Training Generative Question-Answering on Synthetic Data Obtained from an Instruct-tuned Model

Kosuke Takahashi, Takahiro Omi, Kosuke Arima
Stockmark
kosuke.takahashi, takahiro.omi, kosuke.arima@stockmark.co.jp

Tatsuya Ishigaki
National Institute of Advanced Industrial Science and Technology
ishigaki.tatsuya@aist.go.jp

Abstract

This paper presents a simple and cost-effective method for synthesizing data to train question-answering systems. For training, fine-tuning GPT models is a common practice in resource-rich languages like English, however, it becomes challenging for non-English languages due to the scarcity of sufficient question-answer (QA) pairs. Existing approaches use question and answer generators trained on human-authored QA pairs, which involves substantial human expenses. In contrast, we use an instruct-tuned model to generate QA pairs in a zero-shot or few-shot manner. We conduct experiments to compare various strategies for obtaining QA pairs from the instruct-tuned model. The results demonstrate that a model trained on our proposed synthetic data achieves comparable performance to a model trained on manually curated datasets, without incurring human costs.

1 Introduction

Fine-tuning large language models (LLMs) has been proven effective for enhancing question-answering systems (Dong et al., 2019). However, extending this approach to languages other than English presents challenges due to the scarcity of adequate QA pairs for training. In this study, we specifically target Japanese as a representative non-English language. We propose a straightforward approach that synthesizes Japanese QA pairs using an instruct-tuned model.[1]

Question-answering tasks can be categorized into two main settings: questions with context and without context (Kurihara et al., 2022). In this study, we focus on the context-based setting as shown in Figure 1. In this setting, the system takes a question along with the accompanying context as input. The model generates an answer by utilizing the information provided within the context.

[1]Our experiments utilize OpenAI's ChatAPI with the *gpt-3.5-turbo-0613* model.

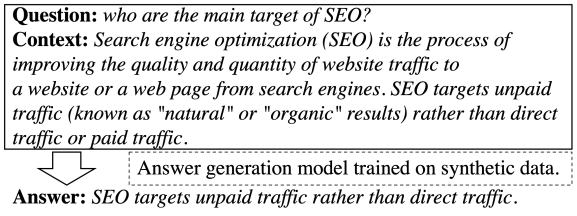

Figure 1: The task of the generative context-aware QA.

On the other hand, the setting without context involves the system processing only the question as input.

We present a straightforward yet cost-effective method for generating synthetic question-answer (QA) pairs. Existing QA systems are trained on either human-authored datasets or automatically generated QA pairs (Sachan and Xing, 2018; Tang et al., 2018), both leading to high labor costs. By contrast, this paper investigates utilizing an instruct-tuned model inspired by their reasonable ability to produce synthetic dataset (Gilardi et al., 2023). We use a context as input and generate both the corresponding question and its answer. The instruct-tuned model allows us to produce QA pairs in a zero-shot or few-shot manner, eliminating the need for manual curation.

Our experiments compare question-answering systems fine-tuned on synthetic data generated through various strategies. Specifically, we explore different sources of contexts, the number of shots fed into the instruct-tuned model, and the quantity of QA pairs generated. The evaluation on JSQuAD's evaluation dataset (Kurihara et al., 2022) provides three findings. Firstly, employing contexts extracted from a corpus with similar characteristics to the evaluation dataset yields improved performance. Secondly, the one-shot strategy outperforms the zero-shot approach. Lastly, generating three QA pairs for each context is more effective than generating a lower number of QA pairs. The top-performing model fine-tuned on our

synthetic data exhibits comparable performance to models trained on manually curated data.

2 Related Work

Existing QA focus on two major settings: "closedQA" with context and "commonsens-QA" without context (Kurihara et al., 2022). For the former, which we target, the QA systems receive a question along with a context, such as a Wikipedia article, and generate an answer. On the other hand, in the latter setting, the systems only receive a question as input.

There are two types of QA systems: extractive and generative. Extractive methods extract an answer as it is from the context by models like BERT (Rajpurkar et al., 2016), while generative methods often use the expressions that are not in the context by models like T5 (Raffel et al., 2020) or GPT (Brown et al., 2020). Our focus is on the latter.

