MH6812 Group Project Report

Answer Science Multiple-Choice Questions Using LLM

Group 9
Gu Yuqing: *ygu019@e.ntu.edu.sg*
Liu Jiayang: *LIUJ0149@e.ntu.edu.sg*
Pan Yiyi: *ypan014@e.ntu.edu.sg*
Wu Xiaoshi: *xwu035@e.ntu.edu.sg*

Abstract

Our study introduces a deep learning-based approach for tackling multiple-choice questions from a Kaggle competition focused on science exams. Utilizing the DeBERTa-v3-large transformer model, the methodology aims to deliver precise solutions to multiple-choice questions within science exams. The dataset employed is curated from the Kaggle competition, encompassing high-quality examples spanning diverse scientific disciplines. Extensive preprocessing, tokenization, and data collation procedures are tailored to the dataset's unique characteristics, while techniques like Low-Rank Adaptation (LoRA) and selective layer freezing are integrated to optimize training efficiency and mitigate overfitting. Performance evaluation using metrics like Mean Average Precision (MAP) at 3 showcases the model's ability to accurately predict correct answers. Analysis of trainable parameters offers insights into the model's complexity and computational requirements. Meanwhile, we implemented Retrieval-Augmented Generation (RAG) as the booster of the model performance, which contributes impressive improvement in the increment of MAP scores, while combining with the finetuning methods. Overall, the study demonstrates the effectiveness of the proposed approach in addressing the challenges of science exam questions, contributing to the advancement of automated assessment systems in science education and professional evaluation processes.

1 Introduction

In today's educational landscape, the demand for efficient and accurate assessment methodologies has led to the exploration of innovative solutions harnessing the power of artificial intelligence and natural language processing. Multiple-choice question answering systems represent a critical component of automated assessment frameworks, offering scalability and objectivity in evaluating learners' comprehension across various domains. Our study presents a comprehensive analysis of a deep learning-based approach designed to tackle multiple-choice questions sourced from a Kaggle competition focused on science exams. The dataset utilized in this study originates from a coding competition on Kaggle, where participants endeavor to answer multiple-choice questions generated by Large Language Model (LLM). The competition's test set comprises 200 sample questions with answers, providing a glimpse into the question format and the nature of the questions posed by the LLM, albeit without divulging the specifics of the question generation process. Each question in the dataset includes a prompt (the question itself), five options labeled A, B, C, D, and E, and the correct answer denoted as 'answer'. Leveraging state-of-the-art transformer models, particularly the DeBERTa-v3-large architecture, this study aims to provide a robust solution to the challenges posed by complex and diverse question types encountered in science assessments.

2 Background

With the continual advancement of large language models (LLMs), researchers are increasingly focused on understanding and characterizing these models. Traditional benchmarks in natural language processing (NLP) have proven insufficient for state-of-the-art models, prompting exploration into more challenging tasks tailored to push these models' limits. Techniques such as quantization and knowledge distillation are utilized to effectively compress language models, facilitating their operation on less powerful hardware setups. Transformer-based language models, pioneered by Vaswani et al. (2017), have propelled NLP research into a new era, leveraging self-attention structures and extensive pre-training on vast self-supervised datasets. Previous research has established a scaling law, suggesting that the efficacy of language models correlates with both model size and pre-training data magnitude. This principle has spurred a shift towards Large Language Models (LLMs), with parameters increasing from millions to billions.

Recent advancements, including supervised fine-tuning and alignment with human values, have enhanced LLM capabilities, enabling closer adherence to human instructions and ethical

considerations. However, challenges persist, particularly in Multiple Choice Question Answering (MCQA) scenarios, where LLMs may exhibit response inconsistencies. Robinson & Wingate (2022) coined the term "multiple-choice symbol binding" (MCSB), highlighting variations in this ability across models. Wang et al. (2023) identified vulnerabilities in candidate response ranking, susceptible to manipulation by altering presentation orders. Zheng et al. (2023) investigated token bias in LLMs, potentially introducing biases during option prediction. Building on these studies, our research conducts a comprehensive array of experiments on DeBERTa models, with the use of finetuning methods, aiming to assess their performance consistency across diverse settings. This endeavor seeks to provide a comprehensive understanding of LLM capabilities and limitations in scientific multiple-choice questions.

