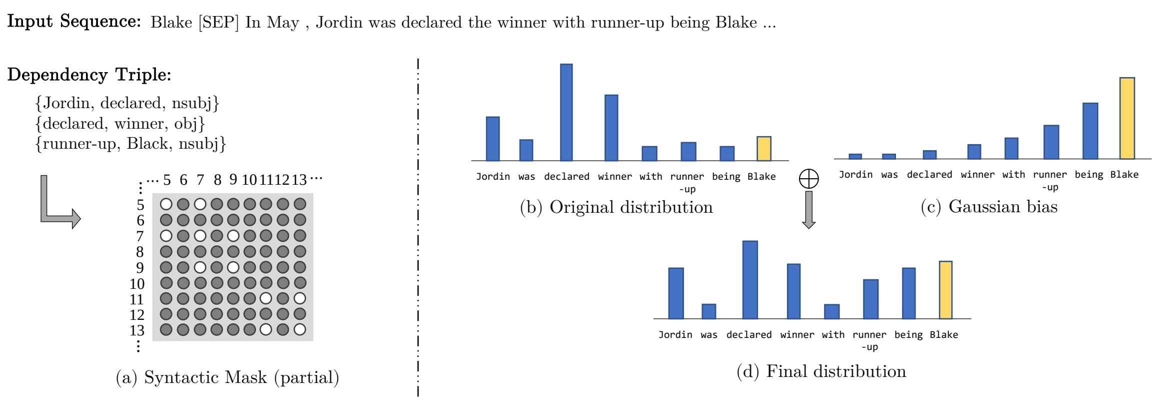
Input Sequence: Blake [SEP] In May , Jordin was declared the winner with runner-up being Blake ...
Dependency Triple:
{Jordin, declared, nsubj}
{declared, winner, obj}
{runner-up, Black, nsubj}
5 6 7 8 9 10 11 12 13
(a) Syntactic Mask (partial)
Jordin was declared winner with runner -up being Blake
(b) Original distribution
(c) Gaussian bias
(d) Final distribution

Figure 2: A diagram of Syntactic Mask (left) and Localness Modeling (right). (1) In subfigure (a), we show a partial mask attention with index ranging from 5 to 13 (representing tokens from 'Jordin' to 'Black' in the input sequence), where the black dot denotes value $-\infty$ and the white denotes 0. (2) Subfigures (b)-(d) show an example of distribution changes in calculating attentions weights of token 'declared', where we convert G to the form of distribution for better illustration.

rate coreference into QG model; Kim et al. (2019) propose a keyword-net to help the model better capture important information; Jia et al. (2020) integrate paraphrase knowledge into QG systems to generate more human-like questions. Chen et al. (2020) and Pan et al. (2020) try to capture text structure via graph neural network for QG.

The attention mechanism has been widely utilized to incorporate knowledge representation, which is obtained by calculating attention over knowledge representations. Multitask training is also an alternative way to incorporate knowledge (Kim et al., 2020; Jia et al., 2020).

In this paper, based on the pre-trained model for QG, we further incorporate dependency relations and answer position, which can be regarded as structure knowledge. Our task is more challenging since most of the previous works are based on the Transformer only. In order not to increase the computation consumption, we don't use any other sophisticated neural networks beyond the pre-trained model itself.

3 Preliminaries

3.1 Self-Attention

Here we briefly introduce the encoder of Transformer architecture, on which our methods will base. Given an input sequence $x = \{x_1, x_2, \cdots, x_t\}$, the transformer encoder maps these tokens into hidden representations. More concretely, the $l-1$-th layer hidden states are first linearly transformed into query vectors q, key vectors k and value vectors v by multiplying three separate weights, and then obtain the l-th layer hidden states via attention mechanism:

$$H^l = \text{Attn}(q, k, v) \tag{1}$$

$$\text{Attn}(q, k, v) = \text{softmax}\left(\frac{scores}{\sqrt{d}}\right)v \tag{2}$$

$$scores = qk^T \tag{3}$$

where $\sqrt{d}$ is the scaling factor.

