Project report:
Language Model for Question-Answering
Mistral on Alpaca

Nguyen Dang Minh Quan,
Pham Lan Phuong

CS312 - Natural Language Processing

Dr. Huynh Viet Linh

Fulbright University Vietnam

May 15, 2024

Contents

1 Introduction **2**

2 Model: Mistral 7B **2**
2.1 Architectural design 2
2.2 Mistral innovation 1: Grouped Query Attention 3
2.3 Mistral innovation 2: Sliding Window Attention 3
2.4 Mistral innovation 3: Rolling Buffer Cache 3

3 Method **4**
3.1 Finetune dataset: cleaned Alpaca 4
3.2 QA benchmark dataset: SQuAD 4
3.3 Evaluator model: Llama 3 8B Instruct 4

4 Results **5**
4.1 Baseline model 5
4.1.1 Task 1 5
4.1.2 Task 2 6
4.2 Finetuned model 6
4.2.1 Task 1 6
4.2.2 Task 2 8

5 Discussion **9**

References **10**

1 Introduction

A Language Model (LM) is a type of artificial intelligence that predicts the probability of a sequence of tokens (usually words). One of the most common application of a LM is a chatbot, which became increasingly more popular after OpenAI's release of ChatGPT. Language models are versatile tools with applications ranging from machine translation, text classification, to named entity recognition. The primary function of a LM for most users is to generate text, which often require the model to have the ability to answer questions.

Question-Answering (QA) is a task in natural language processing, where the goal is to build a program that can answer questions asked by human in natural language. QA has always been considered the most difficult task for a language model, as it requires the model to understand the context well to give good answers.

For this project, we use Mistral 7B as the baseline model, then finetune it on a cleaned Alpaca dataset to gauge the ability of a finetuned small model in question answering.

2 Model: Mistral 7B

2.1 Architectural design

Mistral 7B is a model developed by Mistral AI company. Upon released, Mistral was known to be a highly efficient LM that does not require a large amount of computational resources. Mistral 7B shares a lot of similarity with Llama, with some innovation in the attention mechanism to reduce number of computations while maintain comparable performance.

Mistral is a decoder-only LM, designed to handle tasks involving autoregressive text generation. We chose to work with this model because we did not have a lot of computation resources, and we also want to learn about the architectural differences between it and vanilla transformers.

The core philosophy that we observed from Mistral 7B design is: **Local context matters more.** Mistral 7B has 7 billion parameters, which requires about 14GB to store if each weight is stored as a 8-bit floating point number. The number of parameters can be calculated as Figure 1 below.