3 Approach

3.1 Overview

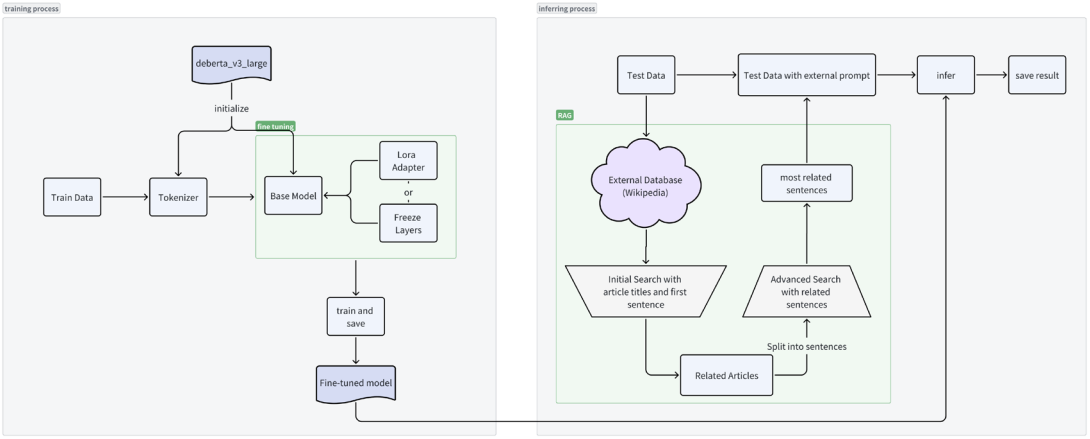

Figure 1: Our workflow (training & inferring)

Our workflow can be divided into two parts, training and inferring.
In the training process, deberta_v3_large was chosen as the base model, loaded into the AutoModelForMultipleChoice instance, which is a model with a multiple-choice classification head provided by Huggingface. Then, we leveraged a finetuning method to train the model on the science exam dataset.
In the inferring process, we fetched all articles from the 2023-07-01 dump of Wikipedia as the external knowledge library, and applied Retrieval-Augmented Generation to append additional "open-book" prompt for each science question in the test dataset. Finally, we used the finetuned model in the training process to predict.

3.2 Base Model

DeBERTa v3's capacity to deal with lengthy text sequences, a common problem in many NLP tasks, is one of its most significant advantages. The model can process up to 4096 tokens in a single pass, which is significantly more than popular models such as BERT and GPT-3. In our inferring process, prompts after RAG contain thousands of tokens, therefore DeBERTa v3 is particularly useful in this situation.
The DeBERTa V3 large model comes with 24 layers and a hidden size of 1024. It has 304M backbone parameters with a vocabulary containing 128K tokens which introduces 131M parameters in the Embedding layer. This model was trained using the 160GB data as DeBERTa V2.

3.3 Finetuning methods

We introduced two finetuning methods in our project and compared their advantages and disadvantages by quantitative analysis.

3.3.1 Layer Freezing

Layer freezing is a common approach used in LLM Finetuning. The idea behind layer freezing is to

selectively update or fine-tune only specific layers of the pre-trained model while keeping the parameters of other layers fixed. Layer freezing can avoid catastrophic forgetting, where new features overwrite old knowledge. By freezing certain layers, the model retains its ability to understand general language structure while adapting to the nuances of the specific task.

