BERT for Question Generation

Ying-Hong Chan
Department of Computer Science
National Chung Hsing University
Taichung, Taiwan
harry831120@gmail.com

Yao-Chung Fan
Department of Computer Science
National Chung Hsing University
Taichung, Taiwan
yfan@nchu.edu.tw

Abstract

In this study, we investigate the employment of the pre-trained BERT language model to tackle question generation tasks. We introduce two neural architectures built on top of BERT for question generation tasks. The first one is a straightforward BERT employment, which reveals the defects of directly using BERT for text generation. And, the second one remedies the first one by restructuring the BERT employment into a sequential manner for taking information from previous decoded results. Our models are trained and evaluated on the question-answering dataset SQuAD. Experiment results show that our best model yields state-of-the-art performance which advances the BLEU 4 score of existing best models from 16.85 to 21.04.

1 Introduction

Question generation (QG) task, which takes a context and an answer as input and generates a question that targets the given answer, have received tremendous interests in recent years from both industrial and academic communities (Zhao et al., 2018)(Zhou et al., 2017)(Du et al., 2017). The state-of-the-art models mainly adopt neural approaches by training a neural network based on the sequence-to-sequence framework. So far, the best performing result is reported in (Zhao et al., 2018), which advances the state-of-the-art results from 13.9 to 16.8 (BLEU 4).

The existing QG models mainly rely on recurrent neural networks (RNN) augmented by attention mechanisms. However, the inherent sequential nature of the RNN models suffers from the problem of handling long sequences. As a result, the existing QG models (Du et al., 2017)(Zhou et al., 2017) mainly use only sentence-level information as context. When applied to a paragraph-level context, the existing models show significant performance degradation. However, as indicated by (Du et al., 2017), providing paragraph-level information can improve QG performance. For handling long context, the work (Zhao et al., 2018) introduces a maxout pointer mechanism with gated self-attention encoder for processing paragraph-level input. The work reports state-of-the-art performance for QG tasks.

Recently, the NLP community has seen excitement around neural learning models that make use of pre-trained language models (Devlin et al., 2018)(Radford et al., 2018). The latest development is BERT, which has shown significant performance improvement over various natural language understanding tasks, such as document summarization, document classification, etc. In this study, we investigate the employment of the pre-trained BERT language model to tackle question generation tasks. We introduce two neural architectures built on top of BERT for question generation tasks. The first one is a straightforward BERT employment, which reveals the defects of directly using BERT for text generation. As will be shown in the experiment, naive employment of BERT offers poor performance, as, by construction, BERT produces all tokens at a time without considering decoding results in previous steps. Thus, we propose a sequential question generation model based on BERT as our second model for taking information from previous decoded results. Our model is simple but effective. We think this is a feature of BERT, as the power of BERT is able to simplify neural architecture design for natural language processing tasks. Our model outperforms the existing best models (Zhao et al., 2018) and pushes the state-of-the-art result from 16.85 to 21.04 (BLEU 4).

The rest of this paper is organized as follows. First, in Section 2, we review the BERT model which is the building block for our models. In Sec-

Proceedings of The 12th International Conference on Natural Language Generation, pages 173–177,
Tokyo, Japan, 28 Oct - 1 Nov, 2019. ©2019 Association for Computational Linguistics

tion 3, we introduce two BERT adaptions for QG tasks. Section 4 provides the performance evaluation and Section 5 concludes our findings and discuss future works.

2 BERT Overview

The BERT model is built by a stack of multi-layer bidirectional Transformer encoder (Vaswani et al., 2017). The BERT model has three architecture parameter settings: the number of layers (i.e., transformer blocks), the hidden size, and the number of self-attention heads in a transformer block. There are two BERT models with different model size released.

- **BERT$_{\text{base}}$**: 12 layers, 768 hidden dimensions and 12 attention heads (in transformer) with the total number of 110M parameters.
- **BERT$_{\text{large}}$**: 24 layers, 1024 hidden dimensions and 16 attention heads (in transformer) with the total number of 340M parameters.

