Optimizing Language Models for Education: Integrating Fine-Tuning, DPO, RAG, and Quantization for Scientific Question Answering

Luca Engel | 329977 | luca.engel@epfl.ch
Damian Kopp | 324944 | damian.kopp@epfl.ch
Nino Gerber | 351542 | nino.gerber@epfl.ch
Anne-Marie Rusu | 296098 | anne-marie.rusu@epfl.ch
start-unk-stop-pad

Abstract

This project aims to develop a generative language model to be used as an AI tutor specializing in course content given at EPFL. We fine-tune a GPT-2 model to improve its performance in answering educational questions, specifically, multiple choice question answering (MCQA) and integrate advanced techniques such as Direct Preference Optimization (DPO), Retrieval-Augmented Generation (RAG), and post-training quantization (GPTQ).

Our contributions include demonstrating effective fine-tuning on specialized datasets, enhancing performance with RAG for additional context and answer complexity, using DPO to improve answer quality based on user preferences, and applying GPTQ to reduce model size for efficient deployment. Our experiments show improvement in MCQA accuracy, with the highest accuracy achieved by the fine-tuned model augmented with RAG. The quantized model also maintains performance while substantially reducing size, making it suitable for real-world educational applications. These findings highlight the potential of optimized language models to support students by providing accurate and contextually relevant answers.

1 Introduction

In nowadays' fast-moving learning environments, it is more and more important to include advanced technologies in the learning and teaching process.

The main aim of this project is the development of a generative language model tailored specifically for EPFL's educational context, called an "AI tutor". This model will enable students to engage better with their study materials by providing them with comprehensive, concise, and relatable answers to their questions. The results are meant to provide reliable support for students seeking further understanding in their courses at EPFL.

This project was motivated by the difficulty students face when trying to get immediate personalized help outside class. This may not always be possible through traditional methods such as tutoring or office hours since they may not address the diversity of student needs adequately.

Therefore, we have applied different enhancements on OpenAI's GPT-2 model (openai community, 2024) to improve its capability of such question answering. This includes fine-tuning for both multiple choice and open-ended question answering, Direct Preference Optimisation, and Retrieval Augmented Generation. Lastly, we also analyze the impact of Quantization on our model's size and performance.

2 Related Work

This work's research context is based on the diverse literature already existing in the field of modern natural language processing (mNLP). Many previous works leverage GPT-2 for question answering purposes, such as (Klein and Nabi, 2019), however, these often use datasets such as the Stanford Question Answering Dataset (SQuAD) which contain a wide range of topics, SQuAD being based particularly off of Wikipedia articles. Since this paper is tailoring specifically to EPFL course material, we consider only relevant datasets such as actual EPFL exam questions and answers and other STEM related question and answer datasets. Since our model needs to specialize in MCQA, we also need to apply different methodologies and processing to adapt to this, further explained in section 3.

Direct Preference Optimization: One foundational paper that aligns with the aim of this project is (Rafailov et al., 2023) which elaborates how this method can improve model performance, in particular, improving the relevance and quality of the generated answers. By giving the model hints of what "good" answers should look like in the form of data samples, the model should generate more outputs that have the preferred format and amount of detail.

Quantization: This augmentation is particularly useful to reduce the size of very large models. There are two methods of quantization: quantization-aware training and post-training quantization (Gholami et al., 2021). The former involves quantizing models during retraining or fine-tuning using approximate differentiation for rounding (Gholami et al., 2021; Nagel et al., 2021). The latter ("one-shot"), quantizes pre-trained models using limited samples and hours of computation. Because of its effectiveness and efficiency, we focus on layer-based post-training methods, in particular GPTQ (Frantar et al., 2023a) which is based on Optimal Brain Quantization (Frantar et al., 2023b).

Retrieval Augmented Generation: (Lewis et al., 2021) states that, when using a pre-trained neural retriever, their model outperforms parametric seq2seq models on various tasks, including open domain QA tasks and language generation. However, they do not focus on multiple choice questions as we do. Additionally, they use a seq2seq model while our research focuses on a causal language model. Since RAG has shown to be promising, particularly in the STEM domain with code generation (Zhou et al., 2023), we decide to adapt it and incorporate it in our model.

3 Approach

3.1 Fine-Tuning

(HuggingFace, 2024a) serves as a starting point to fine-tune the pre-trained model. In our case, GPT-2 was chosen to enable fine-tuning a more capable model. The fine-tuning process is done in two phases, each using a different approach.