3.3.2 Low-Rank Adaption

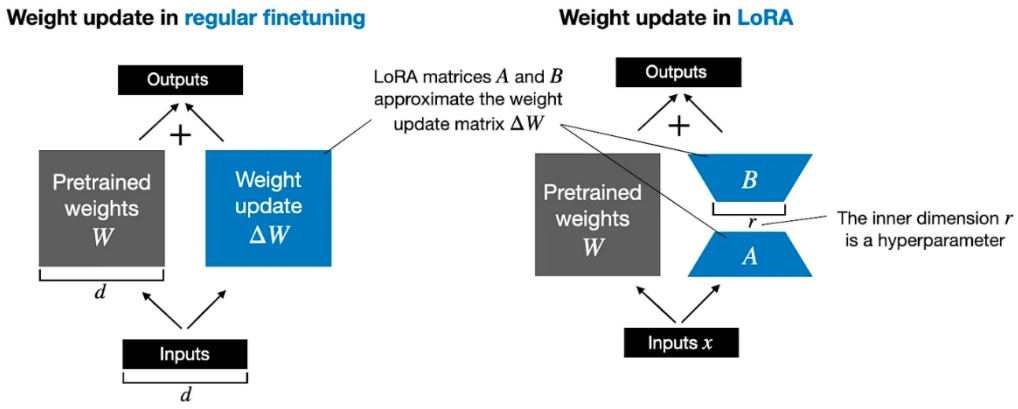

Figure 2: Comparison of weight update in regular finetuning and LoRA

Low-Rank Adaption is a finetuning method proposed by the Microsoft research team. Traditional finetuning methods usually update all the parameters of the pre-trained model, and as the sizes of large language models are getting larger and larger, the all-update methods have faced critical deployment challenges. Microsoft researchers hypothesize that the change in weights during model adaptation also has a low "intrinsic rank", leading to their proposed Low-Rank Adaptation (LoRA) approach. LoRA allows us to train some dense layers in a neural network indirectly by optimizing rank decomposition matrices of the dense layers' change during adaptation instead, while keeping the pre-trained weights frozen. As shown in figure 2, the decomposition of ΔW means that we represent the large matrix ΔW with two smaller LoRA matrices, A and B. If A has the same number of rows as ΔW and B has the same number of columns as ΔW, we can write the decomposition as ΔW = AB. In this way, we can save a lot of memory and weight update.

3.4 Retrieval-Augmented Generation

Retrieval-Augmented Generation is a concept proposed by the Facebook research team, aimed at enhancing the performance of language understanding and generation tasks. RAG combines elements of both generative and retrieval-based models. The retrieval component is key of RAG. It enables the model to access up-to-date information and overcome the limitations of purely generative approaches, which might rely solely on pre-existing knowledge.

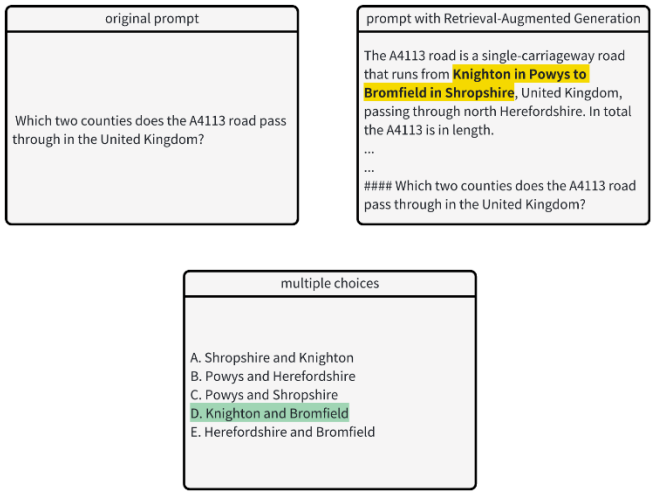

Figure 3: RAG provides relevant knowledge, strengthening inferential capability

In our model, we used the 2023-07-01 dump of Wikipedia as the external knowledge library. For every science question in the test dataset, we combined the question and the choices as a query, applied initial search with all article titles and the first sentence, to find the top five articles most similar to the query, then searched in these five articles, to find the top twenty sentences most similar to the query. Finally, we combined the original prompt and these sentences as the new prompt. When our model was not integrated with the RAG part, it could only infer the answer to science questions with the original prompts and implicit patterns learned before and in finetuning, which may be inaccurate, incomplete and out of date. But with the RAG function, the model could search relevant knowledge from Wikipedia Database in real time, which turned the science exam to open-book, and provided well-informed and trustworthy prompts.