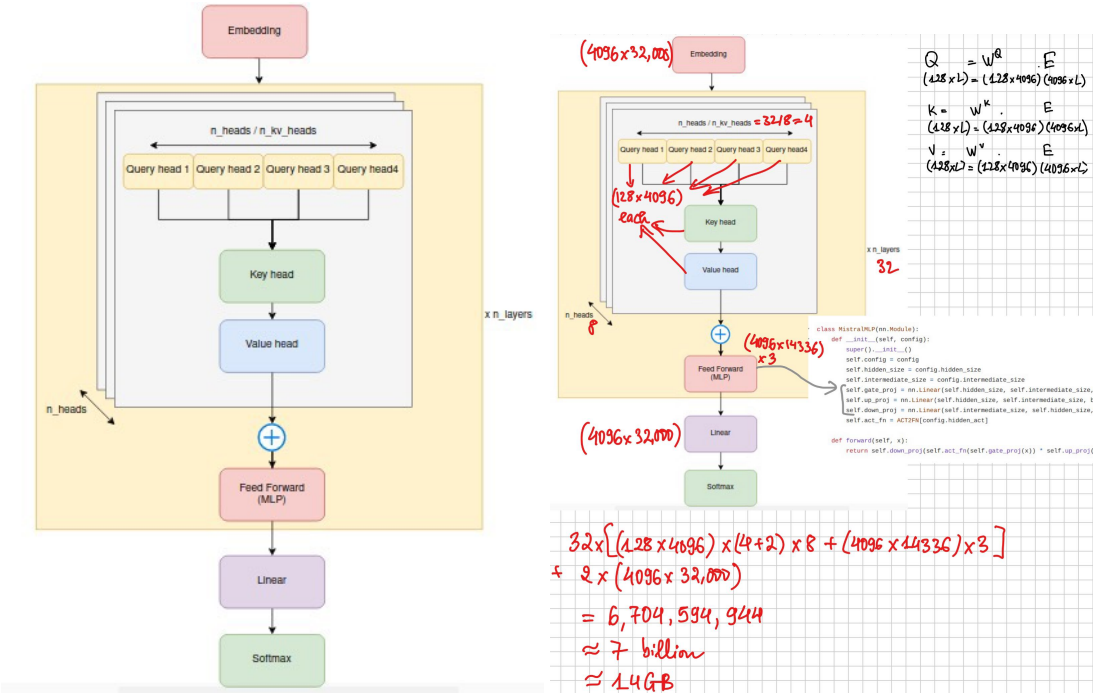


Figure 1: Mistral 7B architecture

2.2 Mistral innovation 1: Grouped Query Attention

Grouped Query Attention (GQA) is a self-attention mechanism that modified and built upon Multi-Head Attention (MHA) architecture. The idea is to group multiple query heads for each key and value heads. This reduces the number of parameters of the model, but more significantly recduces the number of dot product needs to be computed for each token [1].

Without GQA, each attention head would need to store 1.6 billion parameters. With GQA, the number of parameters becomes 8 million (half). Although this does not seem like a lot, this allows for higher batch size (processing more data at once). which is good for real-time applications like a chatbot [2].

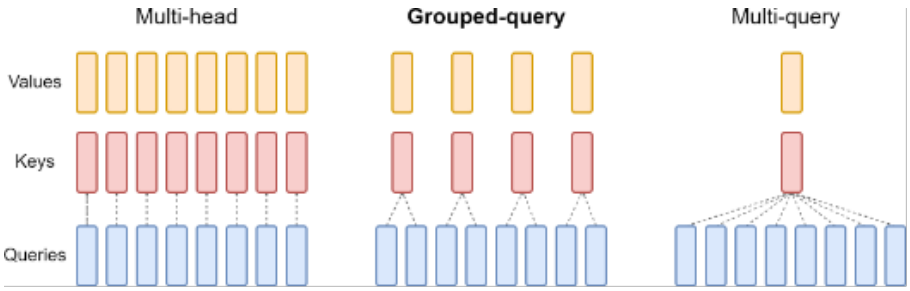


Figure 2: Grouped Query Attention

2.3 Mistral innovation 2: Sliding Window Attention

Sliding Window Attention (SWA) is another modification of the original attention architecture. SWA can deal wth long context at a lower computational cost comparing to Multi-Head Attention.

Context tokens that are further than a fixed window size will be masked and therefore not computed. This reduces the number of dot products needed for each token from a quadratic order $O(L^2)$, where L is the input length, to a constant $O(1)$.

This, however, does not completely disregard context tokens outside of the window. The token that is very far away will indirectly influence the current token in a recursive manner. This was reported to runs twice as fast as a baseline attention layer [3] [4].

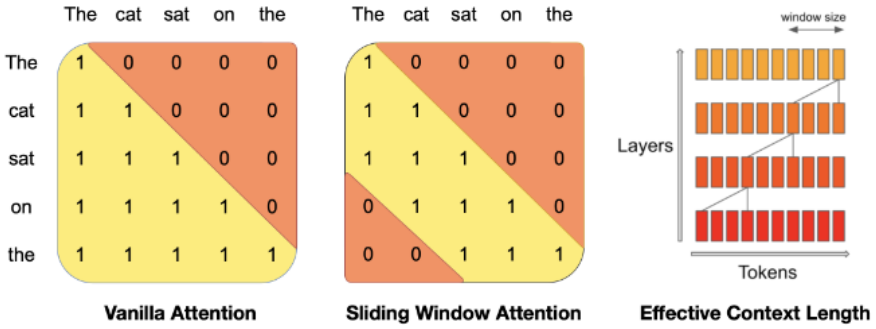


Figure 3: Sliding Window Attention

2.4 Mistral innovation 3: Rolling Buffer Cache

A Rolling Buffer Cache is a fixed size cache used to store the result of most recent tokens. Because of this, the model can avoid re-computing previous tokens. This mechanism exists to support the implementation of SWA and speed up computation time. Note that because the buffer size is fixed, it does not pose a big burden on computer memory, as it does not scale with input size.

