Question Generation: Finetuned on Pretrained with QA-pair Evaluation

***Wang, Shixiang**
sxwang6-c@my.cityu.edu.hk

***Chen, Shiwei** *
shiwechen6-c@my.cityu.edu.hk

City University of Hong Kong
July 6, 2022

1 Introduction

Asking a question is a basic component of conversations. Asking fluent and appropriate questions is quite natural to humans but not for machines. Automatic Question generation is a task that requires the machine to generate not only linguistically fluent but also valid questions against a given context. The possible choices for context include but are not limited to the knowledge base, data tables, text corpus, and images. More specifically, in the realm of Natural Language Processing (Hereinafter referred to as NLP), Automatic Question generation falls under the Natural Language Understanding tasks category. The task's framework usually demands a context, often a paragraph of text, to generate a question whose answer should be a span of the context. A specific answer for the to-be-generated question can be provided to make a variant of this framework, where output questions are expected to be more accurate and specific.

Besides its popularity in the research community, Automatic Question Generation (Hereinafter referred to as QG) has great potential in various fields. For example, in the Education Technology business, a QG model can be capable of generating straightforward exams consisting of multiple choices, true-false, and fill-in-the-blank questions. Another similar scenario can be found in every company when an important document is updated. QG can always generate quick quizzes to ensure everybody has a full understanding of the changes. Moreover, QG can be part of other NLP tasks as well. For example, instead of passively waiting to be triggered, a dialogue system empowered by QG can lead a conversation with questions based on the previous context.

In the early days, the proposed algorithms of QG were mostly rule-based. [Mitkov and Ha, 2003] describes a multiple-choice tests generation system based on text corpus with the aid of term extraction and shallow parsing. [Heiman and Smith, 2010] apply logistic regression and manually written rules to

*Indicates Equal Contributions.

rank overly generated questions. In recent years, the advancement of the neural network has brought the study of QG into a new era. Using the Knowledge Base formulation, [Serban et al, 2016] trains a neural network to generate triplet questions. Finetuning on pre-trained models has also achieved impressive performance with the help of large-scale question-answer databases, such as SQuAD [Rajpurkar et al, 2016, Rajpurkar et al, 2018]. These pre-trained models can be roughly categorized as encoder-only models such as BERT [Devlin et al, 2019], decoder-only models such as GPT [Radford et al, 2018], and encoder-decoder models such as BART [Lewis et al, 2020] and T5 [Raffel et al, 2020]). In this project, We use the pre-trained BART-Large and finetune it on the downstream QG task. Then we finetune a pretrined BERT on the question-answer pairs to serve as the evaluator on the candidate questions predicted by BART and T5. The results show that the pre-trained models achieved high performance on the dataset and a well-calibrated evaluator has a positive effect on it.

2 Related Works

2.1 The SQuAD

SQuAD is the Stanford question answering dataset. There are two different versions: SQuAD and SQuAD_v2 until now. SQuAD [Rajpurkar et al, 2016] has total 87599 different data. Passages are selected from English Wikipedia, usually 100 to 150 words. The questions are crowd-sourced. The type of each data is a dictionary, and it has five keys: 'id,' 'title,' 'context,' 'question,' and 'answers.' The answer to every question is a segment of context, or span, from the corresponding reading passage. SQuAD_v2 [Rajpurkar et al, 2018] has total 130319 data. The biggest difference between SQuAD_v2 and squad is that squad_v2 has about 50,000 unanswerable questions. In this project, we are going to examine the effectiveness of pre-trained on SQuAD since we need an answer-guided dataset framework.

2.2 Pretrained Models

2.2.1 BERT

BERT represents bidirectional encoder representation from transformers [Devlin et al, 2019]. It uses the encoder of the transformer since the decoder cannot obtain the information to be predicted. The main innovations of the model are in the pre-training method, which uses Masked LM and Next Sentence Prediction to capture word- and sentence-level representations, respectively. Compared to GPT (Generative pre-trained transformer), BERT is bi-direction which captures more contextual information. Compared to ELMo, Although both are bidirectional, their objectives are different. ELMo trains two 'independent' LSTM to gain contextual information while BERT applies attention mechanism. From the perspective of the network structure, Both GPT and ELMo are auto-regressive models, which slows their training process. At the same time, BERT can be trained in parallel benefits on the multi-head attention mechanism.