3.2 Pre-trained Language Model: ProphetNet

We build our method based on the strong pre-trained model ProphetNet (Qi et al., 2020), which achieves outstanding results on QG tasks. Compared to conventional pre-training models which predict the next one token at each step, ProphetNet tends to predict the next n tokens simultaneously to prevent overfitting on local correlations:

$$p(y_t|y_{<t}, x), \cdots, p(y_{t+n-1}|y_{<t}, x) = \text{Decoder}(y_{<t}, H_{enc}) \tag{4}$$

where H_{enc} is the hidden state obtained by the encoder. To achieve this, ProphetNet adopts a n-stream self-attention mechanism to predict next n continuous future tokens, with each stream representing a next i-th predicting.

4 Integrating Text Structure

Our work focuses on answer-aware QG task. Formally, denote the passage, question and answer as P, Q and A respectively, the task is defined as:

$$Q^* = \underset{Q}{\text{argmax}}\, P_\theta(Q|P, A) \tag{5}$$

Practically, we concatenate passage and answer by format "A [SEP] P" as the input of ProphetNet, to generate Q^* as output question. A diagram of our method is illustrated in Figure 2.

4.1 Answer Localness Modeling

In order to enhance model's capability in capturing more local information around the answer and enable the model to generate more answer-related questions, we model localness as a preference bias to regulate the original attention weight in encoder end, which can be defined as:

$$\mathrm{Attn}(q,k,v) = \mathrm{softmax}(\frac{scores + G}{\sqrt{d}})v \quad (6)$$

where $G \in \mathbb{R}^{I \times I}$ is an alignment matrix, and I refers to the length of input sequence. Each element $G_{i,j}$ represents an alignment preference of token w_i to token w_j, which is calculated as:

$$G_{i,j} = -\frac{(j - P_c)^2}{2\sigma_i^2} \quad (7)$$

where P_c denotes the central position of answer appearing in the input passage and σ_i denotes the standard deviation, which is half of the window size D_i. Denote the start and end index of the answer span as s and e, then P_c is calculated as:

$$P_c = \frac{s+e}{2} \quad (8)$$

Consequently, after softmax operation, G will become a Gaussian distribution, indicating that the tokens closer to the central position will get higher attention weights.

The window size D_i measures the major area that token x_i needs to align with. We set D_i to be a variable based on the index i rather than a constant value, which we think will bring our model more flexibility to adjust the concentration area according to different token information. Following Yang et al. (2018), we map query vectors into D_i through a feed-forward neural network:

$$z_i = U_d^T \tanh(W_p q_i) \quad (9)$$
$$D_i = I \cdot \mathrm{sigmoid}(z_i) \quad (10)$$

where $W_p \in \mathbb{R}^{d \times d}, U_d \in \mathbb{R}^d$ are learnable parameters, and I is a scalar factor regulating D_i to a real value between 0 and the length of input sequence.

Thus, we can obtain hidden states of the input sequence with strengthened localness information:

$$H_{local} = \mathrm{MultiHead}(q,k,v,G) \quad (11)$$

Figure 2(b)-(d) shows an example of answer localness modeling. When calculating the attention weights of token 'declared', we first learn a Gaussian distribution centered on the answer 'Blake'. The original attention distribution is then regularized to form the final distribution, which not only pays attention to token 'winner' but also to token 'runner-up' and 'Blake'. As a result, the model is guided to attend to the phrase 'winner with runner-up being Blake'.

4.2 Syntactic Mask Attention

In this module, we strengthen the syntactic structure of input sentence accessible in question generation process. The whole procedure for incorporating syntactic knowledge can be divided into three steps: 1) extract appropriate syntactic relations; 2) build syntactic mask and 3) apply syntactic mask to guide the attention calculation. We will introduce them respectively.