In the first phase, the SciQ dataset is used in conjunction with the ELI5_Category dataset. Since SciQ is not of the same format, it needed reformatting to conform to the ELI5_Category dataset. The two datasets and their formatting are further described in section 4.1.

In the second phase, the focus lies on improving the model for multiple choice question answering. This led to the choice of solely involving the SciQ dataset containing multiple choice data points. The main goal here is to prepare the model to handle a specific format of questions and return the output in a unified way.

3.2 DPO

The dataset for DPO was collected in a collaborative effort of the students of the CS-552 class. The starting point was questions from EPFL courses. Every student then used a provided API to interact with ChatGPT to generate two answers for the question and, then, form a preference pair. This could be done by using two slightly different prompts for the same question. Lastly, every preference pair was ranked and the data samples were aggregated to form a dataset. Every preference pair was annotated with an overall score by the students with the purpose of indicating which of the two answers was better. Any sample pair that did not consist of one answer being preferred over the other was discarded to ensure a clear preference during the learning process. The resulting dataset contained a total of over 26'000 preference pairs.

For the implementation of DPO in our model, Huggingface's DPO Trainer is used (HuggingFace, 2024b).

3.3 RAG

This model enhancement is done by leveraging an existing RAG-Token Model from the Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks paper (Lewis et al., 2021). Specifically, the retriever is used to fetch relevant documents based on the model input. Then, using these documents prepended to the original question to provide the model with a larger context, our model generates a response.

3.4 Quantization

For quantization we use the GPTQ (Frantar et al., 2023a) approach. This method falls into the post-training quantization category. The GPTQ algorithm quantizes weights of neural network layers in a more computationally efficient manner compared to other methods such as Optimal Brain Quantization (OBQ). This is achieved by leveraging arbitrary order quantization, lazy batch-updates, and Cholesky reformulation.

Step 1: Arbitrary Order Insight

OBQ quantizes weights in a greedy order to minimize additional quantization error. However, GPTQ quantizes weights in an arbitrary order which performs similarly, especially on large models. This simplifies the process since the Hessian inverse H_F^{-1} is only updated once per column rather than for each weight, significantly reducing the runtime.

Step 2: Lazy Batch-Updates

To improve GPU utilization, updates are batched together. The algorithm processes $B = 128$ columns at a time, performing updates contained within those columns and their corresponding $B \times B$ block of H^{-1}. After processing a block, global updates to the entire H^{-1} and W matrices are performed using:

$$\delta_F = -\left(w_Q - \text{quant}(w_Q)\right)\left([H_F^{-1}]_{QQ}\right)^{-1}\left(H_F^{-1}\right)_{:,Q},$$

$$H_{-Q}^{-1} = \left(H^{-1} - H_{:,Q}^{-1}\left([H^{-1}]_{QQ}\right)^{-1} H_{Q,:}^{-1}\right)_{-Q}.$$

Step 3: Cholesky Reformulation

To address numerical inaccuracies, a Cholesky decomposition is used to compute the necessary information from H^{-1} upfront. This method is robust enough for large models and offers additional speedup.

Algorithm Pseudocode

The full GPTQ algorithm is summarized in the pseudocode below, taken from (Frantar et al., 2023a):

Algorithm 1 Quantize W given inverse Hessian $H^{-1} = (2XX^\top + \lambda I)^{-1}$ and blocksize B.

$Q \leftarrow 0_{d_{\text{row}} \times d_{\text{col}}}$ {Quantized output}
$E \leftarrow 0_{d_{\text{row}} \times B}$ {Block quantization errors}
$H^{-1} \leftarrow \text{Cholesky}(H^{-1})^\top$ {Hessian inverse}
for $i = 0, B, 2B, \ldots$ **do**
  for $j = i, \ldots, i + B - 1$ **do**
    $Q_{:,j} \leftarrow \text{quant}(W_{:,j})$ {Quantize column}
    $E_{:,j-i} \leftarrow (W_{:,j} - Q_{:,j}) / [H^{-1}]_{jj}$ {Q. error}
    $W_{:,j:(i+B)} \leftarrow W_{:,j:(i+B)} - E_{:,j-i} \cdot H^{-1}_{j,j:(i+B)}$
    {Update weights in block}
  end for
  $W_{:,(i+B):} \leftarrow W_{:,(i+B):} - E \cdot H^{-1}_{i:(i+B),(i+B):}$
  {Update all remaining weights}
end for

4 Experiments

4.1 Data

To fine-tune the pre-trained GPT-2 model, two datasets are used: ELI5_Category for open-question data points (Gao et al., 2021), and SciQ for multiple choice questions (Johannes Welbl, 2017).