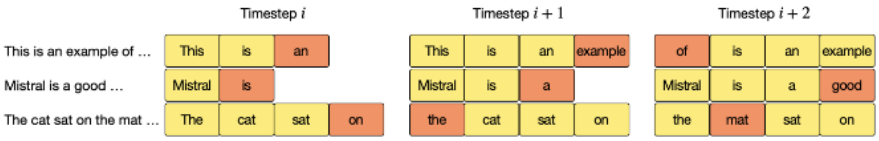


Figure 4: Caption

3 Method

1. Import pretrained Mistral 7B to use as baseline model
2. Finetune baseline model on cleaned Alpaca dataset
3. Generate 250 samples for the question-generation task
4. Generate 250 samples for the question-answering task
5. Evaluate the finetuned model's inference result:
    - Using Llama 3 8B Instruct with system prompt to grade task 1 result
    - Ussing Llama 5 8B Instruct with system prompt to grade task 2 result
    - Grade task 2 by comparing inference result with SQuAD dataset
6. Visualize the scores

3.1 Finetune dataset: cleaned Alpaca

The Alpaca dataset is a collection of 52,000 instructions and demonstrations designed to facilitate instruction-tuning for language models, aiming to improve their ability to follow complex instructions. We use a cleaned version of the original Alpaca dataset, as the documentation claimed that it reduces the instances of hallucination, empty output, and wrong answers [5].

When choosing the dataset to finetune the baseline model, we were looking for the popular dataset for this job. There is not a lot of reasons further than this as we did not know the common practices of the field.

3.2 QA benchmark dataset: SQuAD

The Stanford Question Answering Dataset (SQuAD) is a popular text comprehension dataset consisting of 100,000+ questions posed by internet users on a set of Wikipedia articles [6] [7].

SQuAD's reference answers are a result of an extractive model, which is inherently different to Mistral being a generative model. We shall see that the Exact Match and F1 score of finetuned Mistral is not reflective of its ability to answer questions at all. We do not recommend using SQuAD, Exact Match, and F1 score to evaluate decoder-based model for this reason.

3.3 Evaluator model: Llama 3 8B Instruct

Llama 3 is a family of LM made and published by Meta, which comes in 8B and 70B size. The Llama 3 instruction tuned models are optimized for dialogue use cases and outperform many of the available open source chat models on common industry benchmarks [8].

We use Llama 3 8B Instruct, which is a Llama 3 8B model that has been finetuned to follow instructions. Using a 8B model to grade a 7B model is not the best idea, but we went with it because it works with the time and computation resources we had.

4 Results

4.1 Baseline model

4.1.1 Task 1

The baseline model demonstrates some ability to generate questions and answers, so we believe that this model is suitable for the Question-Answering task.

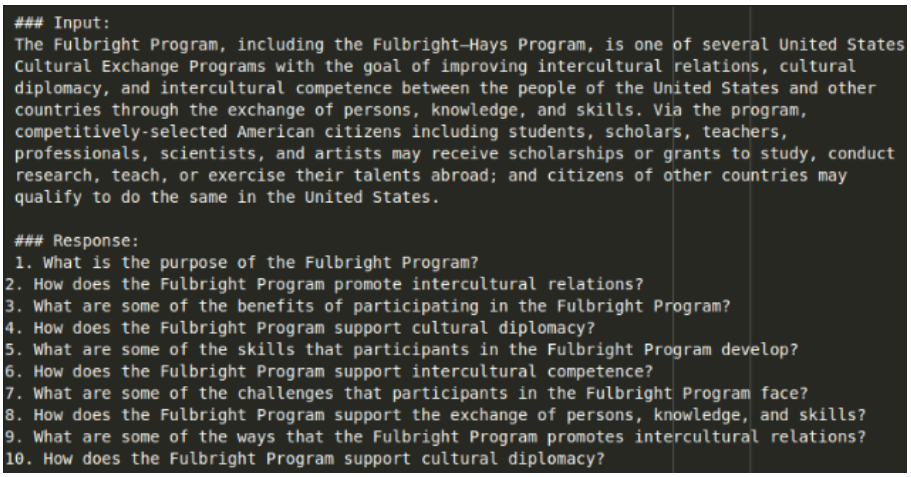


Figure 5: Baseline model can generate decent questions for some contexts

However, there are multiple instances of the model making too simple what-questions(Figure 6), repeating questions to meet quota (10 questions per document) (Figure 7), or sometimes just quit generating mid-way and move on to next document (Figure 8).