The first pre-training task in BERT is Masked language modeling (MLM). The primary motivation of this

task is to enable bidirectional pre-training. The author randomly chooses 15% of the token positions for prediction in the original paper. The chosen token has 80% probability of being replaced with [MASK], 10% probability becomes a random token, and 10% keep it as it is. The masking scheme here is to adapt the model to downstream tasks better as there is no [MASK] token in the fine-tuning part. The other pre-train task is Next sentence prediction (NSP). This task is designed to make the model understand the relationship between two sentences, such as if sentence B is the following sentence of A. During the training process, around 50% of the time B is the following actual sentence of A. otherwise, it is a random sentence.

BERT is quite good at NLU tasks such as classification tasks and question answering tasks. However, As BERT uses the bidirectional information during the pre-training process, it is not easy to fine-tune BERT to adapt to NLG tasks which can only use one direction information, such as question generation.

2.2.2 BART

BART represents a denoising auto-encoder for pre-training seq2seq models [Lewis et al, 2020]. It combines the bidirectional and auto-regressive transformers. The pre-training process of BART has two stages: it first corrupts the input text using different ways and then learns a seq2seq model to rebuild the original text. In BERT, it just replaces the chosen token. This simple replacement results in the input on the encoder side may carry some information about the sequence structure, such as the sequence length, which is generally not provided to the model in text generation tasks. However, BART uses a more diverse noise intended to destroy this information about the sequence structure and prevent the model from "depending on" such information. BART is outstanding in generating tasks based on the one-way decoder. Following lists some nosing functions BART uses, which are also illustrated in figure 1.

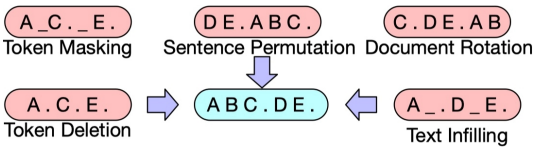

Figure 1: Nosing functions used by the BART model from [Lewis et al, 2020]

- **Token Masking**: This is same as BERT, randomly replaces a token with MASK. The purpose is to train the model's ability to infer a single token.
- **Token Deletion**: Randomly delete some tokens. The purpose is to train the model's ability to infer tokens and their locations.
- **Text Infilling**: Randomly replace continuous tokens with a [MASK], and the length of the span obeys the poisson distribution. Note that a span length of zero is equivalent to inserting a [MASK]. The purpose is to train the model's ability to infer how many tokens a span corresponds to.
- **Sentence Permutation**: Scramble the sentences of a document. The purpose is to train the model's ability to reason about relationships between sentences.
- **Document Rotation**: Randomly select a token from the sequence of documents, and make that token the beginning of the document. The purpose is to train the model to find the ability to start the document.

2.2.3 T5

T5 is the text-to-text Transfer Transformer proposed by Google Research [Raffel et al, 2020]. The model is pre-trained based on a BERT-base size encoder-decoder transformer on the common crawl web extracted text[1]. Finetuning for different downstream tasks is done by prefixing, which is shown in figure 2. Three kinds of attention masks are used in T5, including the Fully-visible Mask where every output entry can observe the entire input, the Casual Mask where the future is blocked from being observed, and the Casual Prefix Mask, which is the same as the Casual Mask but the task prefix is allowed to be observed. Three pretraining objectives are explored, including Language Modeling (next word prediction), BERT-style (predict randomly masked word/span), and Deshuffling (predict original text of shuffled text). Corruption strategies such as Mask, Span Replace, and Drop are applied to enhance robustness.

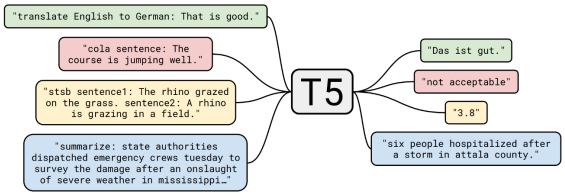

Figure 2: Diagram of the T5 text-to-text framework with downstream tasks considered: translation, question answering, and classification. From [Raffel et al, 2020].

2.3 NLG Metrics

2.3.1 BLEU Score

BLEU represents **Bi**lingual **E**valuation **U**nderstudy [Papineni et al, 2002]. It compares the machine translation with reference human translation based on the n-gram precision and brevity penalty. The calculation formula is as follows.

$$S_{\text{BLUE}} = \exp\left(\min\left(0, 1 - \frac{\text{len}_{\text{label}}}{\text{len}_{\text{pred}}}\right)\right) \prod_{n=1}^{k} p_n^{1/2^n}$$

p_n is the n-gram precision. It gives high weight to the long matches. The first term is to penalize the too short translations. BLUE is instrumental, but it has some disadvantages. There may be many valid translations, and a good translation may get a poor score as it has a low-gram overlap compared to the reference.