To maximize the benefits of the fine-tuning process throughout numerous tasks, different combinations of the two datasets are used. During the fine-tuning for DPO, both ELI5_Category and SciQ are used. During the fine-tuning for MCQA, RAG, and Quantization, only the SciQ dataset is used because these tasks are to be ultimately evaluated on a multiple choice dataset.

4.1.1 ELI5_Category Dataset

ELI5_Category contains a diverse set of scientific questions with open-ended answers taken from the subreddit *r/explainlikeimfive*. The goal of this dataset is to align our model to scientific questions similar to the open-ended question answering expected in the DPO dataset. As this data was taken from a public platform where anyone can submit an answer, there are some concerns such as biases, accuracy and quality of answers. However, this is mitigated on the platform itself through use of moderators and strict rules, as well as additional filtering during construction of the dataset.

In detail, each sample contains multiple attributes, table 1 shows the particular ones that we use for fine-tuning our model.

Feature	Description
title	`string` - Question title.
answers	`dict` - Answers.
text	`list[string]` - Answer texts.
score	`list[int]` - Answer scores, higher scores representing better answers.

Table 1: Features of Preprocessed ELI5 Open Questions and Detailed Answers for Long-form QA

The split of this dataset is shown in table 2.

Split	Entries
Train	91772
Validation 1	5446
Validation 2	2375
Test	5411

Table 2: ELI5 Dataset Split Information

4.1.2 SciQ Dataset

The SciQ dataset serves the purpose of improving the model's ability to handle multiple choice questions. The data was crowdsourced, through guided

question and answer creation by crowd workers to ensure quality question-answer pairs.

The split of this dataset is shown below:

Split	Nb Entries
Train	11679
Validation	1000
Test	1000

Table 3: Dataset Split Information

The SciQ dataset is reformatted to be compatible with ELI5_Category by assigning to the "title" feature, the concatenation of the question and the answer options. The correct answer is treated the same way as a high scoring answer in the ELI5_Category dataset, by attributing a large score (10) to it while the distractors receive a comparatively small one (1).

Feature	Description
question	`string` - Science question.
correct_answer	`string` - Correct answer.
distractor1	`string` - First distractor.
distractor2	`string` - Second distractor.
distractor3	`string` - Third distractor.
support	`string` - Question context.

Table 4: Features of SciQ Multiple Choice Science Questions

Additionally, to ensure that the multiple choice question format aligned with the one used for the evaluation, every sample in the SciQ dataset is reformatted. This ensures that MCQA inputs are more easily recognizable for our model.

4.1.3 Preference Data

Huggingface's DPOTrainer (HuggingFace, 2024b) is used for DPO. This trainer expects data in preference pairs, described in table 5, therefore we preprocess the preference data samples to conform to this.

Feature	Description
prompt	`string` - Science question.
chosen	`string` - Preferred answer.
rejected	`string` - Rejected answer.

Table 5: Features of Preference Data Pairs

4.2 Baselines

The original reference model "openai-community/gpt2" serves as our baseline model (openai community, 2024), which we augment in the fine-tuning, enhancement and alignment stages.

4.3 Experimental details

For fine-tuning the pre-trained GPT-2 model, the training configurations are displayed in table 6.

Parameter	Value
Learning Rate (LR)	3×10^{-5}
Train Batch Size	8
Evaluation Batch Size	8
Seed	42
Grad. Acc. Steps	2
Total Train Batch Size	16
Optimizer	Adam ($\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1e-8$)
LR Scheduler Type	Linear
Number of Epochs	10

Table 6: Training Configurations for Fine-Tuning GPT-2 (start-unk-stop pad, 2024b) (start-unk-stop pad, 2024c)

Similarly, the training configurations shown in table 7 are used for direct preference optimization.

Parameter	Value
Learning Rate (LR)	1×10^{-6}
Train Batch Size	8
Evaluation Batch Size	8
Seed	42
Grad. Acc. Steps	4
Total Train Batch Size	32
Optimizer	Adam ($\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1e-8$)
Weight Decay	0.01
LR Scheduler Type	Cosine
Number of Epochs	10

Table 7: Training Configurations for DPO (start-unk-stop pad, 2024a)

For the MCQA evaluation, the model predicts one answer which is then compared to the true answer. While our model fine-tuned on the SciQ dataset is trained to generate the answer letter, i.e., "A", "B", "C", or "D", as the first token, this is not always the case. Therefore, we apply a post-

processing step to extract the predicted letter. This step consists of calculating the BERTScore between the generated text and the four answer options. The option with the highest score is picked as the answer. This allows a consistent way of picking a single letter answer and, more importantly, allows comparing the fine-tuned model's performance with the baseline, DPO and RAG models which have not all been fine-tuned to answer questions in this manner.