For using BERT model, the input is required to be aligned as the BERTs specific input sequence. In general, a special token `[CLS]` is inserted as the first token for BERT's input sequence. The final hidden state of the `[CLS]` token is designed to be used as a final sequence representation for classification tasks. The input token sequence can be a pack of multiple sentences. To distinguish the information from different sentences, a special token `[SEP]` is added between the tokens of two consecutive sentences. In addition, a learned embedding is added to every token to denote whether it belongs to sentence A or sentence B. For example, given a sentence pair (s_i, s_j) where s_i contains $|s_i|$ tokens and s_j contains $|s_j|$ tokens, the BERT input sequence is formulated as a sequence in the following form:

$$X = (\texttt{[CLS]}, t_{i,1}, ..., t_{i,|s_i|}, \texttt{[SEP]}, t_{j,1} ..., t_{j,|s_j|})$$

The input representation of a given token is the sum of three embeddings: the token embeddings, the segmentation embeddings, and the position embeddings. Then the input representation is fed forward into extra layers to perform a fine-tuning procedure. BERT can be employed in three language modeling tasks: sequence-level classification, span-level prediction, and token-level prediction tasks. The fine-tuning procedure is performed in a task-specific manner. The details of our fine-tuning procedure are introduced in the later subsections.

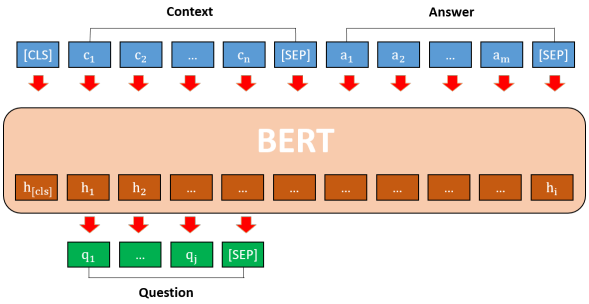

Figure 1: The BERT-QG architecture

3 BERT for Question Generation

3.1 BERT-QG

As an initial attempt, we first adapt the BERT model for QG as follows. First, for a given context paragraph $C = [c_1, ..., c_{|C|}]$ and an answer phase $A = [a_1, ..., a_{|A|}]$, the input sequence X is aligned as

$$X = (\texttt{[CLS]}, C, \texttt{[SEP]}, A, \texttt{[SEP]})$$

Let BERT() be the BERT model. We first obtain the hidden representation $\mathbf{H} \in \mathbb{R}^{|X| \times h}$ by $\mathbf{H} = \text{BERT}(X)$, where $|X|$ is the length of the input sequence and h is the size of the hidden dimension. Then, $\mathbf{H}$ is passed to a dense layer $\mathbf{W} \in \mathbb{R}^{h \times |V|}$ followed by a softmax function as follows.

$$Pr(w|x_i) = softmax(\mathbf{H} \cdot \mathbf{W} + \mathbf{b}), \forall x_i \in X$$

$$\hat{q}_i = \text{argmax}_w Pr(w|x_i)$$

The softmax is applied along the dimension of the sequence. All the parameters are fine-tuned jointly to maximize the log-probability of the correct token q_i. The model architecture is illustrated in Figure 1. As shown in the figure, we align a given context paragraph and a given answer as the input sequence and feed the input sequence into the BERT model to generate a sequence of tokens as a generated question.

3.2 BERT-SQG

In text generation tasks, as suggested by (Sutskever et al., 2014), considering the previous decoded results has significant impacts on the