4.2.1 Extract Syntactic Relations

We extract dependency relations and explicitly introduce them into the generation process. Due to the huge time needed for parsing, we only select some key sentences to do syntactic parsing rather than the whole passage. For every answer-passage pairs, we select the sentence from the passage where answer spans are located as our key sentence. If the answer spans do not appear completely in any sentences (this is possible because the sentence splitting toolkit is not perfect and sometimes it will mistake punctuation for a terminator), we will select the most similar sentence by computing ROUGE score as an alternative.

Then we apply dependency parser to these selected key sentences, and choose some major relations as the final dependency knowledge, including pred, subj, nsubjpass, csubj, csubjpass, obj, iobj and xcomp. The selection of dependency relations is based on de Marneffe et al. (2014). We have also tried some other relations, the analysis of which is given in Section 6.2. The dependency relation is presented in the form of triples, which can be denoted as $\varepsilon = \{x_i, x_j, r_k\}$, where x_i and x_j are tokens from the input sequence, and r_k denotes the relation between them.

In our experiment, we utilize the Stanford NLP toolkit [1] to do tokenization, sentence splitting and dependency parsing. A more powerful and sophisticated procedure to extract dependency triples might

[1]https://nlp.stanford.edu/software/

capture the syntactic structure more accurately, and we leave this to future work.

4.2.2 Syntactic Mask

Building syntactic mask is the key process in calculating knowledge context vector. Based on the extracted syntactic relation triples, we design a visible mask that prevents some tokens from seeing other ones, which can be defined as below:

$$M_{ij} = \begin{cases} 0, & x_i \ominus x_j \\ -\infty, & x_i \oslash x_j \end{cases} \tag{12}$$

where $x_i \ominus x_j$ means x_i and x_j are in a relation triple, while $x_i \oslash x_j$ means they are not.

We display an example in Figure 2(a) to show how we construct the syntactic mask from a sentence. We construct the relation triples like {'Jordin', 'declared', 'nsubj'} through extracting dependency relations and build syntactic mask to ensure 'Jordin' and 'declared' could see each other while blind to tokens outside the triples. And tokens not appearing in any triple, like 'was', 'the' and 'being', are not visible to each other.

4.2.3 Mask Attention

The process of mask attention is actually an extension to the original self-attention module in encoder end, whose attention score is calculated as:

$$scores = qk^T + M \tag{13}$$

where M is the syntactic mask constructed above.

After the softmax function, the attention score where $M_{ij} = -\infty$ will be 0, indicating that the hidden state of x_i will make no contribution to x_j. In this way, we build a syntactic context vector that gathers information of tokens appearing in the extracted syntactic relations while discards unrelated information, which can be a supplementary to the original context vector. Consequently, we obtain the knowledge context vector by:

$$H_k = \mathrm{MultiHead}(q, k, v, M) \tag{14}$$

Then the syntactic context vector will be added to the original vector to form the final syntactic-aware representation:

$$\tilde{H} = H_{local} \oplus H_k \tag{15}$$

4.3 Question Generation

After obtaining the encoder hidden states $\tilde{H}$, we pass them to the decoder and get the distribution on vocabulary at each step (defined in Equation 4). During the training stage, the loss is calculated as:

$$\mathcal{L} = -\sum_{j=0}^{n-1} \alpha_j \left(\sum_{t=1}^{T-j} \log p_\theta(y_{t+j} | y_{<t}, x) \right) \tag{16}$$

which can be viewed as the weighted sum of n future token predicting loss. And during the inference stage, we disable n-gram predicting and generate a sequence with the highest likelihood:

$$Q = \underset{y}{\mathrm{argmax}}\, p_\theta(y_{1:t} | x) \tag{17}$$

5 Experimental Setup

5.1 Dataset

We conduct experiments on the widely-used reading comprehension dataset SQuAD (Rajpurkar et al., 2016). It consists of questions posed by crowd workers on Wikipedia articles, and the answer to every question is a span extracted from the given passage. There exist different split sets on SQuAD. Following the work of ProphetNet (Qi et al., 2020), we do experiments on two splits: (1) Du split (Du et al., 2017), which splits SQuAD 1.1 dataset into training, development and test sets, consisting of 75,722, 10,570, 11,877 instances respectively; (2) Zhao split (Zhao et al., 2018), which uses the reversed dev-test setup as opposed to the original setup. Following previous work for QG, we adopt BLEU-4 (Papineni et al., 2002), Meteor (Denkowski and Lavie, 2014) and Rouge-L (Lin, 2004) as evaluation metrics.

5.2 Training Details

We adopt most of the configurations in ProphetNet released by Microsoft group [2]. However, due to computation resource limitation, we reimplemented their codes with some different configurations. We use one NVIDIA GeForce RTX 2080 Ti to support fine-tune, and during the training stage we truncate overlong input sentences to 512 tokens [3] and the batch size is set to 12. These might bring some decrease in performance, and we take the new results as our baseline for fair comparison. Inspired

[2]https://github.com/microsoft/ProphetNet

[3]Since ProphetNet only adopts input strings less than 512 tokens.

Methods (conference-year)		Du split			Zhao split		
		BLEU-4	Meteor	Rouge-L	BLEU-4	Meteor	Rouge-L
Unpre-trained	s2s (ACL-17)	12.28	16.62	39.75	-	-	-
	CorefNQG (ACL-18)	15.16	19.12	-	-	-	-
	MP-GSN (EMNLP-18)	-	-	-	16.38	20.25	44.48
	SemQG (EMNLP-19)	18.37	22.65	46.68	20.76	24.20	48.91
	RefNet (EMNLP-19)	-	-	-	18.16	23.40	47.14
	ParaphraseQG (ACL-20)	17.21	20.96	-	-	-	-
	EGSS (AAAI-21)	18.93	22.04	47.73	-	-	-
Pre-trained	ERINE-GEN$_{BASE}$ (IJCAI-20)	22.28	25.13	50.58	23.52	25.61	51.45
	CopyBert (ACL-20)	22.71	24.48	51.60	-	-	-
	ProphetNet (EMNLP-20)	23.91	**26.60**	52.26	25.80	**27.54**	53.65
	UniLM-v2 (ICML-20)	**24.43**	26.34	51.97	26.29	27.16	53.22
	Our model	24.37	26.26	**52.77**	**26.30**	27.25	**53.87**

Table 1: Experimental results on SQuAD dataset comparing with previous work

by Yang et al. (2018), who show pre-trained models tend to capture localness information in shallow layers while global in higher layers, we apply localness modeling in the first 4 encoder layers and syntactic attention mask in all 12 encoder layers[4].

5.3 Comparing Methods

There is a large number of work for QG on SQuAD dataset. Our focus in this paper is to enhance pre-trained models with text structure, so we mainly compare our method with other pre-trained language models. Also, we list some notable sequence-to-sequence works for reference, which did experiments on Du split or Zhao split.

The unpre-trained models for QG. s2s (Du et al., 2017): an attention-based seq2seq framework for QG. CorefNQG (Du and Cardie, 2018): incorporating coreference knowledge into seq2seq model. MP-GSN (Zhao et al., 2018): addressing the challenge of processing long text inputs. SemQG (Zhang and Bansal, 2019): introducing two semantic-enhanced rewards to regularize generation. RefNet (Nema et al., 2019): a two stage model which creates an initial draft first and then refine it. ParaphraseQG (Jia et al., 2020): incorporating paraphrase knowledge into question generation by back-translation. EGSS (Huang et al., 2021): an entity guided question generation model.