4 Experiments

4.1 Data

4.1.1 Training Set

The training dataset comprises 15,000 high-quality examples obtained from Kaggle, specifically sourced from the "15k_gpt3.5-turbo.csv" dataset. These examples are meticulously curated to encompass diverse and intricate scenarios relevant to our task.

4.1.2 Validation Set

A subset extracted from the "kaggle-llm-science-exam/train.csv" dataset is employed for validation purposes. This validation dataset serves to fine-tune the model's performance on unseen data and validate its generalization capacity.

4.1.3 Test Data

The test dataset is created by combining the first 400 rows from two datasets: "15k-high-quality-examples/5900_examples.csv" and "kaggle-llm-science-exam/train.csv". This amalgamation facilitates a comprehensive evaluation across various types of questions.

4.1.4 Test_with_infer Data

The Test_with_infer dataset was generated through Retrieval-Augmented Generation, incorporating external information sourced from Wikipedia. The Accuracy infer values reported in Tables 1 and 3 represent performance evaluated using the test data.

4.2 Evaluation Method

Mean Average Precision at 3 (MAP@3) is an evaluation metric commonly used in information retrieval and recommendation systems. It measures the model's accuracy in ranking the correct answer within the top three predictions.
Formula for MAP@3:

$$\text{MAP@3} = \frac{1}{Q}\sum_{q=1}^{Q}\sum_{k=1}^{3} \text{p@k}(k) \times rel(q, k)$$

where:

- Q is the total number of questions.
- P@k represents the precision at k for question q, which is the fraction of the top 3 predicted answers that are correct for question q.
- rel(q, k) is an indicator function that takes a value of 1 if the correct answer for question q appears in the kth predicted position, and 0 otherwise.

A higher MAP@3 value indicates better model performance. It signifies that the model is effectively retrieving relevant answers and placing them within the top three positions of the ranked list.

4.3 Layers Freezing

4.3.1 Model Settings

1. Freeze Layer:
The number of frozen layers determines the number of parameters fine-tuned in the model. Higher

freeze layer values lead to fewer trainable parameters, faster training times, but potentially lower inference accuracy. In our experiment, we varied this parameter across values of 6, 12, 18, and 20 to assess its impact on model performance.

2. Learning Rate:

The learning rate regulates the size of parameter updates during the training process. Higher learning rates accelerate learning but may introduce instability or overfitting, whereas lower learning rates foster stability and enhance generalization. In our model settings, we experimented with learning rates of 2.00E-04, 1.00E-05, and 1.00E-06 to evaluate their effects on training dynamics and model performance.

3. Learning Rate Scheduler Type:

The learning rate scheduler controls how the learning rate changes throughout training. Different schedulers can lead to distinct convergence behaviors and optimization trajectories. In our experiments, we employed either a cosine or linear scheduler to explore their respective impacts on training dynamics and model performance.

4. Warmup Ratio:

The warmup ratio controls the initial rate of learning rate increase during training. Warmup helps stabilize training by preventing large initial updates. We incorporated a ratio of either 0 or 0.1 in the project.

4.3.2 Results

Freeze layers	Learning rate	lr Scheduler type	Warmup ratio	Train Time	GPU Memory	Model Size	Trainable parameters	Accuracy	Accuracy infer
DeBERTa v3 baseline model without training								0.26583	0.35194
6	2.00E-04	linear	0.00	0:28:40	14.7GB	1.74GB	228,308,993	0.39667	0.34139
6	1.00E-06	cosine	0.10	0:29:40	15.2GB	1.74GB	228,308,993	0.30833	0.45556
6	1.00E-05	cosine	0.10	0:29:43	15.1GB	1.74GB	228,308,993	0.73944	0.74833
12	2.00E-04	linear	0.00	0:26:00	15.1GB	1.74GB	152,731,649	0.54028	0.45472
12	1.00E-05	cosine	0.10	0:26:53	15.1GB	1.74GB	152,731,649	0.74611	0.57694
18	**2.00E-04**	**linear**	**0.00**	**0:24:19**	**15.1GB**	**1.74GB**	**77,154,305**	**0.73611**	**0.82722**
20	**2.00E-04**	**linear**	**0.00**	**0:24:13**	**15.1GB**	**1.74GB**	**51,961,857**	**0.76028**	**0.80556**