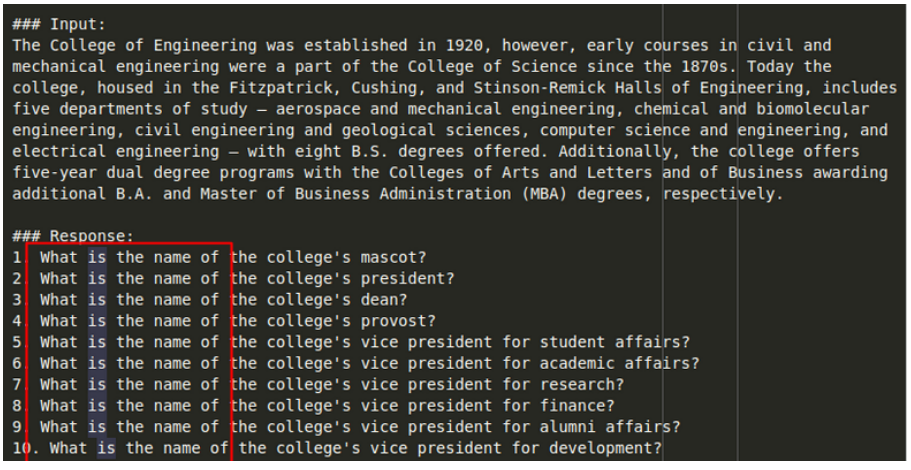


Figure 6: Baseline making simple, monotonous questions

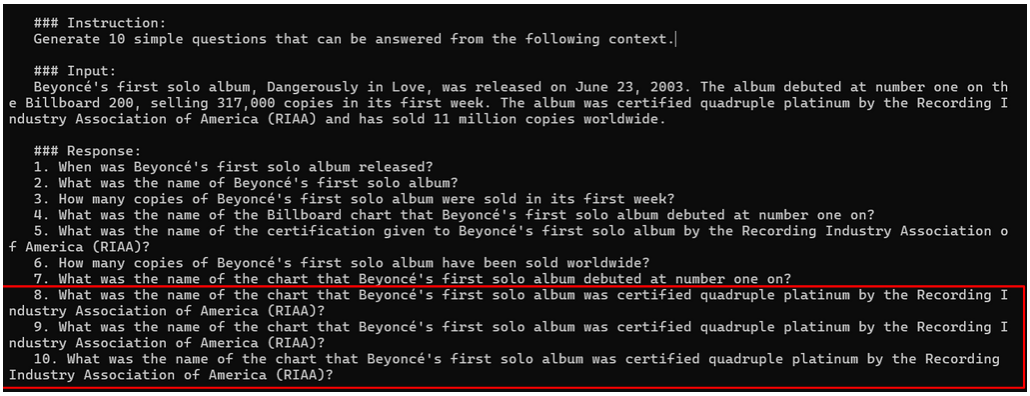


Figure 7: Baseline repeating questions to meet quota

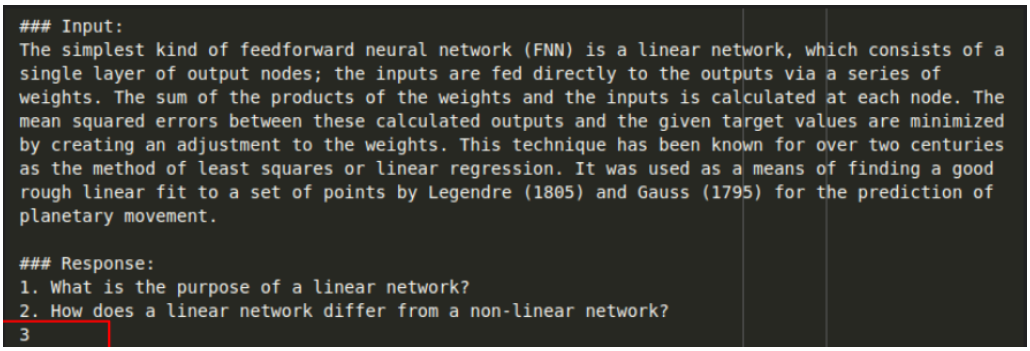


Figure 8: Baseline quit generating prematurely

4.1.2 Task 2

Baseline model failed to retrieve answer from the given context, despite the fact that the answer is present in the given context.

