Generative Reader Optimization in the RAG-system

Andrey P. Sokolov
Moscow State University
Moscow, Russia
andrey.sokolov@yadro.com

Pavel Zamelin
National Research University Higher School of Economics
Nizhniy Novgorog, Russia
p.zamelin@yadro.com

Yulia Kamelina
National Research Lobachevsky State University of Nizhny Novgorod
Nizhniy Novgorog, Russia
y.kamelina@yadro.com

Polina Plastova
National Research Lobachevsky State University of Nizhny Novgorod
Nizhniy Novgorog, Russia
p.plastova@yadro.com

2024 V International Conference on Neural Networks and Neurotechnologies (NeuroNT) | 979-8-3503-6373-9/24/$31.00 ©2024 IEEE | DOI: 10.1109/NEURONT62606.2024.10585446

***Abstract*— This paper proposes the simple methodology for optimizing a generative reader subsystem as part of a RAG QA-system. Described methodology can be used as a preliminary optimization that could help building the question-answering system of the reasonable quality in a short time period. The main principle of this methodology is the usage of the OpenAI 's GPT-4 generative model as a gold reference generative reader. Our methodology describes the procedure for generation of the synthetic dataset and defines main optimization metrics. Usage of synthetic datasets makes it possible to accurately control that all changes in the QA-system make its answers closer in average to the gold reference reader. Based on the developed methodology we performed the set of basic optimization experiments to quickly find the better configuration of the reader subsystem. These experiments made it possible to significantly improve the quality of the answers of the reader subsystem. As a result, we achieved relative improvement of the semantic similarity between the answers of our reader and the reference one based on GPT-4 for almost 50% by BLEURT scale and 26% by the SAS scale. Our methodology was verified on the QA-dataset in Russian language but it's also applicable to any other language without significant modifications.**

Keywords*— *retriever augmented generation, RAG, semantic similarity, QA-systems, LLM

I. Introduction

Question answering systems is a rapidly growing field in the computer science and natural language processing. The development of these systems presents multiple challenging problems. One of the main issues is open-domain question answering scenario (ODQA) [1]. In this scenario, the user asks a question in a natural language format, and the system must provide an accurate and factual response.

In ODQA task setting it's assumed that QA-system may have and access to some large knowledge base. Wikipedia [2] is often used as such knowledge base for evaluation of different ODQA systems.

There are various architectures for open-domain question answering (ODQA) systems [3]. However, two key components are typically identified: the retriever and the reader. The retriever subsystem is responsible for searching the relevant documents in the knowledge base in response to the user's question. On the other hand, the reader component must provide an accurate and factual answer to the question, using the relevant documents as support. There're architectures that lack one these components. For example, in the generator-only systems developers try to aggregate all the data from the knowledge base in a single generative reader model. Some examples of such generator-only systems are T5 [4], GPT-3 [5], GPT-4 [6] and other models.

One promising architecture for solving the ODQA problem is the Retrieval Augmented Generation (RAG) architecture [7]. This architecture defines the tight integration of the non-parametric retriever with the parametric generative reader model. As a result, the RAG-system with the «small» reader with just 400M parameters outperforms generator-only system with 11B parameters by about 10% of the exact match (EM) scale on the NQ dataset [9]. Soon after the RAG architecture appearance the Fusion-In-Decoder architecture (FiD) was published [8]. This architecture could be thought of as the variation of the RAG architecture. In the FiD the retriever and the reader subsystems integration became less tight. Moreover, each component of the system was trained independently on its own specific task. Such distancing of the components made advantage of the RAG-like FiD system over fully parametric generative models even larger. Thus, FiD with reader with just about 770M parameters outperformed GPT3 reader with 175B parameters on the NQ dataset by more than 20% of the EM scale. These excellent achievements brought very close attention of the AI community to the RAG architecture in the last years.

In this paper we describe simple methodology that is suitable for the first stage preliminary optimization of the generative reader subsystem as one of the key parts of the RAG system. First, we postulate the OpenAI's GPT-4 reader as the gold reference generative reader. Based on the well-known NQ dataset by means of the machine translation and the reference reader API we generate ruNQ_GPT4 dataset. This dataset is suitable both for the choice of the main optimization metrics and for optimization of the reader subsystem configuration. Then we select semantic similarity metrics that correlate with the decisions of the reference reader the most. After that we perform set of simple experiments varying key configuration parameters of the generative reader subsystem. Such as the base generative model, its parameters and prompt. As a result, we achieve significant improvement in average similarity of the answers of the optimized reader and the reference one.