2.3.2 ROUGE Score

ROUGE is the abbreviation of **R**ecall-**O**reiented **U**nderstudy of **G**isting **E**valuation [Lin, 2004]. In this project we use the ROUGE-L score, which measures longest matching sequence of words using the Longest common subsequence (LCS). An advantage of using LCS is that it does not require consecutive matches but in-sequence matches that reflect sentence level word order. Since it automatically includes longest in-sequence common n-grams, you don't need a predefined n-gram length.

$$S_{\text{ROUGE-L}} = \frac{(1+\beta^2) R_{\text{lcs}} P_{\text{lcs}}}{R_{\text{lcs}} + \beta^2 P_{\text{lcs}}}$$

[1] Common Crawl's web archive consists of petabytes of data collected since 2011. The data are collected by crawlers and recorded every month.

Where $R_{\text{lcs}} = \frac{\text{LCS}(X,Y)}{m}$ and $P_{\text{lcs}} = \frac{\text{LCS}(X,Y)}{n}$, X is the candidate and m is the length of it, Y is reference and n is the length of it, $LCS(X,Y)$ is the longest common subsequence of X and Y.

2.3.3 METEOR Score

METEOR stands for **M**etric for **E**valuation of **T**ranslation with **E**xplicit **OR**dering and it is proposed by [Banerjee and Lavie, 2005]. It was designed as a fix for the BLEU metric and also generate good correlation with human judgement at segment level. This differs from the BLEU who focus on corpus level. It can be defined as follows.

$$S_{\text{METEOR}} = F_{\text{mean}}(1-p)$$

Where $p = 0.5(\frac{c}{u_m})^3$ is a text segment's penalty, c is the number of chunks and u_m is the number of mapped unigrams. $F_{\text{mean}} = \frac{10PR}{R+9P}$ is the harmonic mean of the unigram's precision and recall. It also has several features that are not found in other metrics, such as stemming and synonymy matching, along with the standard exact word matching.

3 Experiment and Result

First, we use BART to do pre-training. The data set we choose is the squad. The tokenizer is also pre-trained, which has two different versions. One is called Bart-base, and the other one is bart-large. BART-base uses six layers of encoder and decoder while BART-large doubles both the encoder and decoder layer. In this project, we choose to use bart-large. As the resource-limited, we only use ten thousand samples to fine-tune the model. The learning rate cannot be very large. Otherwise the training process will be unstable and may miss the global optimal solution. The optimizer we applied is AdamW which is pretty good for summarization tasks such as question generation. Another technique we use in the training part is 'Warm up'. Since the weights of the model are randomly initialized at the beginning of training, if a larger learning rate is selected at this time, it may cause instability (oscillation) of the model. Selecting 'Warm up' to warm the learning rate can make the learning rate in several epochs or some steps at the beginning of training is small. Under the small learning rate, the model can gradually become stable. After the model is relatively stable, select the preset learning rate for training, so that the model converges Faster and better. We train the model in four epochs and the average loss of each epoch can be seen in Figure 3. We can find that the loss becomes stable after about three epochs.

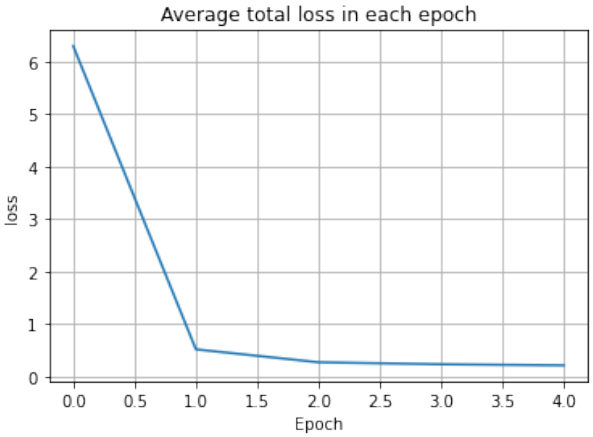

Figure 3: Average Loss in each Epoch of finetuning BART.

