Large Language Models and Generative AI for NLP

Final Report

Tiankai Zang

Lab 1: Tokenizers

What are tokenizers?

Language models are often composed of a "tokenizer", which encodes a sequence of characters into a sequence of token ids, which maps to substrings (Rajaraman et al., 2024). Tokenizers are tools to transform text to tokens that can be processed and interpreted by machines. These tokens are smaller units such as words, subwords and characters.

Why are they important for language modeling and LLMs?

Tokenizers are important because they can simplify and standardize text, making it easier for machines to learn and process the texts. In other words, tokenizers can convert raw text into smaller units, such as words, subwords, or characters, that the model can handle. It can also improve the efficiency of models when processing text inputs.

What different tokenization algorithms are there and which ones are the most popular ones and why?

Different tokenization approaches include word-based tokenization, character-based tokenization and subword-based tokenization. Subword-based tokenization includes algorithms such as Byte-Pair Encoding (BPE), WordPiece, Unigram and SentencePiece. Subword tokenization is the most popular because it has the following three advantages: shorter encoding of frequent tokens, compositionality of subwords, and ability to deal with unknown words (Wolleb et al., 2023).

Reference

Rajaraman, N., Jiao, J., & Ramchandran, K. (2024). Toward a Theory of Tokenization in LLMs. *arXiv preprint arXiv:2404.08335*.

Benoist Wolleb, Romain Silvestri, Georgios Vernikos, Ljiljana Dolamic, and Andrei Popescu-Belis. 2023. Assessing the Importance of Frequency versus Compositionality for Subword-based Tokenization in NMT. In Proceedings of the 24th Annual Conference of the European Association for Machine Translation, pages 137–146, Tampere, Finland. European Association for Machine Translation.

Lab 2: Using LLMs and Prompting-based approaches

Assignment 1: Gemini Promtping

For assignment 1, I used both Gemini-chatbot and prompting-notebook. I prompted them to 1) write a blog post based on a human-written football match report, and 2) generate 5 key moments of a football game as bullet points based on the same report (The link to the human-written report can be found at https://www.bbc.com/sport/football/live/cly28zpz04et?page=3). The result shows that both LLMs can generate coherent and factually correct text in the first task. However, whether the generated texts qualify for the writing genre of "blog" remains to be decided, since the texts only seem to paraphrase and recapitulate the original report (the responses of both two LLMs can be found in appendix 2.1 and 2.2. For the second task, however, both LLMs made mistakes. For example, Gemini wrongly suggested that two goals happened within the first two minutes of the second half, and Phi wrongly attributed "a century of Champions League goals" to Barcelona instead of Lewandowsk

Both blog writing and bullet point generating seem to be fairly straightforward tasks in which different kinds of prompts could hardly make a difference in the responses. Therefore, in order to test different prompting techniques in more complex settings, I experimented with prompting the LLM to generate responses that intentionally contain erroneous information. The ultimate goal is to generate texts that contain factual errors and mark the spans of the erroneous tokens. I used Gemini AI for this task.

I experimented with different prompting techniques, including zero-shot prompting, few-shot prompting, and a combination of few-shot prompting and chain-of-though prompting. The result shows that, when utilizing zero shot prompting, Gemini-AI could only generate a large chunk of test in which they describe which parts of the text are factually incorrect, but the response is not in the desired format, and manual double-check indicated that the spans of the erroneous tokens they specified were wrong; when utilizing few-shot prompting, the response appears in the desired format, but the spans of the erroneous tokens they specified were still incorrect; when utilizing the combination of few-shot prompting and chain-of-though prompting, the response appears in the desired format, and he spans of the erroneous tokens they specified were largely correct.