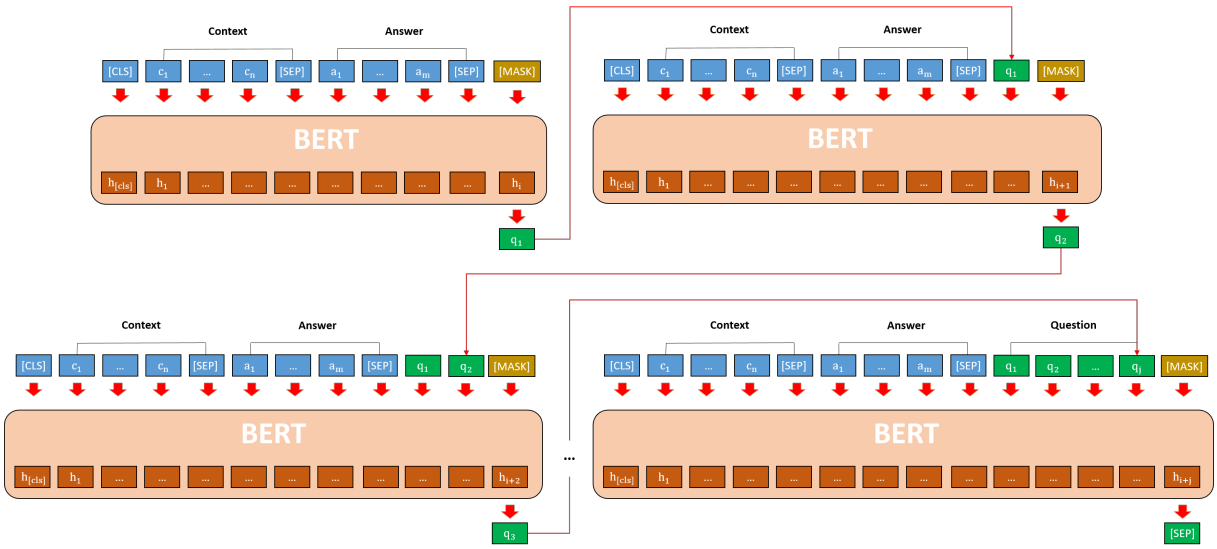

Figure 2: The BERT-SQG architecture

quality of the generated text. However, in BERT-QG, the token generation is performed without previously decoded result information. Due to this consideration, we propose a sequential question generation model based on BERT (called BERT-SQG).

In BERT-SQG, we take into consideration the previous decoded results for decoding a token. We adapt the BERT model for question generation as follows. First, for a given context paragraph $C = [c_1, ..., c_{|C|}]$ and an answer phase $A = [a_1, ..., a_{|A|}]$, and $\hat{Q} = [\hat{q1}, ..., \hat{q}_i]$ the input sequence X_i is formulated as

$$X_i = (\texttt{[CLS]}, C, \texttt{[SEP]}, A, \texttt{[SEP]}, \hat{q}_1, ..., \hat{q}_i, \texttt{[MASK]})$$

Then, the input sequence X_i is represented by the BERT embedding layers and then travel forward into the BERT model. After that, we take the final hidden state of the last token `[MASK]` in the input sequence. We denote the final hidden vector of `[MASK]` as $\mathbf{h}_{\texttt{[MASK]}} \in \mathbb{R}^h$. We adapt BERT model by adding an affine layer $\mathbf{W}_{\text{SQG}} \in \mathbb{R}^{h \times |V|}$ to the output of the `[MASK]` token. We compute the probabilities $Pr(w|X_i) \in R^{|V|}$ by a softmax function as follows.

$$Pr(w|X_i) = softmax(\mathbf{h}_{\texttt{[MASK]}} \cdot \mathbf{W}_{\text{SQG}} + \mathbf{b}_{\text{SQG}})$$

$$\hat{q}_i = \text{argmax}_w Pr(w|X_i)$$

Subsequently, the newly generated token $\hat{q}_i$ is appended into X and the question generation process is repeated (as illustrated in Figure 2) with the new X until `[SEP]` is predicted. We report the generated tokens as the predicted question.

4 Performance Evaluation

4.1 Datasets

The SQuAD dataset contains 536 Wikipedia articles and around 100K reading comprehension questions (and the corresponding answers) posed about the articles. Answers of the questions are text spans in the articles.

We follow the same data split settings as previous work on the QG tasks (Du et al., 2017)(Zhao et al., 2018) to directly compare the state-of-the-art results on QG tasks. Table 1 summarizes some statistics for the compared datasets.

- **SQuAD 73K** In this set, we follow the same setting as (Du et al., 2017); the accessible parts of the SQuAD training data are randomly divided into a training set (80%), a development set (10%), and a test set (10%). We report results on the 10% test set.
- **SQuAD 81K** In this set, we follow the same setting as (Zhao et al., 2018); the accessible SQuAD development data set is divided into a development set (50%), and a test set (50%).

4.2 Implementation Details

We use the PyTorch version of BERT [1] to train our BERT-QG and BERT-SQG models. The

[1] https://github.com/huggingface/pytorch-pretrained-BERT

Table 1: Dataset statistics: SQuAD 73K is the setting of (Du et al., 2017), and SQuAD 81K is the setting of (Zhao et al., 2018).