The pre-trained models applied to QG. ERNIE-GEN (Xiao et al., 2020): using multi-granularity target sampling in pre-training and span-by-span generation flow in predicting stage. CopyBert (Varanasi et al., 2020): utilizing information from self-attention modules of BERT in generation. ProphetNet (Qi et al., 2020): our baseline model, as described before. UniLM-v2 (Bao et al., 2020): pre-trained of bi-directional language modeling via auto-encoding and seq2seq generation via partially autoregressive modeling.

[4]In fact, we also conducted experiments on different number of layers, and the results will be reported in our Github repository.

6 Results and Analysis

6.1 Main Results

The main experimental results are shown in Table 1. For QG on SQuAD dataset, there exists a significant performance gap between the unpre-trained and pre-trained models. The best pre-trained model, UniLM-v2, achieves a 24.43 BLEU-4 on Du split, which receives 5.5 point improvement compared with the best seq2seq model EGSS. The ProphetNet model achieves a competitive BLEU-4 with UniLM-v2, and yields the best result on Meteor and Rouge-L among existing pre-trained models.

Considering the excellent performance of these pre-trained models, it's exciting to see that our proposed method with syntactic mask and localness modeling brings further improvement over the basic ProphetNet model, and obtains state-of-the-art results on Zhao split and a close performance with UniLM-v2 on Du split. It's worth noting that our model yields best results in terms of Rouge-L on both datasets among all existing works.

6.2 Ablation Study & Analysis

Ablation Study We conduct experiments by applying syntactic mask and localness modeling separately to study their effects on the baseline model. There is a tiny decrease between the result reported by Qi et al. (2020) and our reproduced result on some metrics, which we think is reasonable since we adopt some different configurations foremen-

Method	Du split			Zhao split		
	BLEU-4	Meteor	Rouge-L	BLEU-4	Meteor	Rouge-L
ProphetNet*	23.91	25.95	52.28	25.71	26.99	53.63
ProphetNet + Syntactic Mask	24.22	26.24	52.60	26.20	27.01	53.74
ProphetNet + Localness	24.11	26.01	52.52	25.88	**27.28**	53.62
ProphetNet + Syn. Mask + Localness	**24.37**	**26.26**	**52.77**	**26.30**	27.25	**53.87**

Table 2: Ablation study results by applying different modules on top of the pre-trained ProphetNet model. We report the mean over 3 random seeds. *We re-implemented the ProphetNet released code[2] and the results are a little lower than the original paper. Underline represents the value is better than baseline with significance ($p < 0.05$).

tioned in Section 5.2. As illustrated in Table 2, both components separately enhance the performance of basic ProphetNet and obtain better scores on almost all metrics on two data splits. The performance is further improved when combing two modules together, with a 0.56 BLEU-4 increase on Du split and 0.59 on Zhao split. It indicates that syntactic relations and position information are from two different feature subspace and are able to complement each other in enhancing model's awareness of text structure. And this combining improvement is guaranteed by conducting significance test.

Analysis on Dependency Relations We conduct experiments to investigate the effect of different dependency relations. According to de Marneffe et al. (2014), the dependency relations can be divided into two categories: core arguments and non-core arguments, and core arguments can be further divided into nominal relations and clause relations. Inspired by this, we adopt three strategies to select dependency relations: 1) using all dependency relations, which is equal to mask any other tokens except for those in the key sentence; 2) using core arguments, as in Section 4.2.1, which keeps the stem of sentence and gets rid of attributive words; 3) only using nominal relations in core arguments, which removes the clause relation of the stem and thus is the totally plain trunk of the sentence. The choices of core arguments are based on de Marneffe et al. (2014).

The experimental results are shown in Table 3. All three strategies bring performance gain over the basic ProphetNet, which validates the effectiveness of our syntactic mask attention. Especially, strategy 2) achieves the best results on both data splits, which provides appropriate dependency knowledge. In contrast, too much dependency relation incorporation may divert the sentence from its correct meaning, which can be found in Strategy 1), and the lack of dependency structure cannot produce a marked effect, which Strategy 3) exemplifies.