979-8-3503-6373-9/24/$31.00 ©2024 IEEE

II. Optimization Methodology

A. Key principles

This methodology is aimed on the preliminary first stage optimization of the generative reader subsystem of the RAG QA-system. Thus, it doesn't utilize fine-tuning of the generative reader model. Only the choice of the base model and its key configuration parameters is considered. Evaluation of the quality of the reader's answers is performed on evaluation dataset, that consists of the set of question-answer examples. Each example contains a question, relevant document and the answer given by the reference «gold» reader. As a reference reader we consider the OpenAI's GPT-4 model with some fixed question-answering prompt and parameters.

Our target is to build question answering system that would support Russian language. Because there's a lack of dedicated QA-datasets in Russian we generate specific QA dataset based on the well-known NQ dataset. Questions and documents of the NQ dataset are translated in Russian by means of the machine translation. For translation we use Yandex Translate API [10]. It demonstrated high quality of translation from English to Russian in our experiments. Instead of using answers from the original human labeled NQ dataset we generate paired (double) reference answers by the GPT-4 reader.

As the measure for semantic similarity between readers answers we consider several metrics: lexical and semantic ones. To choose the main optimization metric we again utilize GPT-4 model as the source of the «gold» reference information about the semantic similarity of the pairs of reference answers. The metric that correlates with GPT-4's judgements the most is chosen as the main one.

Finally, after generation of the evaluation dataset and choice of the main optimization metric we perform set of experiments varying configuration of the reader subsystem to make it's answers more similar to the reference reader's answers.

B. ruNQ_GPT4 Dataset

Dataset ruNQ_GPT4 is a question answering dataset with quadruple labelling. For each question in the dataset there're also two answers given by the two reference labelers and two semantic similarity estimations between two answers. Semantic similarity estimations are provided by another two reference labelers. As a reference labeler in both cases, we use GPT-4 model. For answer generation and estimation of the semantic similarity two different prompts are used. To generate double labels (both for the answers and for semantic similarity estimations) two consecutive calls of the API with the same prompt and parameters are used.

Each record in the dataset contains following fields:

- question (q) on the natural language;
- document (d) that contains answer for the question q;
- answer of the labeler #1 (a_1);
- answer of the labeler #2 (a_2);
- semantic similarity of answers a_1 and a_2 judged by the labeler #3 (s_1);
- semantic similarity of answers a_1 and a_2 judged by the labeler #4 (s_2).

Semantic similarity estimations are given in the following scale: «0» - answers have totally different meaning (semantic); «1» - answers have similar meaning but there're important differences; «2» - answers have identical meaning.

There're some important details of the dataset generation procedure that should be mentioned. Questions and documents were translated from the NQ dataset by means of the Yandex Translate API. Answer labels a_1 and a_2 were generated with GPT-4 API. For both answers same prompt was used. Semantic similarity labels s_1 and s_2 were generated by means of the GPT-4 API. Same prompt was use for generation of each of the labels.

As a result, we were able to quickly generate dataset ruNQ_GPT4 that contains 1K records of the defined structure.

C. Metrics Analysis

Various metrics are used to measure similarity between generated and the reference answers. Most common are lexical and semantic similarity metrics. Lexical metrics are based on the number of unigrams (words) that present in both of the compared answers (sentences). Semantic similarity metrics target to estimate how similar are two compared answers in terms of their semantic meaning. We've investigated following lexical similarity metrics: F1-score [11] and METEOR [12]. And following semantic similarity metrics: BERTScore [13], BLEURT [14] and SAS [15].

Table I TABLE I. shows the Pearson correlation coefficients between similarity metric values and similarity judgements of the reference labelers - s_1 and s_2 on the ruNQ_GPT4 dataset.

TABLE I. Correlation of Semantic similarity metrics with GPT4 judgements

Metric	s_1	s_2
SAS	0.816	0.817
BLEURT	0.785	0.791
BERTScore (F1)	0.769	0.772
F1	0.767	0.773
BERTScore (precision)	0.736	0.743
BERTScore (recall)	0.713	0.711
METEOR	0.713	0.715
s_1	1	0.972
s_2	0.972	1

It can be seen that popular lexical and semantic similarity metrics are in a good agreement with GPT4 estimates. Lexical metrics perform worse than modern semantic metrics. Double judgements of semantic similarity given by the reference GPT4 labeler are highly consistent with each other (note very high correlation between s_1 and s_2). The maximum Pearson correlation is achieved for the SAS and BLEURT metrics. In this regard, we will further consider these metrics to be the main ones for further optimizations of the reader subsystem.