	Train	Test	Dev
SQuAD 73K	73240	11877	10570
SQuAD 81K	81577	8964	8964

pre-trained model uses the officially provided $\textbf{BERT}_{\textbf{base}}$ model (12 layers, 768 hidden dimensions, and 12 attention heads.) with a vocab of 30522 words. Dropout probability is set to 0.1 between transformer layers. The Adamax optimizer is applied during the training process, with an initial learning rate of 5e-5. The batch size for the update is set at 28. All our models use two TITAN RTX GPUs for 5 epochs training. We use Dev. data for epoch model to make predictions and select the highest accuracy rate as our score evaluation model. Also, in our BERT-SQG model, we use the Beam Search strategy for sequence decoding. The beam size is set to 3.

4.3 Model Comparison

In this paper, we compare our models with the best performing models (Du et al., 2017)(Zhao et al., 2018) in the literature. The compared models in the experiment are:

- **NQG-RC** (Du et al., 2017): A seq2seq question generation model based on bidirectional LSTM.
- **PLQG** (Zhao et al., 2018): A seq2seq network which contains a gated self-attention encoder and a maxout pointer decoder to enable the capability of handling long text input. PLQG model is the state-of-the-art models for QG tasks.

4.4 Evaluation Results

Table 2 shows the comparison results using sentence-level context and Table 3 shows the results on paragraph level context. We compare the models using standard metric BLEU and ROUGE-L ((Papineni et al., 2002)).

We have the following findings to note about the results. First, as can be observed, BERT-QG offers poor performance. In fact, the performance of BERT-QG is far from the results by other models. This result is expected as BERT-QG generates the sentences without considering the previous decoded results. However, when taking into account the previous decoded results (BERT-SQG), we effectively utilize the power of BERT and yield the state-of-the-art result compared with the existing RNN variants for QG. As shown in Table 2, BERT-SQG outperforms the existing best performing model by 2% on both benchmark datasets.

Second, the results in Table 3 further show that BERT-SQG successfully processes the paragraph-level contexts and further push the state-of-the-art from 16.85 to 21.04 in terms of BLEU 4 score. Note that NQG-RC and PLQG both use the RNN architecture, and the RNN-based models all suffer from the issue of consuming long text input. We see that the BERT model based on transformer blocks effectively addresses the issue of processing long text. The results of our BERT-SQG model are consistent in two data set and have achieved the best score at the paragraph level.

5 Conclusion

In this paper, we demonstrate that BERT can be adapted to question generation tasks. We concede that our BERT-SQG model is simple. However, we think this is a feature of BERT, as the power of BERT is able to simplify neural architectures design for specific tasks. While our model is simple, our model achieves state-of-the-art performance at both sentence-level and paragraph-level input and provides strong baselines for future research.

Acknowledgments

This work is supported in part by the Ministry of Science and Technology, Taiwan, under grant No: 107-2221-E-005-064-MY2.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

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

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.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language under-

Table 2: Comparison between our model and the published methods using sentence level context

	Model	BLEU 1	BLEU 2	BLEU 3	BLEU 4	METEOR	ROUGE-L
SQuAD 73K	NQG-RC	43.09	25.96	17.50	12.28	16.62	39.75
	PLQG	43.47	28.23	20.40	15.32	19.29	43.91
	BERT-QG	34.17	15.52	8.36	4.47	14.78	37.60
	BERT-SQG	**48.38**	**33.15**	**24.75**	**19.08**	**22.43**	**46.94**
SQuAD 81K	PLQG	44.51	29.07	21.06	15.82	19.67	44.24
	BERT-QG	34.18	15.51	8.57	4.97	14.57	37.65
	BERT-SQG	**50.18**	**35.03**	**26.60**	**20.88**	**23.84**	**48.37**

Table 3: Comparison between our model and the published methods using paragraph level context

	Model	BLEU 1	BLEU 2	BLEU 3	BLEU 4	METEOR	ROUGE-L
SQuAD 73K	NQG-RC	42.54	25.33	16.98	11.86	16.28	39.37
	PLQG	45.07	29.58	21.60	16.38	20.25	44.48
	BERT-QG	37.49	18.32	10.47	6.10	16.80	41.01
	BERT-SQG	**50.00**	**34.54**	**25.98**	**20.11**	**23.88**	**48.12**
SQuAD 81K	PLQG	45.69	30.25	22.16	16.85	20.62	44.99
	BERT-QG	32.61	14.50	7.70	4.08	14.18	37.94
	BERT-SQG	**50.89**	**35.49**	**26.87**	**21.04**	**24.25**	**48.66**

standing by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/research-covers/languageunsupervised/language understanding paper. pdf*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

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.

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.