Dependency Relations	Du split	Zhao split
ProphetNet	23.81	25.71
+ all relations	23.93	26.17
+ core arguments	24.22	26.20
+ core nominal relations	24.01	26.13

Table 3: Experimental results of different dependency relations in the syntactic mask attention module.

Methods	Du split	Zhao split
ProphetNet	23.81	25.71
+ predicting center	23.77	25.72
+ answer center	24.11	25.88

Table 4: Experimental results of different center strategy in the localness modeling module.

Analysis on Localness Modeling Further, we conduct experiments on different methods to set center position of Gaussian distribution in localness modeling: (1) following Yang et al. (2018), who predict the central position by applying a feed-forward transformation to query vector

$$p_i = U_p^T \tanh(W_p q_i) \tag{18}$$

where W_p is a set of parameter to learn; (2) adopt the position of answer as center, as Equation (8).

The experimental results are shown in Table 4. Automatically predicting center position doesn't bring performance gain, while using answer center gets better results on both data splits. We argue that this is because mapping query vector to predict center will lead one token to align with other tokens that are similar with it, which is not applicable in our task. For answer-aware QG, we need each token to pay enough attention to the context around answer, so that the model will obtain better representations with answer as guidance.

Analysis on ProphetNet One may notice the contradiction between the goal of ProphetNet and our methods: ProphetNet predicts the next several

		Base	Ours	ρ
Fluency	No	4.33%	4.67%	0.428
	Med.	10.33%	9.33%	(3.8e-06)
	Yes	85.33%	86%	
	Avg.	2.81	2.81	
Relevancy	No	1.67%	1%	0.507
	Med.	14.33%	13.67%	(2.4e-07)
	Yes	84%	85.33%	
	Avg.	2.82	2.84	
Answerability	No	4%	3%	0.456
	Med.	12%	10.67%	(1.8e-06)
	Yes	84%	86.33%	
	Avg.	2.80	2.83	

Table 5: Human evaluation results on the generated questions by ProphetNet ("Base") and our full model ("Ours"). Avg. represents the average score of 100 samples. The last column is the Spearman coefficients with p-values in the parentheses.

Passage: ...Meanwhile, a cargo train carrying 13 petrol tanks derailed in Hui County, Gansu, and caught on fire after the *rail was distorted*. **Answer:** rail was distorted **Generated Questions:** **Golden:** why did the train catch fire? **Base:** why did the cargo train derail in hui county? **Ours:** what caused the cargo train to catch on fire?
Passage: ...A Japanese family with Malaysian citizenship and their 5-year-old child who unfurled a Tibetan flag were hit by a group of Chinese nationals with plastic air-filled batons and heckled by a crowd of Chinese citizens during the confrontation at Independence Square where the relay began, and the Chinese group shouted: "*Taiwan and Tibet belong to China.*" Later during the day... **Answer:** Taiwan and Tibet belong to China. **Generated Questions:** **Golden:** what did the chinese group yell? **Base:** what did the chinese say to the japanese family? **Ours:** what did the chinese yell at the japanese family?

Table 6: Two examples of generated questions, where answers are highlighted in *Italic font.*

tokens simutaneously to prevent overfitting on local correlations, however, our methods enhance the local syntactic structure in representing stage. Actually, they work on different parts of the pretrained model. ProphetNet uses n-gram prediction strategy in decoder end, while we enhance locality information in encoder end. The enhanced representations of input text, which guides the decoder to generate tokens with fully considering local structure information of the answer, can still benefit from n-gram prediction strategy, since the model would dynamically focus more on neighbour information when previous tokens appearing in the selected structure triples, while keep relatively low information when not. Therefore, more than ProphetNet, our methods should also contributes to other pretrained models.

6.3 Human Evaluation

In addition to automatic evaluation, we also evaluate the quality of generated questions by eliciting human judgements. We randomly select 100 {passage, question, answer} samples generated by ProphetNet baseline model and our full model, and asked 3 college students to evaluate them. They are required to annotate $yes(3), no(1)$ or $medium(2)$ for each sample from the following aspects: (1) fluency, whether the generated question is grammatical and fluent; (2) relevancy, whether the question is semantically relevant to the input context and (3) answerability, whether the answer is valid to the generated question based on the context.