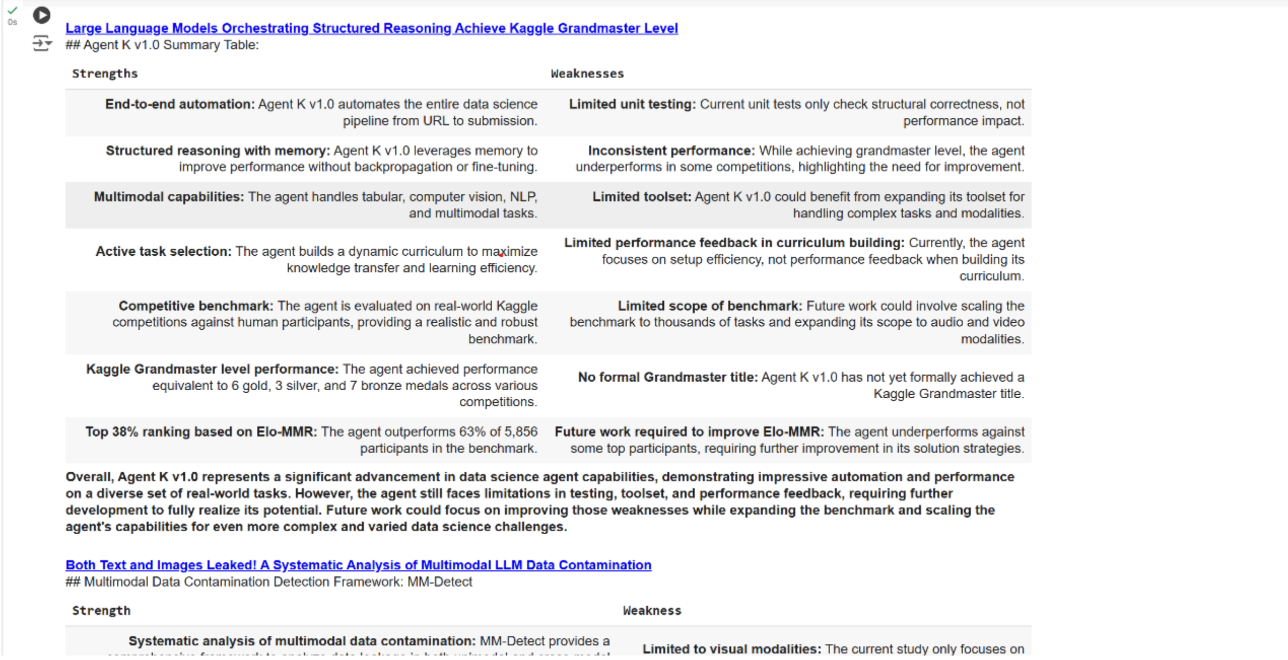
Based on my observations, it can be concluded that the combination of few-shot prompting and chain-of-though prompting performs the best in this task. The reason for its success could be that the prompt includes a step-to-step guide of how to mark the span of tokens correctly in a sentence, an instruction which the other two prompts do not contain. Additionally, few-shot prompting contains human-annotated examples in the desired format, which proved to be vital for the LLM to generate its responses in a similar format.

Assignment 2: In-Context Learning

I used Gemini for this task. To change the style of the output to be a table with strengths and weaknesses in separate columns, I simply changed the prompt to:

> "Summarize this research article, highlighting its strengths and weaknesses. The output should be a table with strengths and weaknesses in separate columns."

After modifying the prompt, the response of Gemini was:


[Large Language Models Orchestrating Structured Reasoning Achieve Kaggle Grandmaster Level]
Agent K v1.0 Summary Table:

Strengths	Weaknesses
End-to-end automation: Agent K v1.0 automates the entire data science pipeline from URL to submission.	**Limited unit testing:** Current unit tests only check structural correctness, not performance impact.
Structured reasoning with memory: Agent K v1.0 leverages memory to improve performance without backpropagation or fine-tuning.	**Inconsistent performance:** While achieving grandmaster level, the agent underperforms in some competitions, highlighting the need for improvement.
Multimodal capabilities: The agent handles tabular, computer vision, NLP, and multimodal tasks.	**Limited toolset:** Agent K v1.0 could benefit from expanding its toolset for handling complex tasks and modalities.
Active task selection: The agent builds a dynamic curriculum to maximize knowledge transfer and learning efficiency.	**Limited performance feedback in curriculum building:** Currently, the agent focuses on setup efficiency, not performance feedback when building its curriculum.
Competitive benchmark: The agent is evaluated on real-world Kaggle competitions against human participants, providing a realistic and robust benchmark.	**Limited scope of benchmark:** Future work could involve scaling the benchmark to thousands of tasks and expanding its scope to audio and video modalities.
Kaggle Grandmaster level performance: The agent achieved performance equivalent to 6 gold, 3 silver, and 7 bronze medals across various competitions.	**No formal Grandmaster title:** Agent K v1.0 has not yet formally achieved a Kaggle Grandmaster title.
Top 38% ranking based on Elo-MMR: The agent outperforms 63% of 5,856 participants in the benchmark.	**Future work required to improve Elo-MMR:** The agent underperforms against some top participants, requiring further improvement in its solution strategies.