While several manually created datasets exist in English, such as SQuAD (Rajpurkar et al., 2016) and QuALITY (Pang et al., 2022), these resources do not directly apply to the Japanese language. For Japanese, JSQuAD (Kurihara et al., 2022) and JAQKET[2] are available. We use JSQuAD[3] because the evaluation data of JAQKET is not public.

Existing studies synthesize QA pairs by two main approaches: supervised (Lee et al., 2020; Sachan and Xing, 2018; Tang et al., 2018) and unsupervised (Puri et al., 2020). The supervised approaches train question-answer generators using manually created datasets. Our approach generates QA pairs from contexts in a zero-shot or few-shot manner, eliminating the need to train generators. In the unsupervised approach, Puri et al. (2020) uses a named entity recognizer (NER) for answer candidate extraction while our approach uses only an instruct-tuned model in end-to-end and does not require NER.

3 Synthesizing QA Pairs

We describe our approach in this section.

[2] https://www.nlp.ecei.tohoku.ac.jp/projects/jaqket/#Reference

[3] Strictly, JSQuAD is not for evaluating generative QA, but the span extraction-based setting. We use this data because there is no common evaluation data in Japanese for generative QA. Our models generate answers not extract spans, thus, we also conduct human evaluations.

```
Based on the given texts, please make a pair of answerable question and answer.
Please make the answer in Japanese polite language.
Please respond in the JSON format.

## example
texts:"texts to extract the pair of question and answer"
output:{"Question":"the question that can be answered from the texts", "Answer":"the answer to the question"}

## input
texts:{QA context}
output:
```

Figure 2: An example of zero-shot prompt to generate a pair of QA.

```
texts:"Resolving technical debt is difficult; we look at JAL's challenge...(omitted)...JAL's watchword is Go To Cloud...(omitted),

output:{"Question":"What watchwords does Japan Airlines stand for?", "Answer":"JAL's watchword is Go To Cloud."}
```

Figure 3: An translated sample of the "## example" part in one-shot prompt. Note that the original is in Japanese.

3.1 Source Contexts and Filtering

We generate N question-answer pairs from each context. N is set to one or three in our experiments. We compare three specific sources of contexts: 1) a random sample of 6,000 Japanese Wikipedia articles (`wiki`), 2) a random sample of 6,000 news articles (`news`), and 3) contexts in JSQuAD's training dataset (`JSQuAD`). To collect the news articles, we gathered the most accessed articles from a search engine [4] during the period from May 2022 to May 2023. We limit each context to the first 300 characters before generating QA pairs by the instruct-tuned model.

3.2 Prompts for Generating QA Pairs

We provide examples of zero-shot and one-shot prompts with the setting $N = 1$ in Figure 2 and

[4] The URL of the engine/dataset is hidden to preserve the anonymity of authors, and will be shown after acceptance

Figure 3, respectively. These prompts aim to generate QA pairs from a context. In the zero-shot prompt, we first present the task instructions, followed by an explanation of the structure oh how an input text is represented, and their desired output JSON structure as shown in the "## example" section. For the setting $N > 1$, we modify the example of the JSON structure to include more QA pairs. Then, we write an input text in the "## input" section. In the zero-shot prompt setting, we only write the format of input and output structures, without including actual texts or the expected question-answer pairs corresponding to the context. On the other hand, in the one-shot prompt, we replace the "## example" section in 2 with the prompt shown in Figure 3. Unlike the zero-shot prompt, the one-shot prompt includes actual example contexts and their corresponding expected QA pairs. To better understand the effects of prompt engineering, we compare these two prompts in our experiments. The tuples of a context and generated QA pairs are used to fine-tune a GPT by the prompt shown in Figure 4.

4 Experiments

Evaluation Dataset and Compared Models: We use the JSQuAD (Kurihara et al., 2022) for evaluation. This evaluation data contains 4,470 human-authored QA pairs given Wikipedia articles as contexts. We use whole evaluation data for the automatic evaluation while randomly sampled 500 instances are used for manual evaluation.

We conduct a comprehensive comparison by exploring various combinations of contexts, the number of generated QA pairs denoted as N and prompts. Regarding contexts, we consider three options: `wiki`, `news`, `JSQuAD`, and, as detailed in Sec. 3.1. For N, we compare $N = 1$ and $N = 3$. We compare zero-shot and one-shot prompts [5].