Table 1: Freeze layer experimental results

4.3.3 Discussion

Across all configurations in Table 1, the trained models consistently outperformed the baseline in terms of accuracy. It is also empirically observed that higher numbers of frozen layers combined with higher learning rates tended to yield higher accuracy, while models with fewer frozen layers and lower learning rates also achieved higher accuracy. This nuanced approach in adjusting learning rates and number of freeze layers contributes to optimizing model convergence and ultimately achieving improved performance outcomes.

4.4 Low-Rank Adaption

4.4.1 Model Settings

1. Model Base:

Our model is built upon the DeBERTa v3 large architecture, renowned for its robustness and efficiency in natural language processing tasks.

2. Hyperparameters:

r (Radius): Determines the radius of the LoRA mechanism, controlling the scope of local, relational, and global interactions. 'lora_alpha' adjusts the relative weight of the LoRA mechanism compared to other components of the model, and 'lora_dropout' controls the dropout rate applied within the LoRA mechanism to prevent overfitting.

3. Training Settings:

Mixed Precision Training (FP16): Utilizes lower-precision floating-point numbers to accelerate training without sacrificing model performance.

r	LoRA_ alpha	LoRA_dropout	Learning Rate	Batch Sizes	Epochs	Precision
8	16	0.1	'2e-4'	'4' (train), '8' (eval)	'1' for rapid adaptability	FP16 for efficiency
16	32					
32	64					
64	128					

Table 2: Model Setting of LoRA Experiment

4.4.2 Results

r	a	Train Time	GPU Memory	Model Size	trainable_parameters	Accuracy	Accuracy infer
baseline, DeBERTa v3 without training:						0.26583	0.35194
8	16	00:23:30	15.6GB	7.39MB	2,887,682	0.731666	0.7025
16	32	00:23:36	15.6GB	10.53MB	3,674,114	0.744167	0.745556
32	64	00:23:41	15.6GB	16.82MB	5,246,978	0.748056	0.753611
64	128	00:23:49	15.6GB	29.41MB	8,392,706	0.748611	0.801667
128	256	00:23:49	15.6GB	54.47MB	14,684,162	0.723333	0.785278

Table 3: Result of LoRA experiment

5 Analysis

5.1 Training difficulty

5.1.1 Stability

When the rank parameter of LoRA is small (for example, 8 or 16), the training result is unstable. Sometimes it performs well, but when retrained with same config it may perform badly. The same problem occurs when the number of unfrozen layers is large in layer-freezing method.
The instability can be attributed to the sensitivity of the training dynamics. For LoRA, a small rank r means that the adaptation is more constrained, potentially leading to higher sensitivity to initial conditions or stochastic aspects of training such as mini-batch ordering. This sensitivity can cause variability in training outcomes. Similarly, unfreezing more layers increases the number of parameters being updated, which can lead to unstable gradients and training dynamics. In other words, a small rank or too many layers unfrozen may cause catastrophic forgetting.

5.1.2 Time Cost

LoRA takes almost the same time to train when r varies from 8 to 128, while the more layers unfrozen, the more training time needed. LoRA introduces additional parameters and computations that are relatively lightweight and do not significantly increase with r, hence the training time remains stable across different r values. On the other hand, unfreezing more layers in layer-freezing directly increases the number of parameters that need to be updated during backpropagation, leading to longer training times as more computational resources are required.

5.2 Resource Consumption

5.2.1 GPU Memory Usage

Both the LoRA and Layer-freezing method can run without problem on a 16G P100 GPU, with the batch size per GPU for training = 4. But when we try to use the data with RAG (thousands of tokens per sample) as train dataset, an out of GPU memory error is triggered.
This is because the computational complexity increases quadratically with the sequence length n. Due to this quadratic dependency on the sequence length, training of Transformer models is often infeasible for very long sequences. It also explains why we need to apply RAG in the inferring process.