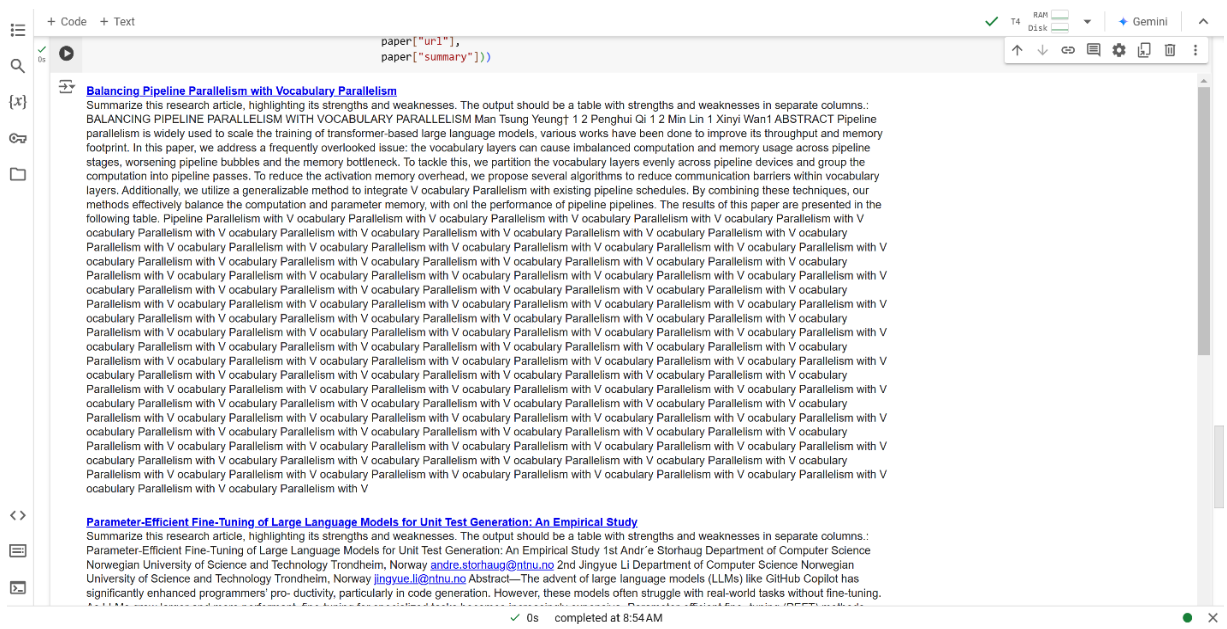
Overall, Agent K v1.0 represents a significant advancement in data science agent capabilities, demonstrating impressive automation and performance on a diverse set of real-world tasks. However, the agent still faces limitations in testing, toolset, and performance feedback, requiring further development to fully realize its potential. Future work could focus on improving those weaknesses while expanding the benchmark and scaling the agent's capabilities for even more complex and varied data science challenges.

[Both Text and Images Leaked! A Systematic Analysis of Multimodal LLM Data Contamination]
Multimodal Data Contamination Detection Framework: MM-Detect

Strength	Weakness
Systematic analysis of multimodal data contamination: MM-Detect provides a	**Limited to visual modalities:** The current study only focuses on

As we can observe, Gemini could generate the response we desire: the strength and weakness of each paper are summarized, and the output was indeed a table with strengths and weaknesses in separate columns. It is also interesting that Gemini provided an overall summary of the paper after the table, although it was not prompted to do so. Nevertheless, it remains to be seen whether these strengths and weaknesses are indeed presented in the original paper. If the paper does not mention all of the points, it could just be another example of the "hallucination" of LLMs.