Our proposed models are compared with two models: 1) a plain GPT model without fine-tuning and 2) a model fine-tuned on QA pairs from the JSQuAD training dataset (`Human`), where these QA pairs are human-authored while our proposed QA pairs are not human-authored.

Fine-tuning We use the synthesized QA pairs to fine-tune the Japanese version of GPT-NeoX (Black et al., 2022)[6]. To achieve improved speed, we employ LoRA fine-tuning (Hu et al., 2022). In generating answers, we use a prompt in the zero-shot setting (Figure 4).

```
## Instruction
{QUESTION}

## Context
{CONTEXT}

## Response
```

Figure 4: The prompt to generate answers with the fine-tuned GPT-NeoX.

Batch Size: {4, 8},
Learning Rate: {0.00001, 0.00005, 0.000001},
Epoch: {3, 4, 5,}, r: {4, 8, 16, 64, 128}, α: {1, 4, 16}

Table 1: The search range values in LoRA fine-tuning.

Metrics: For automatic evaluation, we employ BERTScore (Zhang et al., 2020) and BLEU (Papineni et al., 2002). BERTScore is implemented on our own with a Japanese BERT model.[7] As for BLEU, SacreBLEU library (Post, 2018) is used.

These automatic metrics may not directly capture the correctness of an answer to a given question. To address this, we also conduct manual evaluations by human judges. We ask four judges, who are experts in natural language processing or linguistics, to assess whether the generated answer is correct or not. We showed tuples of questions, answers, and contexts to the judges. We report the accuracy obtained from the manual evaluation.

Parameters We conducted a grid search for tuning parameters: batch size, learning rate, the number of epochs, as well as LoRA's hyperparameters (specifically α and r). The range of values explored during this search is provided in Table 1. Subsequently, the model that attained the highest BERTScore was chosen for evaluation.

5 Results

In this section, we present the results on JSQuAD.

5.1 Automatic Evaluation

Our primary interest lies in examining the impact of each strategy for synthesizing QA pairs on the

[5] We are constrained to one-shot due to the input length limit of ChatGPT.

[6] https://huggingface.co/cyberagent/open-calm-7b

[7] https://huggingface.co/cl-tohoku/bert-base-japanese-v3

performance of the downstream question answering task. Specifically, we focus on comparisons involving different contexts, prompts, and the quantities of automatically generated QA pairs.

Table 2 presents the scores of BERTScore and BLEU obtained by varying the contexts while keeping other settings, i.e., N and prompts are fixed. The table is divided into five sections. Starting from the top, the first section displays scores for QA models trained on human-authored QA pairs (Human) from the JSQuAD training dataset, along with the plain GPT model (GPT) without fine-tuning. The second and third sections showcase scores obtained when N is fixed to one, but we vary the prompts to zero-shot and one-shot. The fourth and fifth sections represent scores when we use $N = 3$.

Impact of Context on Performance: We observe that using contexts extracted from the news dataset yields relatively low scores, e.g., 0.713 and 0.747 in terms of BERTScore for zero-shot and one-shot settings with $N = 3$, respectively. The wiki context performs better (0.706 and 0.838) than news (0.713 and 0.747) for the same settings. Notably, the JSQuAD context achieves the highest BERTScore of 0.863 and 0.889 with $N = 1$ and $N = 3$, respectively. The results suggest that using Wikipedia as context provides an advantage, likely because the JSQuAD evaluation data is also derived from Wikipedia.

Impact of Prompts on Performance: The one-shot prompt is more effective. As shown in Table 2, the model fine-tuned on the zero-shot QA pairs ($N = 1$) generated from the contexts in JSQuAD training dataset achieves a BERTScore of 0.724. However, the one-shot prompts with $N = 1$ exhibit a significant performance gain, reaching a BERTScore of 0.863.

Effect of the Number of Generated QA Pairs on Performance: As we increase the number of QA pairs for context, there is a gain of 2.6 points in BERTScore (from 0.863 to 0.889). Remarkably, the achieved BERTScore of 0.889 is comparable to that of a model trained on human-authored QA pairs (0.899), despite our approach not utilizing any human-authored QA pairs.

