17-630 Prompt Engineering

Final Project

The final project is an opportunity for students to explore prompt engineering on their own in a multi-week assignment. Students may work independently or in a group. Students are encouraged to think about *prompt engineering-in-the-small*, which is the design of the prompt instruction, demonstrations (if any) and expected answer format; as well as, *prompt engineering-in-the-large*, which concerns the orchestration of multiple prompts to achieve a combined goal, and any environmental effects, such as data pre-processing or how the prompt responses would be used, on the reliability and accuracy of the overall solution. Overall, engineering concerns how we measure past performance and why we expect this performance to continue into the future, including how we handle errors and deviations in performance.

Learning Objectives

- Study the composition of multiple prompt strategies taught in class
- Design and evaluate advanced prompt-based systems

Assignment Deliverables

This is an individual or group assignment. Students should plan their work incrementally, beginning with the "least acceptable unit" of achievement to demonstrate their overall objective. The project proposal is an early means to collect feedback from the instructor and teaching assistants to help narrow the scope of the project toward a feasible deliverable.

Submit a single archive with the following files and exact filenames:

- *presentation.pptx* or *presentation.pdf* – an extended presentation that includes what you plan to present during the last week of class. This file can include more slides than what you plan to present to help explain your work. Please follow the guidelines below when preparing the presentation.
- *repository.txt* – a URL to a private repository (e.g., on GitHub, Google Drive, DropBox, etc.) containing your code used to collect data and develop examples presented in your presentation. The repository may be private but should be accessible by the instructor and teaching assistants. Document your codebase using the same language that you use in your presentation to help the graders trace the details between your presentation and code base.

Guidance on Presentation Format

The presentation should be planned for 10 minutes and have the following sections:

- **Problem Statement** – a brief description of the problem that you are trying to solve. You can illustrate the problem with an example, or you can report statistics or cite news sites and/or publications to show the "realism" of the problem (e.g., how many people experience this, what is the cost to individuals, companies or society, etc.)
- **Approach** – how did you approach the problem? What does the overall architecture of your solution look like? Did you combine prompt strategies together into a

workflow? What variations in your approach did you try (i.e., what worked and what did not work?)

- **Results and Recommendations** – what were you final conclusions? Did you define a metric for success (e.g., accuracy, or some other metric?) What are your recommendations based on your findings? Be positive where you have strong evidence of success, but also be cautious were the results were not consistently positive.
- **Next Steps** – briefly identify what you would try if you had an additional 3-6 months. This could be one slide.

Because you only have 10 minutes, try to keep your presentation down to around 10 slides. If you exceed 10 slides, practice your talk to ensure you do not exceed the time limit. You can include extra slides in your final submission to help the graders understand what you achieved. At 10 minutes into your presentation, the teaching assistants will cut you off, so be prepared!

Do not fill your slides with the text that you plan to say during your presentation. Instead, focus on developing examples that you can use to walk your classmates through your work. Include example prompts and responses, show overall statistics to illustrate both the positive and negative results. Ensure your recommendations are meaningful given what you observed. The presentation is intended to teach yourself and others what worked and what did not work.

Evaluation Criteria

- Clarity of your presentation and documentation of your code base. Document your code so that the instructor and teaching assistants can review your presentation and then examine your code to obtain a more detailed and holistic understanding of what you have achieved. Your code-level documentation can include your thoughts and reflections.
- Consistency between your recommendations and your results. The recommendations should logically follow from your results.