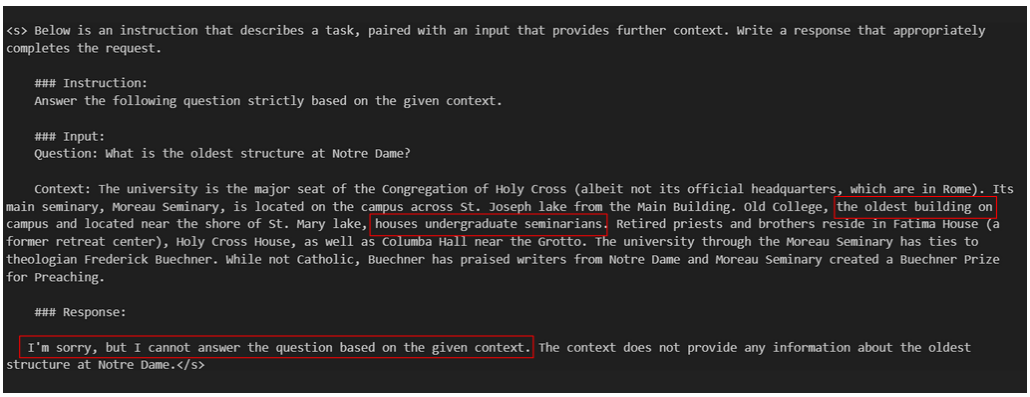


Figure 9: Baseline unable to find answer in context

4.2 Finetuned model

4.2.1 Task 1

Four criteria were used to evaluate the model's output. **Relevance**, to see if the questions address the main subject of the context. **Clarity**, which is the lack of ambiguity or multiple possible meanings of a question. **Depth**, which is the sophistication of the question. **Answerability**, to see if the answer to the generated question can be found in the context.

We used this prompt to ask the evaluator model to grade the questions:

```
You will be given a context and a question about content in that context.
<context>
<question>

Your task is to assess the question's quality using the following metrics:
- Relevance: Does the question directly address the information in the context
and focus on a specific aspect?
- Clarity: Is the question easy to understand, unambiguous, specific, and
avoids vague terms or generalizations?
- Depth: Does the question encourage deeper analysis or critical thinking,
going beyond simple recall and potentially sparking further questions?
- Answerability: Can the question be reasonably answered based on the context,
with limited ambiguity or multiple interpretations?

Provide your assessment in JSON format, using the keys \"relevance\",
\"clarity\", \"depth\" and \"answerability\" each with a score ranging from
0 to 10.

Remember you hold a really high standard and you don't easily give out
perfect or near perfect score. Only include the JSON evaluation, without any
additional explanation.
```

Which yields the following results:

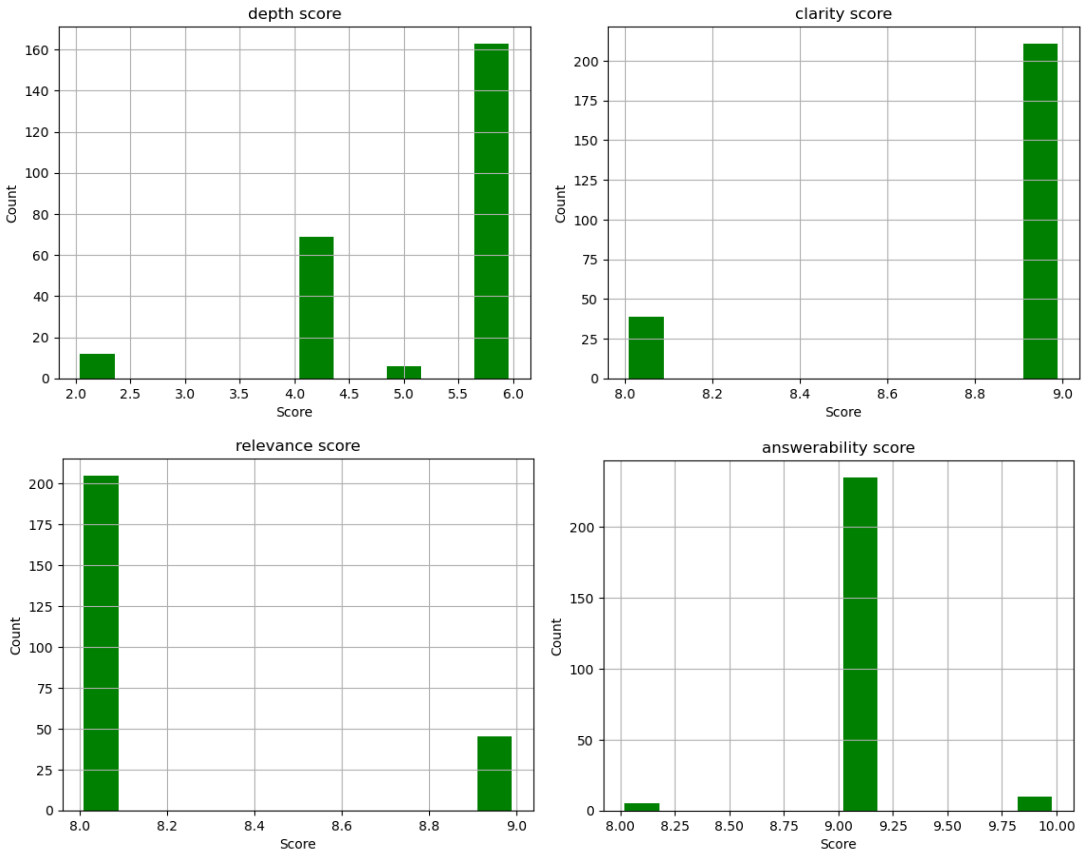


Figure 10: Score distribution of generated questions

This result aligns with our expectation. The finetuned Mistral was able to make clear, relevant questions whose answers can be found in the context. The questions were not highly critical or advanced–as judged

by the authors who are college students with ability to ask questions about abtract concept–which was relfected in a lower depth score.

4.2.2 Task 2

To evaluate the quality of the answers generated in task 2, we use five criteria. **Factuality**, which is the factually correctnesss of the answer. **Relevance**, to see if the questions address the main subject of the context. **Completeness**, to see if the question was answered in full. **Conciseness**, because we don't want the output to be too verbose. **Clarity**, which is the lack of ambiguity or multiple possible meanings of a question.

We used this prompt to ask the evaluator model to grade the answers:

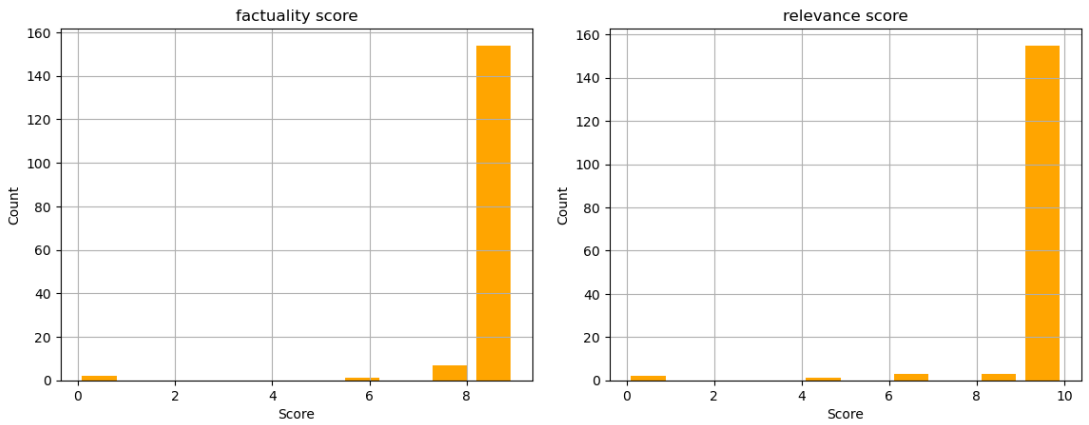
```
You will be given a context, a question about content in that context and
an answer.
<context>
<question>
<answer>

Your task is to assess the answer's quality using the following metrics:
- Factuality: Is the answer factually correct and does it align with
established knowledge or reliable sources?
- Relevance: Does the answer directly address the question and focus on
the specific information requested?
- Completeness: Does the answer provide all necessary information to fully
satisfy the question, without leaving out any crucial details?
- Conciseness: Is the answer clear, succinct, and free of unnecessary
wordiness while conveying essential information?
- Clarity: Is the answer well-structured, easy to understand, and does it
flow logically with clear connections between ideas?

Provide your assessment in JSON format, using the keys \"factuality\",
\"relevance\", \"completeness\", \"conciseness\", and \"clarity\", each with
a score ranging from 0 to 10.

Remember you hold a really high standard and you don't easily give out
perfect or near perfect score. Only include the JSON evaluation, without any
additional explanation.
```