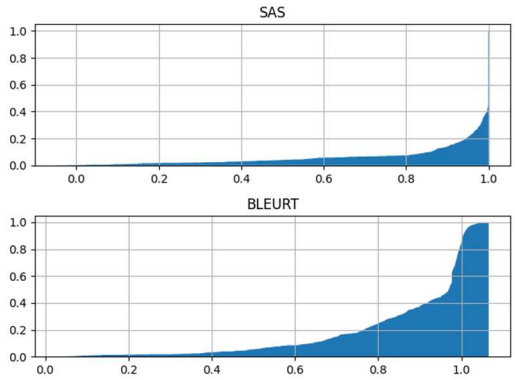

Fig. 1. Distribution of the semantic similarity between GPT4's paired answers

Following question arises: how similar are the answers of GPT4 reader to itself. Because ruNQ_GPT4 dataset contains double answers we can measure it. Histograms on Fig. 1 show distribution of semantic similarity between double answers a_1 and a_2 of the GPT4 model.

We see that despite the fact that the answers were given by the same generative model, there are examples of pairs of answers that are significantly different from the semantical point of view. Therefore, to assess the similarity of two readers, it makes sense to rely on the mean values of the SAS and BLEURT similarity metrics.

III. Optimization Experiments

In this section we analyze results of the experiments that were performed in order to find better configuration of the generative reader subsystem.

A. How prompt affects readers answers quality?

Define abbreviations and acronyms the first time they are used in the text, even after they have been defined in the abstract. Abbreviations such as IEEE, SI, MKS, CGS, sc, dc, and rms do not have to be defined. Do not use abbreviations in the title or heads unless they are unavoidable.

First, we investigate how prompt affects answers of the reader. The evaluation was carried out on ruNQ_GPT4 dataset.

The reader was queried with same question and relevant document but with various prompts. For each run, semantic similarity metrics were calculated between the generated answer and the reference one. Here we used a_1 answer as the reference. Table II contains results of multiple experiments with different prompting techniques.

TABLE II. Efficiency of the prompt engineering techniques

Prompt engineering technique	SAS (mean)	BLEURT (mean)
GPT Prompt Enhancer + Few-Shot Prompting	**0.695**	**0.534**
GPT Prompt Enhancer	0.689	0.527
Few-Shot prompting	0.677	0.514
Simplified Chain-of-thoughts	0.592	0.407
Emotional prompting	0.582	0.398
Advanced Bribe	0.555	0.369
Simplified Tree of thoughts	0.543	0.348
CapsLock Emphasis	0.515	0.316
Simple Bribe	0.499	0.306

It's clear that the prompt engineering is a powerful tool that can significantly improve quality of the answers of the generative reader. Best result was shown by a prompt compiled using GPT prompt enhancer combined with the few-shot prompting technique (adding examples of how the model should respond to the prompt).

B. How model parameters affect answer quality?

To estimate how model inference parameters (temperature, top_k and top_p) [16] affect quality of the generated answers we conducted series of experiments on the ruNQ_GPT4 dataset for the following grid of these parameters: top_p=[0.1, 0.45, 0.8], top_k=[10, 45, 80], temperature=[0.2, 0.45, 0.7].

Experiments showed that variation of these parameters has a minor effect on the average semantic similarity of the reader's answers to the reference ones. Mean SAS and BLEURT metrics varied less than 2%.

C. How base generator model affects quality of the answers?

Next, we've analyzed how variation of the base LLM model affects the semantic similarity of the answers to the reference ones.

To make comparison fair we analyzed only 7B class models. We investigated several publicly available models that were fine-tuned on the various LLM tasks and show high results on Russian SuperGLUE benchmark [17].

List of the compared models and their performance in terms of semantic similarity to the reference answers is presented in the Table III.

TABLE III. How base model affects quality of answers?

Model (on Hugging Face)	SAS (mean)	BLEURT (mean)
IlyaGusev/saiga2_7b_gguf	**0.817**	0.303
TheBloke/saiga_mistral_7b-GPTQ	0.790	0.385
IlyaGusev/saiga_mistral_7b_lora	0.779	0.359
IlyaGusev/saiga_mistral_7b_gguf	0.778	0.356
TheBloke/Llama-2-7B-GGUF	0.748	**0.397**
TheBloke/Llama-2-7B-GGUF	0.748	0.396
TheBloke/Llama-2-7B-GPTQ	0.728	0.363

It's clear that the base generator model significantly affects quality of the answers. Thus, fine-tuning of the base model for generative reader is an important way of optimization of the generative reader subsystem.