5.2 Evaluation by Human Judges:

We present the results of the manual evaluation. Table 3 shows the comparisons between three outputs: answers generated by 1) our best performing model (JSQuAD ($N = 3$), and one-shot prompt) and 2) a model that is fine-tuned on human-authored QA pairs from the JSQuAD training dataset, and 3) gold answers in JSQuAD evaluation dataset. Remarkably, despite our approach does not use any human-authored QA pairs, the achieved accuracy is 45.4% while the model fine-tuned on human-authored QA pairs achieves only 38.4% in terms of accuracy. Gilardi et al. (2023) mention that automatic annotation with an instructor-tuning model has higher quality than annotations by crowd-workers, and our results are consistent with their claim. Note that the performance of both fine-tuned models falls significantly behind the Gold standard (90.4%), indicating ample room for improvement.

context	N	prompt	BERTscore	BLEU
Human	-	-	0.899	5.64
GPT	-	-	0.601	0.00
news	1	zero	0.697	0.02
wiki	1	zero	0.713	0.03
JSQuAD	1	zero	0.724	1.55
news	1	one	0.738	0.11
wiki	1	one	0.775	0.09
JSQuAD	1	one	0.863	4.83
news	3	zero	0.713	0.38
wiki	3	zero	0.706	0.23
JSQuAD	3	zero	0.740	1.85
news	3	one	0.747	1.25
wiki	3	one	0.838	1.66
JSQuAD	3	one	**0.889**	**6.77**

Table 2: Performances on different contexts and numbers of generated QA pairs.

QA Pairs	Accuracy (%)
JSQuAD ($N = 3$, one-shot prompt)	**45.4**
Human	38.4
Gold	90.4

Table 3: Accuracy calculated as the number of correct question-context-answer tuples divided by the total 500 evaluation instances.

6 Conclusions

This paper proposed to use an instruction-tuned model for synthesizing QA pairs. Our experimental results demonstrate that the models trained on automatically generated QA pairs achieve comparable or even superior performance compared to the fine-tuned model trained on human-authored QA pairs. In future studies, we plan to explore the relationship between the diversity of automatically generated QA pairs and their impact on the performance of downstream QA tasks.

References

Sidney Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, Michael Pieler, Usvsn Sai Prashanth, Shivanshu Purohit, Laria Reynolds, Jonathan Tow, Ben Wang, and Samuel Weinbach. 2022. GPT-NeoX-20B: An open-source autoregressive language model. In *Proceedings of BigScience Episode #5 – Workshop on Challenges & Perspectives in Creating Large Language Models*, pages 95–136, virtual+Dublin. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

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.

Fabrizio Gilardi, Meysam Alizadeh, and Maël Kubli. 2023. Chatgpt outperforms crowd workers for text-annotation tasks. *Proceedings of the National Academy of Sciences*, 120(30):e2305016120.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Kentaro Kurihara, Daisuke Kawahara, and Tomohide Shibata. 2022. JGLUE: Japanese general language understanding evaluation. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 2957–2966, Marseille, France. European Language Resources Association.

Dong Bok Lee, Seanie Lee, Woo Tae Jeong, Donghwan Kim, and Sung Ju Hwang. 2020. Generating diverse and consistent QA pairs from contexts with information-maximizing hierarchical conditional VAEs. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 208–224, Online. Association for Computational Linguistics.

Richard Yuanzhe Pang, Alicia Parrish, Nitish Joshi, Nikita Nangia, Jason Phang, Angelica Chen, Vishakh Padmakumar, Johnny Ma, Jana Thompson, He He, and Samuel Bowman. 2022. QuALITY: Question answering with long input texts, yes! In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5336–5358, Seattle, United States. 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.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Raul Puri, Ryan Spring, Mohammad Shoeybi, Mostofa Patwary, and Bryan Catanzaro. 2020. Training question answering models from synthetic data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5811–5826, Online. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, 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*, 21(140):1–67.

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.

Mrinmaya Sachan and Eric Xing. 2018. Self-training for jointly learning to ask and answer questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 629–640, New Orleans, Louisiana. Association for Computational Linguistics.

Duyu Tang, Nan Duan, Zhao Yan, Zhirui Zhang, Yibo Sun, Shujie Liu, Yuanhua Lv, and Ming Zhou. 2018. Learning to collaborate for question answering and asking. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1564–1574, New Orleans, Louisiana. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.