The evaluation results are listed in Table 5. The pre-trained model ProphetNet provides us a strong baseline, where most of the generated questions are satisfying. Our model performs better in relevancy and answerability. We speculate this for our proposed method help model capture the information around answer and makes generated question more concentrated to answer. We also reports the average Spearman's coefficients between the annotations, which could guarantee the credibility of our human evaluation results.

6.4 Case Study

To clearly show the output questions generated by the basic ProphetNet model and our full model, we list two examples in Table 6. In both examples, the questions generated by our model are closer to golden questions. Specially, in the first example, both the baseline model and our model capture the information that "rail was distorted" is the cause to some event. But the event is "derail" in base while "catch on fire" in ours. The latter is more accurate because "rail was distorted" is the direct cause of "catch on fire", but the indirect cause of "derail". We think the enhanced ability of syntactic information helps capture the sentence structure more accurately and thus find the direct relation. In the second example, both models capture the information that "Taiwan and Tibet belong to China" is an utterance said by a Chinese group, but our model focuses more on the answer-surrounded word "shouted", leading to generate a more accurate word "yell".

7 Conclusion

In this work, we enhance the ability of QG systems by incorporating text structure, including syntactic dependency relations and answer position. We strengthen the localness modeling via a learnable Gaussian bias with answer span as center, and present a syntactic mask attention mechanism to fuse structure information. Specifically, we obtain the knowledge-aware context vector by adapting a visible matrix, where each token is only visible to its related token in the knowledge triple. Experimental results on the widely-explored SQuAD dateset demonstrate the effectiveness of our method. This work is based on the pre-trained ProphetNet, but our methods can be easily applied to other Transformer-based pre-trained models. In future work, we will validate and expand our method to other NLP tasks, such as summarization, dialog generation, machine reading comprehension, etc.

Acknowledgement

This work is supported by the National Natural Science Foundation of China (62076008) and the Key Project of Natural Science Foundation of China (61936012).

References

Hangbo Bao, Li Dong, Furu Wei, Wenhui Wang, Nan Yang, Xiaodong Liu, Yu Wang, Jianfeng Gao, Songhao Piao, Ming Zhou, et al. 2020. Unilmv2: Pseudo-masked language models for unified language model pre-training. In *International Conference on Machine Learning*, pages 642–652. PMLR.

Yu Chen, Lingfei Wu, and Mohammed J. Zaki. 2020. Reinforcement learning based graph-to-sequence model for natural question generation. In *International Conference on Learning Representations*.

Marie-Catherine de Marneffe, Timothy Dozat, Natalia Silveira, Katri Haverinen, Filip Ginter, Joakim Nivre, and Christopher D. Manning. 2014. Universal Stanford dependencies: A cross-linguistic typology. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4585–4592, Reykjavik, Iceland. European Language Resources Association (ELRA).

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, Baltimore, Maryland, USA. Association for Computational Linguistics.

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

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*, volume 32. Curran Associates, Inc.

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.

Jian Guan, Yansen Wang, and Minlie Huang. 2019. Story ending generation with incremental encoding and commonsense knowledge. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):6473–6480.

Qingbao Huang, Mingyi Fu, Linzhang Mo, Yi Cai, Jingyun Xu, Pijian Li, Qing Li, and Ho-fung Leung. 2021. Entity guided question generation with contextual structure and sequence information capturing. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(14):13064–13072.

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.

Byeongchang Kim, Jaewoo Ahn, and Gunhee Kim. 2020. Sequential latent knowledge selection for knowledge-grounded dialogue. In *International Conference on Learning Representations*.

Yanghoon Kim, Hwanhee Lee, Joongbo Shin, and Kyomin Jung. 2019. Improving neural question generation using answer separation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):6602–6609.