For the test part, we first load the pre-trained model. Its size is about 1.51 GB. We have opened the source of the model, and it can be downloaded here. After loading the model, we then set the test data.

	BlEU-1	BLEU-2	BLEU-4	ROUGE-L	METEOR
BART	39.97	27.69	12.86	47.88	43.19
T5	39.45	27.17	12.47	46.81	43.54

Table 1: Metrics Evaluation Scores for Pretrained BART and T5.

The test data comes from the same data set as the training process but uses the validation part. We used five hundred samples to test our model. We only keep one sample of those who have the same contexts. We also applied the beam search technique during the process of generating questions. The performance of the trained BART is compared with a T5 model from HuggingFace[2], which is also finetuned on the same dataset, based on the NLG metrics. The result is pretty impressive and is shown in table 1.

	BlEU-1	BLEU-2	BLEU-4	ROUGE-L	METEOR
BART	39.97	27.69	12.86	47.88	43.19
T5	39.45	27.17	12.47	46.81	43.54
Ensemble	**40.92**	**28.47**	**13.35**	**48.84**	**44.4**

Table 2: Metrics Evaluation Scores for Pretrained BART, T5, and BERT-Evaluator Ensemble.

To evaluate question-answer pairs so that we can select the best outputs from the two models, we use a pre-trained BERT-base with a Next Sentence Prediction head to fine-tune on the SQuAD v1 training set. The idea is quite intuitive as we can treat the question as to the preceding sentence and the answer as the following sentence. The negative sample is generated by randomly selecting question-answer pairs (Q_i, A_j) where $i \neq j$. In total, 35,040 samples are used for fine-tuning the BERT-based model on a 6GB RAM GTX1060 for ten epochs using the AdamW optimizer mentioned above with a learning rate of 5e−5. The results in table 2 show that the ensemble method with a finetuned BERT evaluator performs better on all NLG metrics.

4 Future Works

From the perspective of research, we think that answer-unaware question generation is quite interesting. In the answer-guided framework, the exposure bias of the model is the ambiguous answer can be part of the expected output as well. Other NLU tasks include Topic Extraction, Named Entity Recognition can be integrated into solving this objective to optimize the task. Moreover, a less task-reliant Question Answer Evaluator might also be an interesting topic.

From the perspective of engineering, generating different types of question, such as true/false question, multiple choice question, fill-in-the-blanks quesiton, etc., can be a commercially promising task to tackle as the task is natural to the Education Industry. Other related problems are interesting too, for instance, how to generate reasonable wrong answers.

[2]HuggingFace is a company that first built a chat app for bored teens provides open-source NLP technologies. Link:https://huggingface.co/

References

[Mitkov and Ha, 2003] Ruslan Mitkov and Le An Ha. 2003. Computer-Aided Generation of Multiple-Choice Tests. In Proceedings of the HLT-NAACL 03 Workshop on Building Educational Applications Using Natural Language Processing, pages 17-22.

[Heiman and Smith, 2010] Michael Heilman and Noah A. Smith. 2010. Good Question! Statistical Ranking for Question Generation. In Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, pages 609-617, Los Angeles, California. Association for Computational Linguistics.

[Serban et al, 2016] Iulian Vlad Serban, Alberto García-Durán, Caglar Gulcehre, 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.

[Rajpurkar et al, 2016] 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.

[Rajpurkar et al, 2018] Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know What You Don't Know: Unanswerable Questions for SQuAD. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pages 784-789, Melbourne, Australia. Association for Computational Linguistics.

[Radford et al, 2018] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018, Improving language understanding by generative pre-training.

[Devlin et al, 2019] 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.

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

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

[Papineni et al, 2002] Papineni, K.; Roukos, S.; Ward, T.; Zhu, W. J. (2002). BLEU: a method for automatic evaluation of machine translation (PDF). ACL-2002: 40th Annual meeting of the Association for Computational Linguistics. pp. 311-318.

[Lin, 2004] Lin, Chin-Yew. 2004. ROUGE: a Package for Automatic Evaluation of Summaries. In Proceedings of the Workshop on Text Summarization Branches Out (WAS 2004), Barcelona, Spain, July 25 - 26, 2004.

[Banerjee and Lavie, 2005] Banerjee, S. and Lavie, A. (2005) "METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments" in Proceedings of Workshop on Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization at the 43rd Annual Meeting of the Association of Computational Linguistics (ACL-2005), Ann Arbor, Michigan, June 2005