Rodrigo, M. M. T. et al. (Eds.) (2021). Proceedings of the 29th International Conference on Computers in Education. Asia-Pacific Society for Computers in Education

Short Answer Questions Generation by Fine-Tuning BERT and GPT-2

Danny C.L. TSAI[a], Willy J.W. CHANG[a] & Stephen J.H. YANG[a*]
[a] *Department of Computer Science & Information Engineering, National Central University, Taiwan*
*stephen.yang.ac@gmail.com

Abstract: In educational research, artificial intelligence (AI) is suitable for many situations, such as exploring student learning paths and strategies. However, most of them cannot reduce the workload of teachers. In the course, teachers need to spend a lot of effort on setting exams, because exams are the most direct way to understand students' learning performance. In this research, we use modern artificial intelligence model, BERT and GPT-2 to generate questions to reduce the work of teachers frequently setting questions. The type of questions we generate is short answer questions. The main reason is that many researches prove that short-answer questions can enhance students' long-term memory and improve learning performance. We also compare the performance of BERT before and after fine-tuning. The results show that BERT can be used for general reading comprehension questions before fine-tuning, but in the field of domain knowledge, fine-tune BERT's performance is better.

Keywords: Automatic Question Generation, BERT, Keyword Extraction, Learning Performance

1. Introduction

The continuous development of computer and information communication technology has led to the development of artificial intelligence. With the adoption and use of new technologies in education, artificial intelligence has also been widely used in the field of education (Chen, Chen, & Lin, 2020). Intelligent Tutoring System (ITS) is the most common implementation studied in AIED (du Boulay, 2016). For example, provide teachers with automatic evaluation of performance, scoring, and provide feedback to students to ensure continuous improvement of learning (Sharma, Kawachi, & Bozkurt, 2019), or intervening with students at risk or providing feedback and instructional content (Tsai & Gasevic, 2017).

In the education field, various benefits can be gained through short-answer test: (1) Let students construct knowledge through practice questions, (2) Identify wrong concepts through learner feedback, (3) Repeat important concepts to enhance memory, (4) Let the teacher understand the learning situation of each learner (Kurdi, Leo, Parsia, Sattler, & Al-Emari, 2020). In this paper, we use artificial intelligence technology to help teachers to reduce the time cost for setting questions. In the past, setting question was almost an irreplaceable task, because exams and tests are a basic educational evaluation tool, the purpose is to confirm whether students understand the knowledge of the course contents, but setting test questions requires the teacher's professionalism and experience. Fortunately, with advances in the hardware and methods of training neural networks, many neural network models that use abundant data for training have emerged in recent years, and the accuracy of these models in natural language processing tasks has also been significantly improved, such as BERT (Bidirectional Encoder Representations form Transformers) (Devlin, Chang, Lee, & Toutanova, 2018), GPT (Generative Pre-Training) (Radford, Narasimhan, Salimans, & Sutskever, 2018), T5 (Text-to-Text Transfer Transformer) (Raffel et al., 2019), etc. These are pre-trained models by a large amount of unlabeled data, so these models have a certain understanding of natural language.

In this paper, we use two powerful pre-trained model BERT and GPT-2 to generate short answer question. We apply this automatic question generation system in a college python programming course, and use the automatically generated questions as students' after-class exercises. To make the

question generated by the model more in line with the course knowledge, we fine-tune the BERT and evaluate. Finally, the research questions of the study are defined as follows:

- RQ1: Does BERT need to fine-tune according to domain knowledge?
- RQ2: Can the model's performance be improved after fine-tuning?

2. Literature Reviews

2.1 Question Generation (QG)

Question generation (QG) is one of the research goals in the field of natural language processing. There are three methods for question generation, which are syntax-based, semantics-based and template-based. At present, among the methods of question generation, methods based on syntax and semantics account for more than 70% of the total. The other methods are limited by sentence patterns (Kurdi et al., 2020). Therefore, we only considered syntax-based and semantic-based approaches in this study and propose an ensemble method that combines semantic and syntactical approaches to automatically generate questions.