5.2.2 Model Storage Size

LoRA model size is far less than Layer-freezing model size. And LoRA model size increases with the increase of rank. LoRA adds low-rank matrices to existing weights, and the size of these matrices grows with rank. Since each parameter in these matrices requires storage, a larger rank directly translates to more parameters and thus larger model storage requirements. In contrast, layer-freezing does not inherently change the model size but requires storage for the whole updated base model.

5.3 Model Performance

5.3.1 Evaluation Score

LoRA and layer-freezing with proper settings perform better on the test set with RAG inferring than the test set without RAG inferring. And these models have the best performance over all models. This can be attributed to RAG's ability to enrich the model's input with relevant context from an external knowledge source. This additional context can provide the model with more information to make accurate predictions, especially for questions or tasks that require external knowledge not present in the immediate input. Both LoRA and layer-freezing methods benefit from this enriched context as they fine-tune the model to better leverage the additional information provided by RAG.

5.3.2 Generalization Ability

Small-rank LoRA and Few-Layers freezing perform worse on the test set with RAG inferring. The constrained adaptability of small-rank LoRA and few-layers freezing methods limits the models' ability to fully exploit the external knowledge introduced by RAG. This highlights a trade-off between maintaining a model's generalization ability across standard tasks and optimizing for the specialized task of integrating and leveraging external knowledge provided by RAG.

6 Conclusion

In this project, we successfully developed and evaluated a model for science exam multiple-choice questions using advanced techniques in natural language processing (NLP). Our approach incorporated the DeBERTa v3 large model as the base architecture, leveraging its capacity to handle lengthy text sequences effectively. Through the application of two fine-tuning methods, layer freezing and Low-Rank Adaption (LoRA), we explored strategies to enhance model performance and adaptability. Our experiments demonstrated that both layer freezing and LoRA can improve model accuracy, with variations in performance observed across different configurations. Layer freezing proved effective in preserving the model's ability to understand general language structure while adapting to task-specific nuances. Conversely, LoRA offered advantages in memory efficiency and computational resource utilization, albeit with some instability in training dynamics observed for small rank values. Furthermore, we integrated Retrieval-Augmented Generation (RAG) to enrich model prompts with external knowledge from Wikipedia, effectively transforming the science exam into an open-book format. This approach significantly enhanced the model's inference capabilities, leading to more accurate and informed predictions.

However, our study identified several limitations and challenges. Instability in training dynamics, particularly with small rank values in LoRA and extensive layer unfreezing in layer freezing, posed challenges in achieving consistent performance. Additionally, resource constraints, including GPU memory usage and model storage size, limited the scalability of our approach, especially when dealing with lengthy text sequences and large-scale models.

Looking ahead, future research could focus on addressing these limitations by refining fine-tuning techniques, optimizing resource utilization, and exploring alternative approaches to integrating external knowledge sources. Additionally, investigating the transferability of our model to other domains and languages could broaden its applicability and impact. Overall, our findings contribute to the ongoing advancement of NLP models for MCQA tasks and underscore the importance of innovative methodologies in tackling real-world challenges in language understanding and generation.

7 Reference

Kaggle "Kaggle - LLM Science Exam" https://www.kaggle.com/competitions/kaggle-llm-science-exam

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. Advances in neural information processing systems, 30, 2017

Robinson, J. and Wingate, D. "Leveraging large language models for multiple choice question answering". In The Eleventh International Conference on Learning Representations, 2022.

Zheng, C., Zhou, H., Meng, F., Zhou, J., and Huang, M. "On large language models' selection bias in multi-choice questions". arXiv preprint arXiv: 2309.03882, 2023.

SEBASTIAN RASCHKA "Practical Tips for Finetuning LLMs Using LoRA (Low-Rank Adaptation)" https://magazine.sebastianraschka.com/p/practical-tips-for-finetuning-llms

Hu, Edward J., et al. "Lora: Low-rank adaptation of large language models." arXiv preprint arXiv:2106.09685 (2021).

Lewis, Patrick, et al. "Retrieval-augmented generation for knowledge-intensive nlp tasks." Advances in Neural Information Processing Systems 33 (2020): 9459-9474.