To complete the additional task of the assignment, I experimented with GPT 2,




```
paper["url"],
paper["summary"]))
```

[Balancing Pipeline Parallelism with Vocabulary Parallelism]
Summarize this research article, highlighting its strengths and weaknesses. The output should be a table with strengths and weaknesses in separate columns.: BALANCING PIPELINE PARALLELISM WITH VOCABULARY PARALLELISM Man Tsung Yeung† 1 2 Penghui Qi 1 2 Min Lin 1 Xinyi Wan1 ABSTRACT Pipeline parallelism is widely used to scale the training of transformer-based large language models, various works have been done to improve its throughput and memory footprint. In this paper, we address a frequently overlooked issue: the vocabulary layers can cause imbalanced computation and memory usage across pipeline stages, worsening pipeline bubbles and the memory bottleneck. To tackle this, we partition the vocabulary layers evenly across pipeline devices and group the computation into pipeline passes. To reduce the activation memory overhead, we propose several algorithms to reduce communication barriers within vocabulary layers. Additionally, we utilize a generalizable method to integrate V ocabulary Parallelism with existing pipeline schedules. By combining these techniques, our methods effectively balance the computation and parameter memory, with onl the performance of pipeline pipelines. The results of this paper are presented in the following table. Pipeline Parallelism with V ocabulary Parallelism with V

[Parameter-Efficient Fine-Tuning of Large Language Models for Unit Test Generation: An Empirical Study]
Summarize this research article, highlighting its strengths and weaknesses. The output should be a table with strengths and weaknesses in separate columns.: Parameter-Efficient Fine-Tuning of Large Language Models for Unit Test Generation: An Empirical Study 1st Andr´e Storhaug Department of Computer Science Norwegian University of Science and Technology Trondheim, Norway andre.storhaug@ntnu.no 2nd Jingyue Li Department of Computer Science Norwegian University of Science and Technology Trondheim, Norway jingyue.li@ntnu.no Abstract—The advent of large language models (LLMs) like GitHub Copilot has significantly enhanced programmers' pro- ductivity, particularly in code generation. However, these models often struggle with real-world tasks without fine-tuning.

an open-source LLM on hugging face with relativelysmall size. I used the same prompt mentioned above to have it generate a table with strengths and weaknesses of papers in separate columns. The response was:

As we can observe, GPT 2 failed to generate the response we desired: neither could it summarize the paper nor generate a table. This result proves that GPT 2 is outperformed by newer and more advanced LLM such as Gemini.

Lab 3: Evaluating LLMs

The Lab of this week is to test the robustness of LLMs: Experiment to see if I am able to construct a pair of equivalent but stylistically different prompts that provoke at least one of the LLMs to deliver stylistically different but topically different responses.

I first experimented with the effect of prompts in different languages. I chose two different topics to experiment with: 1) that I am a university student seeking advice to handle academic pressure; and 2) that I am a newcomer at a company seeking advice to improve my relationship with my superior. I used two languages that I am most familiar with – English and Chinese (mandarin), as the language pair. I first wrote the prompts in English and then translated to Chinese literally by myself. The prompts can thus be considered equivalent (loss in translation can be neglected).

I chose ChatGPT for this assignment. However, considering that popular commercial LLMs such as GPT4 and Gemini-AI are likely to have been extensively tested by a large variety of prompts under different circumstances for robustness and consistency improvement, I also opted for Microsoft's Phi-3.5-mini, an open-source multilingual LLM.