With the development of deep learning, more and more research implement question generation by neural network. Du, Shao, and Cardie (2017) introduce an attention-based sequence learning model for the QG task, and the question generated by their system are also rated as being more natural and as more difficult to answer. In the application of question generation. Krishna and Iyyer (2019) implemented a SQUASH (Specificity-controlled Question-Answer Hierarchies) pipeline to generate question-answer pairs, they use GPT-2 to convert the input sentence into a question, and BERT is used to answer the question generated by GPT-2 as an output answer. Alberti, Andor, Pitler, Devlin, and Collins (2019) use BERT to introduce a generating synthetic question answering corpora by combining models of question generation and answer extraction.

3. Methodology and Experiments

This research proposes a combination of syntax-based and semantic-based Automatic Question Generation (AQG) system, using BERT (Devlin et al., 2018) for semantic analysis and Stanford CoreNLP (Manning et al., 2014) for syntax analysis, the construction question by GPT-2 (Radford et al., 2018) as Figure 1 shows. We use this automatic question generation system in a python programming course. Questions will be generated by Automatic question generation system according to the content of the class then teacher can select or revise the questions they want for students' after-class practice.

3.1 Automatic Question Generation

The main purpose of semantic analysis is to allow the machine to understand the content of the teacher's textbook and extract important keywords from the textbook. We use BERT to extract the keywords in the textbook. Through unsupervised learning and transfer learning method, using a large amount of unlabeled data to train the model, so BERT can quickly understand natural language and semantics, without the need for developers to collect a lot of data.

After keywords extraction, we use syntax analysis to extract the sentences containing keywords. In this study, we use Standford CoreNLP, a tool developed by Stanford University to preform syntactic analysis, and the results will be parse tree. To find the complete sentence, we use the analysis tree to find the sentence that contains the subject, verb, and object as shown in *Figure 2*. If the label under "root" is "S", it means the input is complete sentence. After syntactic analysis, we get several complete sentence containing keywords from course textbook, and these sentences will be the answer of the question generated by machine. Finally, we input the complete sentences to GPT-2. The main function of GPT-2 is that predicts the next word of the text, and passes the current output back to

the input for repeated predictions. Therefore, we use GPT-2 to generate question according to the input sentences. Finally, teacher chooses which questions can be used.

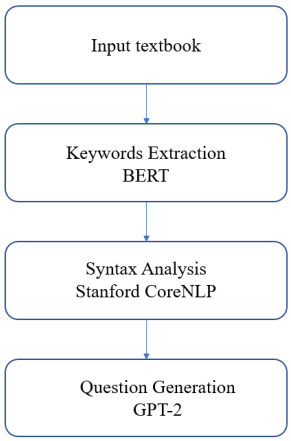

Figure 1. Question generation process.

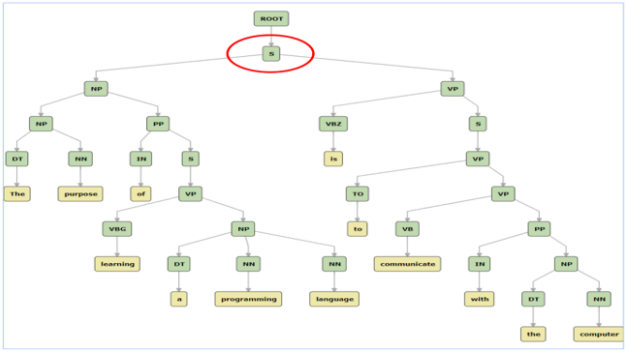

Figure 2. Parse tree of the complete sentence.

3.2 Fine-Tuning BERT

To enable BERT to automatically mark the keywords of the python programming course textbook, we use the "Kaggle-Python Questions from Stack Overflow" data set which contains 607,282 sentences and keywords to fine-tune the model. First, we use the BIO format (Ramshaw & Marcus, 1999) to label the keywords in the sentence. The label B (beginning) indicates that the letter is the beginning of the keyword; the label I (inside) indicates that the letter is part of the keyword; the label O (outside) indicates that the letter is not part of keyword. We also use the early stopping mechanism to avoid overfitting. Figure 3 shows the training loss and validation loss during we fine-tune the model. We can find that the model has the best performance when the Epoch is 3 (Train loss is 0.051, Validation loss is 0.052), and then the model starts to over-fitting. Figure 4 refers to the accuracy of the model when labeling B, I, and O. The model has good performance when Epoch 3 (Accuracy is 0.9778) and Epoch 4 (Accuracy is 0.978), but because the model will over-fitting after Epoch 3, so we choose epoch 3 model to extract keywords from course textbook.