4.4 Results

The different models with their corresponding performances on the test split of the SciQ dataset are listed in table 8. For Finetuned 1, the models are fine-tuned solely on the SciQ dataset while for Finetuned 2, the ELI5 dataset is also leveraged.

GPT-2 Model	MCQA
Baseline GPT-2	0.291
Baseline GPT-2 + RAG	0.319
Finetuned 1	0.227
Finetuned 1 + RAG	**0.334**
Finetuned 1 Quantized 8b	0.227
Finetuned 1 Quantized 8b + RAG	0.329
Finetuned 2 + DPO	0.319
Finetuned 2 + DPO + RAG	0.311

Table 8: Accuracy of the Models on the SciQ Test Dataset

While solely using the Finetuned 1 model for the MCQ answering yields the worst accuracy on the test set, the addition of RAG dramatically improves this score. In fact, this combination yields the highest accuracy overall, surpassing the baseline model and that same model extended with RAG.

In addition to the achieved accuracy scores, this paper analyses the impact RAG has on the token generation speed. The speed comparisons of the baseline, fine-tuned, and RAG augmented models are presented in table 9. The speed was measured for generations using the SciQ test dataset samples as prompts. To ensure a uniform comparison, each generation was run on Google Colab using a T4 GPU and, for every prompt, the model was set to generate 100 tokens. The result presented in the table is the time spent to generate a single token. For each model, 100 inputs were used to compute the mean and standard deviation.

While the Baseline and Finetuned 1 model's both generate at a similar speed, the inclusion of RAG incurs a slowdown of about 16%.

Model	mean	std
GPT-2 Baseline	12.41	1.65
Finetuned 1	12.68	1.77
Finetuned 1 + RAG	14.53	1.79

Table 9: Comparison of Time Spent per Single Token Generation [ms]

Lastly, the effect of quantization on the model's memory footprint is analyzed. Table 10 details the resulting model sizes when compressing the Finetuned 1 model with GPTQ. The resulting model is 35% as big as the original GPT-2 baseline.

Model	Size
GPT-2 Baseline 32b	510.34 MB
Quantization 8b	178.78 MB

Table 10: Quantized (start-unk-stop pad, 2024d) and Baseline Model Sizes

5 Analysis

5.1 MCQA Accuracies

The performances of the various GPT-2 based models on the SciQ test dataset highlight the impact of different fine-tuning strategies and extensions on multiple choice question answering accuracy. The Finetuned 1 model achieves an accuracy of 22.7%, reflecting the challenge of generating accurate responses without additional context. More specifically, as the fine-tuned model was trained to solely predict the correct answer without explanations, the benefits of structured reasoning to gain context and find the correct answer step-by-step, which is the case for the better performing GPT-2 baseline, cannot be leveraged here.

Introducing RAG to the Finetuned 1 model significantly boosts accuracy to 33.4%, thus surpassing the baseline, demonstrating the effectiveness of incorporating external information for context-driven predictions. The second fine-tuned model with DPO achieves a lower accuracy of 31.9% compared to the RAG-enhanced model. DPO enhances the model's ability to generate answers through a structured thought process. Thanks to the BERTScore comparisons to select the correct multiple choice response, this structured thought process

can be leveraged to increase response accuracy.

However, when RAG is combined with DPO, the accuracy drops marginally to 31.1%, suggesting that the benefits of additional context provided by RAG are somewhat diminished when the model already employs a sophisticated alignment mechanism like DPO. One reason for this could be that GPT-2 is a relatively small model and, therefore, might struggle to leverage multiple sophisticated mechanisms.

Overall, these results indicate that while RAG significantly enhances performance by providing context, DPO's structured reasoning approach also offers substantial benefits, although the combination of both methods does not lead to additive improvements. These results underscore the importance of choosing the right extension based on the specific strengths it brings to the task.

5.2 RAG Speed Evaluation

The results of the speed comparison shown in table 9 allow to conclude that, for the given setting, token generation using RAG is around 1/6 slower than without it. This slowdown is incurred due to the need to retrieve documents relevant to the multiple choice question. At the same time, including RAG on the Finetuned 1 model drastically increases its accuracy. This trade-off needs to be evaluated on a case-by-case basis depending on the needs of the target users. For instance, if speed is more important than the correctness of the answers, one could consider not using RAG. However, as our use case is question answering for students, accuracy is of high importance. Therefore, the additional slowdown for each prediction is worth taking.