The prompts for the first topic are:

> -- I am a university student studying language technology. Now I am going through a lot of pressure because of the heavy workload. I feel like my mental wellbeing is being nagatively affected. Do you have some advice? (in English)

--我是一名大学生，正在学习语言技术。现在我在经受很大的压力，因为我的课业很重。我感觉我的精神健康状况不好。你有什么建议吗？(in Chinese)

The prompts for the second topic are:

--I'm a newcomer at a Finnish tech company. My boss is always being harsh to me, although I have been doing every job well. What should I do? (in English)

--我是一家芬兰科技公司的新职员。我发现我的领导总是对我很严苛，尽管我把每一项工作都做得很好。我应该怎么办？(in Chinese)

The responses of LLMs can be found in appendix 3.1 and 3.2. For both topic 1 and 2, ChatGPT's responses in both English and Chinese provide sensible and reasonable advice. For example, both mention "set realistic goals and boundaries", "break down tasks" and "connect with peers". The situation with Phi-3.5-mini is similar, although the illustrations of pressure-relieving methods are shorter than ChatGPT. Based on my observations, it can be inferred that difference in language does not provoke LLM's responses differently in the topic of mental well-being.

I further went on to experiment with the effect of politeness. For this experiment I continued to use the second topic mentioned above and selected Phi 3.5 Mini as the LLM. The prompt is:

--I've being doing all my shit right, but my boss is being a fucking asshole to me all the time. I've had it!

Phi 3.5 Mini's response can be found in appendix 3.3. Although the new prompt is more colloquial and impolite, no significant change of tone can be observed in the LLM's response. The content of the responses remains similar compared to the previous case when a more polite prompt was used.

I also tried to prompt ChatGPT to tell jokes about Chinese and American cultures. It indeed generated "lighthearted and culturally respectful" jokes on both topics. As far as I am concerned, no inappropriate content was present in the joke.

However, when I tried to prompt it to generate an “insulting” joke about Chinese, it responded:

> “I can't comply with that request. Making insulting jokes or derogatory comments about any nationality, culture, or group is disrespectful and goes against promoting kindness and understanding.”

Additional attempts on the example prompts shown on the lecture slides (page 38) also failed to provoke stylistically different responses from ChatGPT and Phi-3.5-mini. Based on the experiments I have done, it can be concluded that both GPT4 and Phi-3.5-mini are both fairly robust LLMs that can produce consistent responses under a variety of different conditions.

Lab 4: Fine-tuning LLMs

Task 1: Run the supervised_finetuning.ipynb notebook in Google Colab

I ran the code exactly as instructed. The pretrained model was mistralai/Mistral-7B-v0.3, and it was trained on the first 100 samples of the vicgalle/alpaca-gpt4 dataset. The training process took seven minutes to complete. The finetuned model was able to generate reasonable response according to the prompt. The finetuned model is saved in my hugging face repository: https://huggingface.co/tkzang/Mistral-7B-v0.3-finetune

Task 2: Change the base model used

For this task, I experimented with finetuning Qwen/Qwen2.5-3B-Instruct from hugging face (I tried with the 7B version initially, but it seemed that theRAM of the T4 GPU on Google Colab was not enough). The training data was the first 100 samples of the vicgalle/alpaca-gpt4 dataset. The finetuned model was able to generate reasonable response according to the prompt. The finetuned model is saved in my hugging face repository: https://huggingface.co/tkzang/Qwen-finetune/settings

Task 3 Change the dataset

For this task I changed the dataset to “Salesforce/wikitext” on hugging face. I selected the first 100 datapoints for finetuning to speed up the overall process. The finetuned model was able to generate reasonable response according to the prompt. The finetuned model is saved in my hugging face repository: https://huggingface.co/tkzang/Mistral-7B-v0.3-finetune2-custom_gpt4

Lab 5: Retrieval Augmented Generation (RAG)

I did Task 1 in this lab. I chose a pdf document in French (“Histoire de France” by Jacques Bainville). The same embeddings did not work well on French text, so I changed the spaCy model to French. After the change, the rest of the code could be executed as expected. Manual checking of the answers indicated that the quality of RAG is satisfactory. The script I ran for Lab 5 is uploaded along with this report.

Lab 6: Use cases and applications of LLMs

The improved script is uploaded along with this report. While it can find relevant information through queries, it cannot perform calculations correctly, such as summing up numbers or calculating the average or mean of several numbers.