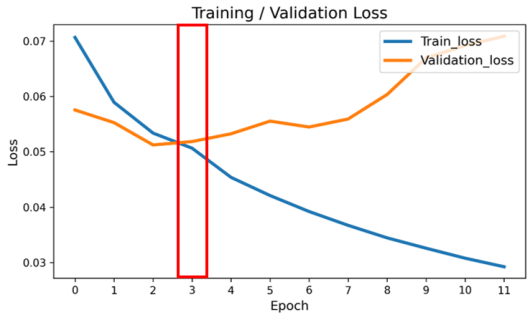

Figure 3. Train Loss and Validation Loss during Fine-Tuning.

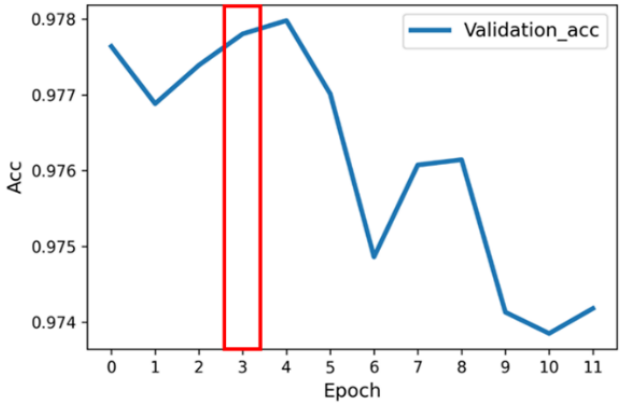

Figure 4. Accuracy during Fine-Tuning.

4. Result and Discussion

4.1 Model's Performance Before and After Fine-Tuning

To evaluate the performance of the model before and after fine-tuning, we use a confusion matrix to calculate the accuracy. Accuracy represents the proportion of keywords correctly predicted by the model. The closer the accuracy is to 1, the more accurately the model can predict important keywords in the text. Conversely, if Accuracy is close to 0, it means that the quality of keywords extracted by the model needs to be strengthened. Figure 5 shows the accuracy of the model before and after fine-tuning. The average accuracy before fine-tuning is 94%, and accuracy after fine-tuning is 98%. We can find that the model has good performance before fine-tuning, we think this is due to the large amount of data during pre-training which allows BERT to learn knowledge in different fields. However, after fine-tuning through the data set of python knowledge, the model will learn more knowledge about python programming, so that the model has better performance.

We compare the prediction of the model on the validation set before and after fine-tuning. The prediction is wrong before the fine-tuning, but correct after the fine-tuning, as shown in Table 1. Although the accuracy of the model before fine-tuning is as high as 94%, but we can see the example in Table 1. The model does not really find keywords related to python before fine-tuning. After fine-tuning, BERT has learned concept about python, so it has better performance.

Finally, we input the textbook used in the course into the two models, and then compare the keywords predicted by the two models and the QA pairs generated by the two models then ask the instructor to evaluate whether the fine-tuned model performs better. The result shown in Table 2 and Table 3. We can see keywords extracted by two models in Table 2, BERT find better keywords like "if condition", "choices" in the "if…else" textbook. The teacher thinks that the model after fine-tuning can extract the keywords that are more in line with the content of the course, which means that the BERT can improve the performance of the model after fine-tuning. Table 3 shows the QA pair generated by the model based on the "if condition" textbook. Since the keywords found before and after fine-tune are different, the generated QA pairs are also different.

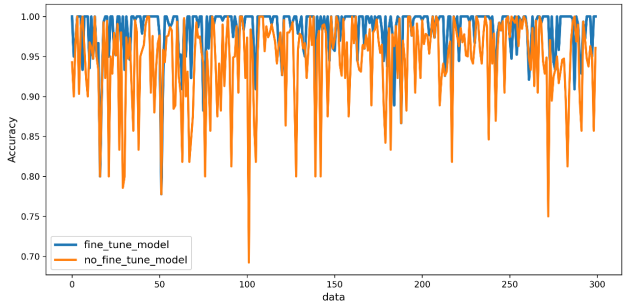

Figure 5. Model's Accuracy before and after Fine-Tuning.

