Test Task

Prompt Engineer Test Task: Comparing Small Language Models

Introduction

While large language models have garnered significant attention, smaller language models (SLMs) with 1-4 billion parameters have shown impressive capabilities for specific tasks. These compact models offer a balance between performance and efficiency, making them suitable for applications where resource constraints matter. For instance, SLMs could potentially enhance text editors with real-time writing suggestions, running smoothly on standard laptops without requiring powerful cloud servers. In this task, we'll explore the potential of these SLMs in another practical application: sentiment analysis of movie reviews.

Task Description

Your task is to compare two small language models for sentiment analysis of movie reviews:

- microsoft/Phi-3-mini-4k-instruct: A lightweight 3.8B parameters model trained on high-quality datasets with a focus on reasoning.
- Qwen/Qwen2-1.5B-Instruct: An even smaller model at 1.5B parameters, designed for efficient performance across various tasks.

For this task, we'll use the ajaykarthick/imdb-movie-reviews dataset. Note that this dataset contains 40,000 entries, which is too large for a complete local run. We expect you to choose a reasonable subset of the data for your analysis.

We expect you to perform local inference within a Jupyter notebook or Python script. Please do not use external services or tools like Ollama for this task. Here's what we want you to do:

1. Set up your local environment to load and run both models. Prepare the dataset for use.
2. Experiment with various prompt engineering techniques to design an effective solution for sentiment analysis.

3. Run your solution through **both models** using two different temperature settings: **0.2 and 0.8**.
4. Analyze the results you've gathered. Create visualizations to compare how the models performed. Discuss how the temperature setting affected the classification results.
5. Write up a summary of what you found. Include your thoughts on how well each model did, how you arrived at your final prompt, and any interesting patterns or insights you discovered during your experimentation.
6. Publish your code, research outcomes, and a `README.md` with instructions for running your solution locally to a GitHub repository. Share the repository link with us.

Time Expectation: This task is designed to be completed in approximately 4-6 hours. While there's no strict time limit, we encourage you to manage your time efficiently and focus on the key aspects of the task.

What We're Looking For

- The effectiveness of your prompt design and its impact on classification results
- Your approach to choosing a reasonable subset of the dataset and the rationale behind your choice
- The insights you were able to draw from the results
- The quality, clarity, and reproducibility of your code

Remember, we need to be able to run your solution on our local machines, so make sure to include any setup steps or list any dependencies we might need in your `README.md`.

We're more interested in seeing your approach to the problem and the insights you uncover than in achieving perfect results. As a point of interest, we've seen solutions achieve up to 94% accuracy, but don't let that number constrain your thinking. Your unique approach might uncover different insights or tradeoffs that we haven't considered. Feel free to explore and be creative with your solution. If you have any questions during the process, please don't hesitate to reach out.

Good luck!