COL772 A2 Spinoff: Table Question Answering with Large Language Models

1 Motivation

In this part, you will use Google's new Gemma model (gemma-7b-it-quant)[1] to solve the tabular QA task. Given a table and a question, you will first form a textual prompt. Gemma will then generate a response to your prompt. Finally, you will post-process the Gemma response to the output row and column of the answer.

The goal of this spin-off is to pitch a state-of-the-art language model against supervised models, as in the main Assignment 2. Let's see if you can beat your own supervised models with just prompt engineering.

2 Data

Feel free to use all available training data from the main Assignment 2 to develop the prompt. A simple strategy can be to sub-sample a reasonably sized validation set. Then, tweak the prompt such that it achieves good performance on the sampled validation set. **Note that no training or validation data will be available at the test time.**

3 Implementation Details

A starter script *prompt_gen.py* is given with the assignment. Update the implementations for the following methods in the *PromptGenerator* class.

1. Constructor (*__init__(self)*): This method must set generation parameters for Gemma. Parameters include output_len, temperature, and top_p. You can modify their values but not delete them. You may add any new variables as needed by your implementation.

2. Prompt creator (*create_prompt(self, sample)*): This method must create a text prompt for the input sample. The input sample contains **table** and **question** as in the case of main Assignment 2.

3. Post-processing (*post_process(self, gen_text)*): Gemma model generates a response to the prompt as text. post_process method must convert the answer from the generated model response to (row, column) format, as in the case of main Assignment 2.

Note: The driver code will perform only one call to Gemma for each input sample. You can refer to the Kaggle notebook given in the next section for a reference driver code.

[1] You can read about Gemma here.

Allowed Libraries:

- pandas for handling tabular data
- Any other standard Python library included in the default installation

In case you need to use additional packages, you can request for them on Piazza. They will be allowed subject to verification.

4 Using Kaggle for Experimentation

We recommend using Kaggle for running experiments related to Gemma. At first, you must apply for a Gemma License, which typically takes only a few minutes. Go to the Gemma Kaggle page and apply for the license by clicking "Request Access" button. Complete the small form to get access.

We have created a starter notebook that you may use.

5 Evaluation Criteria

Your model will be evaluated based on the subset (exact match) accuracy of row, column, and cell predictions individually, with the final score being the mean of these three accuracies. All evaluations will follow an all-or-nothing approach, rewarding only completely correct classifications for each question.

6 Submission Instructions

6.1 Deadline

The assignment must be submitted by **April 3, 2024, 11:59 PM**.

6.2 Submission Format

We will use Moodle for submission. You must submit the updated Python script *prompt_gen.py* that implements the following methods for *PromptGenerator* class.

Note: You don't need to make any calls to Gemma from the PromptGenerator class. Our driver code will take care of the rest.

7 Submission Guidelines

Your submission must adhere to the following instructions:

- The assignment is to be done individually.
- You should use **Python 3.10** for this assignment. Only aforementioned packages are allowed to be used. Any additional packages can be requested over Piazza and are only allowed after verification.
- Using any external source of data is prohibited.
- Your code must not perform any kind of supervised training. Importing libraries like PyTorch, Sklearn in *prompt_gen.py.* is strictly prohibited.

- Your code will be tested on Kaggle with P100 GPU. We will only use *gemma-7b-it-quant* V2 model.
- You must not discuss this assignment with anyone outside the class. Make sure you mention the names in your write-up in case you discuss with anyone from within the class. Please read academic integrity guidelines on the course home page and follow them carefully.
- We will run plagiarism detection software. Any person found guilty will be awarded a suitable penalty as per IIT rules.
- Your code will be automatically evaluated. You will get a 20% flat penalty if it does not conform to output guidelines.
- Your code must be your own. You are not allowed to use ChatGPT or any coding assistant for this assignment.
- We will run plagiarism detection on the model generated code, and any similarity will be penalized as per IIT rules.

8 Clarifications

For any doubts or clarifications, please use the Piazza forum. Ensure your code is well-commented, follows the submission guidelines, and is tested thoroughly to avoid any penalties related to non-conformance to output formats or runtime errors. It is your responsibility to seek clarification on any aspect of the assignment you find unclear before the deadline.