Ghader Kurdi, Jared Leo, Bijan Parsia, Uli Sattler, and Salam Al-Emari. 2020. A systematic review of automatic question generation for educational purposes. *International Journal of Artificial Intelligence in Education*, 30(1):121–204.

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.

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. 2020. K-bert: Enabling language representation with knowledge graph. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(03):2901–2908.

Xiyao Ma, Qile Zhu, Yanlin Zhou, and Xiaolin Li. 2020. Improving question generation with sentence-level semantic matching and answer position inferring. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8464–8471.

Chuan Meng, Pengjie Ren, Zhumin Chen, Christof Monz, Jun Ma, and Maarten de Rijke. 2020. Refnet: A reference-aware network for background based conversation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8496–8503.

Nasrin Mostafazadeh, Ishan Misra, Jacob Devlin, Margaret Mitchell, Xiaodong He, and Lucy Vanderwende. 2016. Generating natural questions about an image. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1802–1813, Berlin, Germany. Association for Computational Linguistics.

Lili Mou, Yiping Song, Rui Yan, Ge Li, Lu Zhang, and Zhi Jin. 2016. Sequence to backward and forward sequences: A content-introducing approach to generative short-text conversation. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3349–3358, Osaka, Japan. The COLING 2016 Organizing Committee.

Preksha Nema, Akash Kumar Mohankumar, Mitesh M. Khapra, Balaji Vasan Srinivasan, and Balaraman Ravindran. 2019. Let's ask again: Refine network for automatic 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 3314–3323, Hong Kong, China. Association for Computational Linguistics.

Liangming Pan, Yuxi Xie, Yansong Feng, Tat-Seng Chua, and Min-Yen Kan. 2020. Semantic graphs for generating deep questions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1463–1475, Online. Association for Computational Linguistics.

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.

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-SequencePre-training. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2401–2410, Online. Association for Computational Linguistics.

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

Iulian Vlad Serban, Alberto García-Durán, Sungjin Ahn, Sarath Chandar, Aaron Courville, and Yoshua Bengio. 2016. Generating factoid questions with recurrent neural networks: The 30M factoid question-answer corpus. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 588–598, Berlin, Germany. Association for Computational Linguistics.

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, New Orleans, Louisiana. 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.

Stalin Varanasi, Saadullah Amin, and Guenter Neumann. 2020. CopyBERT: A unified approach to question generation with self-attention. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 25–31, Online. Association for Computational Linguistics.

Dongling Xiao, Han Zhang, Yukun Li, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2020. Ernie-gen: An enhanced multi-flow pre-training and fine-tuning framework for natural language generation. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3997–4003. International Joint Conferences on Artificial Intelligence Organization. Main track.

Chen Xing, Wei Wu, Yu Wu, Jie Liu, Yalou Huang, Ming Zhou, and Wei-Ying Ma. 2017. Topic aware neural response generation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1).

Baosong Yang, Zhaopeng Tu, Derek F. Wong, Fandong Meng, Lidia S. Chao, and Tong Zhang. 2018. Modeling localness for self-attention networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4449–4458, Brussels, Belgium. Association for Computational Linguistics.

Wenhao Yu, Chenguang Zhu, Zaitang Li, Zhiting Hu, Qingyun Wang, Heng Ji, and Meng Jiang. 2020. A survey of knowledge-enhanced text generation. *arXiv preprint arXiv:2010.04389*.

Shiyue Zhang and Mohit Bansal. 2019. Addressing semantic drift in question generation for semi-supervised question answering. 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 2495–2509, Hong Kong, China. Association for Computational Linguistics.

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, Brussels, Belgium. Association for Computational Linguistics.

Hao Zhou, Tom Young, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. 2018. Commonsense knowledge aware conversation generation with graph attention. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 4623–4629. International Joint Conferences on Artificial Intelligence Organization.

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.