IV. Discussion

In the previous sections we've used GPT4 answers as the «gold» reference. Prompt engineering and base model variation allowed us to significantly improve quality of the answers of the generative reader subsystem without fine-tuning. More precisely it allowed us to make answers of the QA-system much closer to the answers of the GPT4-based reader.

But how did the similarity of generative reader's answers change relative to the answers of the human labelers that are available in original NQ dataset.

TABLE IV. CHANGES OF THE MEAN SEMANTIC SIMILARITY TO REFERENCE AND HUMAN ANSWERS

Dataset	Mean BLEURT relative change, %	Mean SAS relative change, %
ruNQ_GPT4	49.6	26.6
ruNQ (human labels)	-28.7	-24.1

Table IV TABLE IV. shows how much have changed the mean BLEURT and SAS metrics of the original version of generative reader subsystem and the optimized one on two datasets: ruNQ_GPT that contains answers of the GPT4 labeler and ruNQ dataset that contains human labeled answers that were translated from the original NQ dataset. Changes are presented in the relative scale compared to the BLEURT and SAS values before the optimization. We see that after the optimization of the reader subsystem configuration its answers became much close to the GPT4's answers but significantly less similar to the answers of the human labelers. It's not clear whose answers are better: provided by the human or by the robot. It is clear that the result of reader optimization entirely depends on whose answers will be taken as a reference.

V. CONCLUSION

We presented a simple methodology for quick preliminary optimization of the generative reader as part of a RAG question answering system. It has been shown that such simple methods as variation of the base model and prompt engineering can quickly raise the quality of system answers to the acceptable level. Further improvements of the answer's quality can be achieved by the fine-tuning of the base model.

REFERENCES

[1] Ellen M. Voorhees and Dawn M. Tice. 2000. The TREC-8 question answering track. In Proceedings of the Second International Conference on Language Resources and Evaluation (LREC' 00), Athens, Greece. European Language Resources Association (ELRA).

[2] Wikipedia. 2004. Wikipedia. PediaPress.

[3] Qin Zhang, Shangsi Chen, Dongkuan Xu, Qingqing Cao, Xiaojun Chen, Trevor Cohn, and Meng Fang. 2023. A Survey for Efficient Open Domain Question Answering. In Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 14447–14465, Toronto, Canada. Association for Computational Linguistics.

[4] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. J. Mach. Learn. Res., 21(140):1-67.

[5] 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.

[6] Benj Edwards. "OpenAI's GPT-4 exhibits "human-level performance" on professional benchmarks". Ars Technica. Archived from the original on March 14, 2023. Retrieved March 15, 2023.

[7] Patrick Lewis, Yuxiang Wu, Linqing Liu, Pasquale Minervini, Heinrich Kuttler, Aleksandra Piktus, Pontus Stenetorp, and Sebastian Riedel. 2021. PAQ: 65 Million Probably-Asked Questions and What You Can Do With Them. Transactions of the Association for Computational Linguistics, 9:1098-1115.

[8] Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, pages 874-880, Online. Association for Computational Linguistics.

[9] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. Transactions of the Association for Computational Linguistics, 7:453-466.

[10] Yandex Translate API - https://yandex.com/dev/translate/

[11] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1870-1879, Vancouver, Canada. Association for Computational Linguistics.

[12] Banerjee, S. and Lavie, A. "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

[13] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, Yoav Artzi, 2020. BERTScore: Evaluating Text Generation with BERT. International Conference on Learning Representations.

[14] Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning Robust Metrics for Text Generation. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 7881–7892, Online. Association for Computational Linguistics.

[15] Julian Risch, Timo Möller, Julian Gutsch, and Malte Pietsch. 2021. Semantic Answer Similarity for Evaluating Question Answering Models. In Proceedings of the 3rd Workshop on Machine Reading for Question Answering, pages 149–157, Punta Cana, Dominican Republic. Association for Computational Linguistics.

[16] Transformers Library, Text generation parameters, https://huggingface.co/docs/transformers/en/main_classes/text_generation

[17] Russian SuperGLUE - https://russiansuperglue.com/

[18] Hugging Face - https://huggingface.co/