Report | LLM Assignment 2

Garv Makkar 2021530 | IIITD

The objective of the assignment:
Experiment with various open-source large language models (LLMs) and explore different prompting techniques. Analyze and compare the results to gain insight into the underlying reasons for their differences.

LLMs used:

- google/gemma-2b-it
- microsoft/Phi-3.5-mini-instruct
- meta-llama/Meta-Llama-3-8B-Instruct

Dataset used to compare performance:
Subset of Test set of cais/mmlu where subject is college_mathematics.

Prompting Methods used:

- Zero-Shot Prompting
- Chain of thought - Zero Shot Prompting
- ReAct Prompting

Notes and Assumptions:

- In this report, Gemma, Phi, and Llama refer to these particular versions of their models.
- Used Llama 3, not 3.1.
- All the models used are 4-bit Quantized of the above-mentioned models.
- The actual inference data size is 100 samples, but I took subsets as it took a lot of time to run and check for them.
    - For ‘Gemma’, it is the first 30 samples.
    - For ‘Phi’, it is the first 20 samples.
    - For ‘Llama’, it is the first 10 samples.
- The runtime reported is scaled for 10 samples to compare it.
- Once the solutions are generated by these models, the correct option is extracted using a classifier, specifically the ‘facebook/large-bart-mnli’ model.

My Hypothesis for the results:

- For a particular model:
    - The order for run-time should be: zero-shot < chain-of-thought-zero-shot < react
    - The order for accuracy should be: zero-shot < chain-of-thought-zero-shot < react
      This should be because the bigger and better prompt should give better results but would take more time to process. Also, the ReAct prompt gives tools to LLM for calculations and Google searches.
- Across the models:
    - Higher the model size, higher the accuracy.
    - Higher the model size, higher the run time.

Observations

- **Gemma**
    - Zero-Shot
        Accuracy (Out of 1): **0.3**
        Time duration (In seconds): **95.92**
    - Chain of thought (Zero-Shot)
        Accuracy (Out of 1): **0.33**
        Time duration (In seconds): **134.05**
    - ReAct
        Accuracy (Out of 1): **0.2**
        Time duration (In seconds): **85.75**
- **Phi**
    - Zero-Shot
        Accuracy (Out of 1): **0.25**
        Time duration (In seconds): **415.5**
    - Chain of thought (Zero-Shot)
        Accuracy (Out of 1): **0.3**
        Time duration (In seconds): **500.35**
    - ReAct
        Accuracy (Out of 1): **0.35**
        Time duration (In seconds): **561.49**
- **Llama**
    - Zero-Shot
        Accuracy (Out of 1): **0.4**
        Time duration (In seconds): **584.95**
    - Chain of thought (Zero-Shot)
        Accuracy (Out of 1): **0.4**
        Time duration (In seconds): **779.78**
    - ReAct
        Accuracy (Out of 1): **0.3**
        Time duration (In seconds): **732.10**

Comparing the results

- For particular models:
    - Gemma:
        - React performed poorly. If we look at responses generated in code, the model did not follow react prompting at all! It failed and hence took the least time to run and gave the poorest accuracy.
        - Zero-shot and chain-of-thought followed the hypothesis.
    - Phi:
        - All the techniques give results as hypothesized. Accuracy and run-time increase in the same order as mentioned.
    - Llama:
        - Zero-shot and chain-of-thought-zero-shot methods show same accuracy. Maybe because inference set was small, but the run time difference between them follows our hypothesis.
        - ReAct prompting shows a dip in performance and takes almost equivalent time as chain-of-thought-zero-shot.
- Across the models:
    - If we ignore ReAct results,
      Llama is better than gemma which is better than phi in terms of accuracy.
      Run-time follows our hypothesis. Bigger models have higher run-time
    - If we consider ReAct results,
      Gemma fails completely for ReAct strategy (Atleast in the way I did)
      Phi performed better than Llama.

Reasoning and discussion about the results

To find the reasoning, we shall understand the use case of models, their size, and the data models were trained on.

Gemma
Model Size: 2.51B params
Gemma-2B-IT is a lightweight, text-to-text, decoder-only large language model from Google.
Designed for tasks like question answering, summarization, and reasoning.
Trained on 6 trillion tokens from diverse sources: Web Documents for varied linguistic styles, Code for programming-related tasks, Mathematics for logical reasoning.

Phi
Model Size: 3.82B params
Phi-3.5-mini is a lightweight, open model, focused on high-quality, reasoning-dense tasks.
It supports a 128K token context length, enabling the handling of long tasks.
Dense decoder-only Transformer
Primary Use Cases: For memory/compute constrained environments and latency-bound scenarios, Suitable for tasks requiring strong reasoning, especially for code, math, and logic, Designed for general-purpose AI applications in both commercial and research contexts.
Multilingual Support
Trained on 3.4T tokens from high-quality data sources, including filtered public documents, synthetic “textbook-like” data, and supervised chat format data.

Llama
Model Size: 8.03B params
Auto-regressive language model with an optimized Transformer architecture.
It uses supervised fine-tuning (SFT) and reinforcement learning with human feedback (RLHF) to align with human preferences for helpfulness and safety.
Intended Use: This is for commercial and research purposes across multiple languages, Instruction-tuned models are suitable for assistant-like chat applications, Pretrained models can be adapted for various natural language generation tasks, such as synthetic data generation and distillation.
Pretrained on ~15 trillion tokens from publicly available sources.
Fine-tuned on 25M+ synthetically generated examples.

References: google/gemma-2b-it · Hugging Face,
microsoft/Phi-3.5-mini-instruct · Hugging Face,
meta-llama/Meta-Llama-3-8B-Instruct · Hugging Face,
2403.08295 (arxiv.org), 2407.21783 (arxiv.org), 2404.14219 (arxiv.org)

After knowing about the above models. We can draw conclusions and perform an analysis. We will discuss this in 2 parts. With and without the ReAct method.

- Comparison by not considering the ReAct method.
    - Across models:
        - Llama performs best because of its bigger size, bigger training data, and better training strategies following SFT and RLHF. Gemma performs better than Phi possibly because of its training strategy.
        - Run time was as hypothesized, bigger models take more time to understand and process.
    - For particular models:
        - The run time increases for the longer prompt as expected as it takes longer to process a longer prompt.
        - The performance increases because of the better prompt which gives context for math and tells it to think step by step to solve reasoning.
        - Llama would have also shown this possibly if higher inference set was used.

- Discussing results of ReAct method
    - For particular models
        - Gemma entirely failed for this method so it is out of comparison. It failed because its training did not incorporate the step-by-step reasoning kind of process.
        - Phi's run time and accuracy both increased for ReAct. It incorporated the step-by-step strategy and followed the prompt template.
        - Llama's ReAct performance dipped in comparison to other methods because it is not good with long context. [Pg 2 of paper]
    - Across models
        - No point comparing Gemma.
        - Phi was better than Llama because it is not good with long contexts.