From which we get the result as in Figure 11. The lowest score is of conciseness. The answer are often verbose, which is a common problem of machine-generated text from transformers-based model. Completeness is the second lowest score, which we have not been able to explain why. Factuality, relevance, and clarity score were all high as expected.



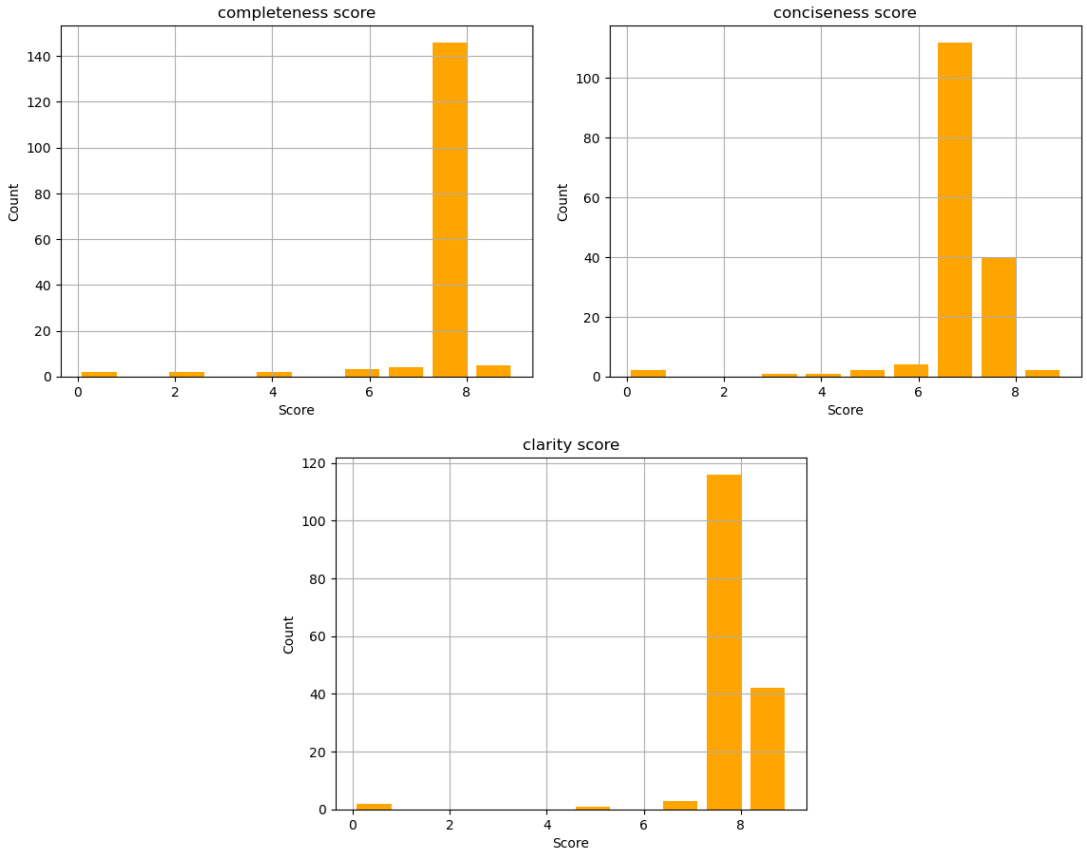


Figure 11: Score distribution of generated answers

We also tried to benchmark the answers of task 2 against SQuAD, and unsuprisingly the score was very bad. SQuAD's reference answers are obtained with an extractive model, so the answers are very short, often contains only of phrases not sentences. We do noth think this score accurately reflects the model's ability to do QA task, so we do not recommend using SQuAD, Exact Match, and F1 score to evaluate decoder-based model for this reason.

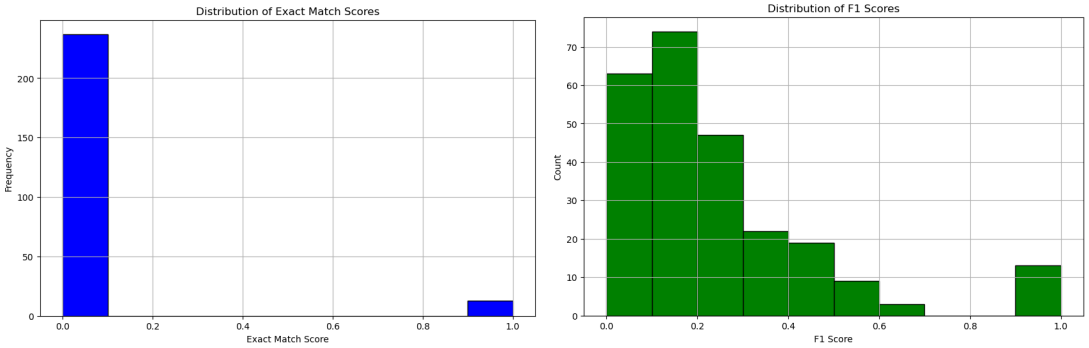


Figure 12: Very poor Exact Match and F1 score

5 Discussion

Mistral 7B shows potential to be a good LM to do question-answering. The architectural design of the attention layer (GQA and SWA) implies that the performance of this model will degrade as the context gets longer and longer. However, for the projects of this tasks where each paragraph is considered a document, it performed relatively well.

We have probably seen this phenomenon in the class demonstration, where the test texts consists of 4-5 paragraphs, and the model answers "I cannot find the answer" 4/6 times where the question is answerable. On the other hand, the live demonstration shows that our model was till able to ask good questions for longer contexts.

