CSC 510 1d1: Aadhir, Frank, Jackson, Sagnik

1. **What are the pain points in using LLMs?**

Sometimes the process of refining a prompt enough to get a usable result is more difficult than doing the work manually. For complicated/hard to explain short tasks it may make more sense to not use an LLM. Long, repetitive processes make the most sense to use an LLM for.

2. **Any surprises? Eg different conclusions from LLMs?**

We found that most LLMs gave similar results for our use cases. They may vary slightly in the content they produce, but we were able to tweak the prompts to get usable results from all of them. The most important aspect of using the LLMs is to develop a unique and detailed prompt for the given problem.

3. **What worked best?**

The way to get the best results from LLMs was to be very clear, specific, and structured with the prompts we give. By giving as much detail as possible to the LLM as well as providing explicit instructions, it allowed for the LLM to provide a more accurate answer. Additionally, we saw much better results when breaking down tasks into smaller tasks. For example, for the previous write-ups, instead of giving the LLM all the instructions at once for each part, we found that giving instructions one part at a time allowed for more detailed and accurate responses. Finally, providing the LLM with context and examples gave much more structured responses. For example, when asking for use cases, without giving the examples from the slides, the LLM formatted the cases in its own way, making it undesirable to use. However, after providing it with the exact structure from the slideshow, it was able to give much more accurate use cases that matched the structure we were looking for.

4. **What worked worst?**

By far the worst strategy was zero shot prompting. Asking the LLM for answers without giving any context gave very unpredictable and varied responses. As mentioned in the previous question, simply asking the LLM “Generate some use cases” vs. “Give me some use cases based on this information (insert background) in this format (insert format)” gave us much better results. Furthermore, prompting without further follow up gave undesirable results. Just taking the first result the LLM gave proved to be a very poor strategy, and we had much better results by giving follow up prompts to further refine the output

5. **What pre-post processing was useful for structuring the import prompts, then summarizing the output?**

During the pre-processing stage, we found that chunking the sources into pages and formatting the retrieved context with source tags allowed the LLM to have a deeper understanding of the material. During the post–processing stage, condensing the answers the model gave while keeping the essential information and mapping each answer to a source tag gave us consistent outputs that could be validated.

---

6. **Did you find any best/worst prompting strategies?**

The best prompting strategies gave the model a clear direction and a strict output structure to follow. Specifically, we gave the model a clear role that it was a procurement requirements analyst tasked with identifying constraints and use cases for a food delivery application. We then gave the model a specific task such as generating 5 use cases relating to the core functionality of the app and instructed the model to use a defined output schema with fields such as Preconditions, Main Flow, Subflows, and Alternative flows. Since the RAG system mapped chunks to source tags, we also instructed the model to directly cite tags used to generate use cases, allowing us to determine whether or not the use case generated was a result of a hallucination.

Another effective prompting strategy used when amplifying use cases was segmenting the task into smaller prompts instead of one-shot prompting. This allowed us to take advantage of the latent space in the model over the same context window. For example, we instructed the model to generate an extensive list of potential requirements for food delivery. Then, for each of these potential requirements, we instructed the model to expand these broader requirements into specific use cases, and required the model to cite source tags to ensure that the model linked its responses to one of the RAG chunks. In the prompt instructions, we also included that if no specific source supports a specific use case, then a blank source tag was acceptable. This ensured that outputs with a blank source tag were either hallucinations or something that was inferred by the model.

The worst prompting strategies were vague, generic prompts that lacked detail and failed to give the model a specific role or task. This resulted in outputs that lacked a specific structure and specific detail regarding the food industry. Since the model did not have a defined structure, this resulted in inconsistencies with the model's output. Additionally, prompts that gave the model too many chunks of information resulted in the model tending to repeat details it had already conveyed. In fact, a large number of chunks (each chunk greater than one page) resulted in the model focusing on more obscure, irrelevant details.