5.3 Quantization

The quantized version of the Finetuned 1 model demonstrates a notable reduction in model size with only a slight impact on performance. The quantization process, applied using the GPTQ method on our custom dataset, reduces the model size from 510.34 MB to 178.78 MB, resulting in a 65% decrease in storage requirements. Despite this substantial reduction, the performance of the quantized model on the SciQ Test Dataset remains consistent with its non-quantized counterpart. Both models achieve an accuracy of 0.227. However, when augmented with RAG, the quantized model's performance is slightly worse, achieving an accuracy of 0.329 compared to the non-quantized, RAG augmented Finetuned 1 model's accuracy of 0.334. This marginal difference suggests that, while quantization effectively reduces model size while maintaining a similar performance, it may introduce performance limitations when combined with additional complex enhancements like RAG. This might also be due to the limited size of GPT-2, which becomes even more constrained after quantization is applied. This trade-off between model efficiency and the nuanced impact on performance is critical for applications where resource constraints and model deployment considerations are paramount.

6 Ethical considerations

If successful, this project allows for several benefits. Students, for example, now have access to a platform they can use to expand their knowledge at their own pace and at the time convenient to them. This means they have an opportunity to create a better understanding of the content covered in their courses, and eventually do better in exams. This can also help teachers and teaching assistants, especially in courses with a high number of students, as they will have less workload answering questions outside the designated hours. Eventually, if the use of the AI tutor is able to be monitored by the school or teacher, it can provide valuable insights into which topics students find challenging and the typical types of questions they have.

Despite these benefits, there are some ethical concerns to be considerate of. Firstly, if the model only returns final answers, which is the case for MCQA, students might exploit this to only obtain answers (especially for courses which do not offer solutions to past exams) and not do the extra work of understanding the explanation behind the answer or train their critical thinking skills. In addition, problems like cheating, can arise in exams or assignments, especially since the model was trained specifically on EPFL course problems. To mitigate this, teachers could enforce strict usage policies or eventually monitor the usage of the AI tutor. Furthermore, since the datasets were augmented with external datasets, potential bias can be a risk. One risk that RAG presents is the knowledge source used. As mentioned by (Lewis et al., 2021), RAG could be used to "generate abuse, faked or misleading content". They also state that the factual accuracy is never entirely guaranteed for external knowledge sources. However, given that the content on Wikipedia is created by an open community,

one can still expect a high level of factual accuracy. The same reasoning can be applied for potential biases.

It is also important to consider the accessibility of the AI tutor. Since EPFL is a multilingual school, other languages should be available to ensure everyone is able to take advantage of the tutor equally, especially if their first language isn't English. For high-resource languages, datasets can be concatenated (with potential reformatting of questions for consistency) to form one multilingual corpus. This can be used along with a multilingual tokenizer to allow the model to output responses in multiple languages at once, and then the output in the desired language can be returned to the user. After this augmented model is established, cross-lingual transfer learning can be leveraged to adapt the model to work for low-resource languages (Otten, 2023). However, the success of this depends on several factors such as language similarity and dataset size, meaning it may be possible that students using the tutor in a low-resource language may get worse results than a student using a high-resource language.

In addition to supporting multiple spoken languages, the model can be adapted to support signed languages. Signed language is a crucial form of communication within the deaf community of around 70 million people (Desai et al., 2023). It has been shown that children can develop literacy and educational issues if not exposed to communication early enough, which can often be the case for deaf children (Yin, 2024). This can be caused by lack of learning resources, but also lack of standardized signs, such as for technical terms which are sometimes signed, sometimes fingerspelled. Adapting a language model to signed language is not as straightforward as for spoken language. American Sign Language (ASL) can be deconstructed into four manual categories: hand shape, orientation, location and movement, and five facial expressions. However, in practice, conversation also takes spatial organization into play and there is still the issue of potentially inconsistent signs for a single concept. These can be extracted from a dataset such as the ASL Citizen dataset of signed videos (Desai et al., 2023) as "glosses", however these cannot be directly mapped to a corresponding English sentence and need to be translated (Moryossef et al., 2021).