References

[1] Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. Gqa: Training generalized multi-query transformer models from multi-head checkpoints, 2023.

[2] Wankyu Choi. Deep-Dive-Into-AI-With-MLX-PyTorch/deep-dives/001-mistral-7b/README.md at master · neobundy/Deep-Dive-Into-AI-With-MLX-PyTorch — github.com. https://github.com/neobundy/Deep-Dive-Into-AI-With-MLX-PyTorch/blob/master/deep-dives/001-mistral-7b%2FREADME.md, 2024. [Accessed 15-05-2024].

[3] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023.

[4] Unsloth-MistralAI. unsloth/mistral-7b-bnb-4bit · Hugging Face — huggingface.co. https://huggingface.co/unsloth/mistral-7b-bnb-4bit, 2024. [Accessed 15-05-2024].

[5] Stanford. yahma/alpaca-cleaned · Datasets at Hugging Face — huggingface.co. https://huggingface.co/datasets/yahma/alpaca-cleaned, 2023. [Accessed 15-05-2024].

[6] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text, 2016.

[7] Stanford. rajpurkar/squad · Datasets at Hugging Face — huggingface.co. https://huggingface.co/datasets/rajpurkar/squad, 2022. [Accessed 15-05-2024].

[8] Meta. NousResearch/Meta-Llama-3-8B-Instruct · Hugging Face — huggingface.co. https://huggingface.co/NousResearch/Meta-Llama-3-8B-Instruct, 2024. [Accessed 15-05-2024].

Acknowledgement

We would like to thank Dr. Huynh Viet Linh for assigning this project so that we get to learn something new on our own. We would also like to thank our friend at ViettelAI to let us borrow their servers to do finetuning.