Table 1. *Wrong prediction before Fine-Tuning, correct after Fine-Tuning (red words mean the keywords of the sentences, "X" means that the model did not find the keyword)*

Sentences	Prediction before fine-tuning	Prediction after fine-tuning
I writing an app in **python** for **google app engine** where each user can submit a post and each post has a ranking which is determined by its votes and comment count.	X	python、google app engine
I have **postgresql** db which I am updating with around 100000 records.	X	postgresql
I am using **Popen** to call a shell script that is continuously writing its stdout and stderr to a log file.	X	Popen
I know how to make question boxes in **HTML**, but not how to get the asked question to print.	X	HTML

Table 2. *The keywords of "If Condition" textbook*

Prediction keywords before fine-tuning	Prediction keywords after fine-tuning
Judgment	if condition
Conditional	conditional
Indentation	Indentation
Relation	Choices
	Operand

Table 3. *QA pair of "If Condition" textbook*

Content of "if condition" textbook	QA pair generated by fine-tuned model	QA pair generated by non-fine-tuned model
The if…elif…else statement is used in Python for decision making. Here, the program evaluates the test expression and will execute statement(s) only if the test expression is True. If the test expression is False, the statement(s) is not executed. In Python, the body of the if statement is indicated by the indentation. The body starts with an indentation and the first unindented line marks the end. Python interprets non-zero values as True. None and 0 are interpreted as False. The if…else statement evaluates test expression and will execute the body of if only when the test condition is True.	Q: What language does the body of the if statement come from? A: Python	Q: What happens if the test expression is False? A: If the test expression is False, the statement is not executed
	Q: What happens when the else statement evaluates? A: else statement evaluates test expression and will execute the body of if only when the test condition is True	Q: What happens when the program evaluates the test expression? A: Here, the program evaluates the test expression and will execute statement only if the test expression is True
	Q: What is the body of if? A: In Python, the body of the if statement is indicated by the indentation	Q: What is the body of if? A: In Python, the body of the if statement is indicated by the indentation
	Q: What does False mean? A: None and 0 are interpreted as False	

5. Conclusion

Nowadays, the application of AI is ever-changing. This research focuses on natural language processing, aiming to realize the application of AI in the educational field, and expects that emerging AI technologies can bring benefits to education. This research is based on advanced machine learning technology and proposes an automatic question generation system (AQG system). We use BERT to extract keywords in the textbook, and use Stanford CoreNLP to filter out the key sentences. Finally, GPT-2 generates the questions for teacher to select or revise the questions they want.

BERT is a general language model that is widely used in various fields. In the experiment, we compared the model's performance in predicting keywords before and after fine-tuning. We found that after the model was fine-tuned, it can better understand important concepts in the programming language field. It can also improve the effectiveness of the model.

References

Alberti, C., Andor, D., Pitler, E., Devlin, J., & Collins, M. (2019). Synthetic QA corpora generation with roundtrip consistency. *arXiv preprint arXiv:1906.05416*.

Chen, L., Chen, P., & Lin, Z. (2020). Artificial intelligence in education: A review. *Ieee Access, 8*, 75264-75278.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

du Boulay, B. (2016). Artificial intelligence as an effective classroom assistant. *IEEE Intelligent Systems, 31*(6), 76-81.

Du, X., Shao, J., & Cardie, C. (2017). Learning to ask: Neural question generation for reading comprehension. *arXiv preprint arXiv:1705.00106*.

Krishna, K., & Iyyer, M. (2019). Generating question-answer hierarchies. *arXiv preprint arXiv:1906.02622*.

Kurdi, G., Leo, J., Parsia, B., Sattler, U., & Al-Emari, S. (2020). A systematic review of automatic question generation for educational purposes. *International Journal of Artificial Intelligence in Education, 30*(1), 121-204.

Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J. R., Bethard, S., & McClosky, D. (2014). *The Stanford CoreNLP natural language processing toolkit.* Paper presented at the Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations.

Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving language understanding by generative pre-training.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., . . . Liu, P. J. (2019). Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Ramshaw, L. A., & Marcus, M. P. (1999). Text chunking using transformation-based learning *Natural language processing using very large corpora* (pp. 157-176): Springer.

Sharma, R. C., Kawachi, P., & Bozkurt, A. (2019). The landscape of artificial intelligence in open, online and distance education: Promises and concerns. *Asian Journal of Distance Education, 14*(2), 1-2.

Tsai, Y.-S., & Gasevic, D. (2017). *Learning analytics in higher education---challenges and policies: a review of eight learning analytics policies.* Paper presented at the Proceedings of the seventh international learning analytics & knowledge conference.