The main challenge is to translate back into signed language, to have a relevant use for users (otherwise using spoken language would've been sufficient). (Jiang, 2022) describes a system to produce large datasets for continuous ASL which can be used to generate ASL, which can for example, then be mapped to computer generated models to produce the visual sign representations. However, this remains to be explored in depth. One main component to the success of this adaptation is collaboration with the deaf community, as they have the understanding of the nuances and complexity of the language (Desai et al., 2023) and are the ones who can provide meaningful insight to create an accurate model.

7 Conclusion

Our project demonstrates several key findings regarding the application of GPT-2 models and enhancements like fine-tuning, RAG, and quantization techniques for scientific question answering. Primarily, the baseline performance of GPT-2 models was suboptimal, particularly for MCQA tasks that demand precise format adherence and correct answer identification. This issue stems from the inherent challenge GPT-2 faces in consistently producing responses in the exact required format.

By incorporating RAG, we significantly improved model accuracy, as this method allowed the model to leverage additional context from retrieved documents, thereby enhancing the relevance and correctness of its responses. Despite the computational slowdown introduced by RAG, we deem the trade-off beneficial since our target users, students, prioritize accuracy over speed.

Furthermore, we explored model quantization using GPTQ, a state-of-the-art technique that substantially reduces the model size by 65% while maintaining comparable performance levels. This reduction in size, achieved through efficient post-training quantization, is crucial for deploying models in resource-constrained environments while minimally sacrificing accuracy.

Our study also highlights the limitations of using smaller models like GPT-2, which underperform compared to larger, more capable models. Future research should focus on utilizing larger models to overcome these limitations and further refine MCQA performance.

8 Contributions

Overall, all team members contributed fairly to the development of the model. Discussions were

done jointly as team, and each member contributed evenly to the writing of the report.

References

Aashaka Desai, Lauren Berger, Fyodor O Minakov, Vanessa Milan, Chinmay Singh, Kriston Pumphrey, Richard E Ladner, Hal Daumé III, Alex X Lu, Naomi Caselli, and Danielle Bragg. 2023. Asl citizen: A community-sourced dataset for advancing isolated sign language recognition.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. 2023a. Gptq: Accurate post-training quantization for generative pre-trained transformers.

Elias Frantar, Sidak Pal Singh, and Dan Alistarh. 2023b. Optimal brain compression: A framework for accurate post-training quantization and pruning.

Jingsong Gao, Qingren Zhou, and Rui Qiu. 2021. ELI5-Category: a categorized open-domain qa dataset.

Amir Gholami, Sehoon Kim, Zhen Dong, Zhewei Yao, Michael W. Mahoney, and Kurt Keutzer. 2021. A survey of quantization methods for efficient neural network inference.

HuggingFace. 2024a. Causal language modeling.

HuggingFace. 2024b. Dpo trainer.

Yehong Jiang. 2022. Sdw-asl: A dynamic system to generate large scale dataset for continuous american sign language.

Matt Gardner Johannes Welbl, Nelson F. Liu. 2017. Crowdsourcing multiple choice science questions.

Tassilo Klein and Moin Nabi. 2019. Learning to answer by learning to ask: Getting the best of gpt-2 and bert worlds.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2021. Retrieval-augmented generation for knowledge-intensive nlp tasks.

Amit Moryossef, Kayo Yin, Graham Neubig, and Yoav Goldberg. 2021. Data augmentation for sign language gloss translation.

Markus Nagel, Marios Fournarakis, Rana Ali Amjad, Yelysei Bondarenko, Mart van Baalen, and Tijmen Blankevoort. 2021. A white paper on neural network quantization.

openai community. 2024. openai-community/gpt2.

Neri Van Otten. 2023. How to implement cross-lingual transfer learning in 5 different ways.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model.

start-unk-stop pad. 2024a. mnlp-project/gpt2-dpo.

start-unk-stop pad. 2024b. mnlp-project/gpt2-finetuned.

start-unk-stop pad. 2024c. mnlp-project/gpt2-finetuned-mcqa-sciq2-safety.

start-unk-stop pad. 2024d. mnlp-project/gpt2-safety-gptq-8bit.

Kayo Yin. 2024. Natural language processing for signed languages.

Shuyan Zhou, Uri Alon, Frank F. Xu, Zhiruo Wang, Zhengbao Jiang, and Graham Neubig. 2023. Docprompting: Generating code by retrieving the docs.

A Appendix

A.1 AI Usage

ChatGPT was used occasionally in the writing of this report, to improve grammar and sentence structure to ensure a coherent writing style as multiple people were contributing at the same time. To ensure correctness, the suggestions were proofread and if needed, discussed among team members.