Prompting Take Home Assessment
April 16, 2025
==========Instructions==========

1. **Submission Format:** zip file with the name **<RollNo>_Prompt.zip** containing all the files.
2. You are not allowed to use any external libraries other than PyTorch, transformers, and required libraries for preprocessing along with scikit-learn, scipy, numpy, pandas etc.
3. Plagiarism will lead to **100% penalty** to all the involved parties. You are neither allowed to discuss among each other nor share code artifacts among each other.
4. **Please** make sure to follow the correct file requirements in terms of command line arguments and proper file naming conventions.

==========

This week we will learn in detail about several prompting techniques on LLMs. The LLM in focus are hosted on Groq a free-to-use website which host a series of LLMs with certain rate limits in place. We will now start by describing the task at hand.

Task for the assignment: SST2 by StanfordNLP

the action is stilted	0 negative
on all cylinders	1 positive
will find little of interest in this film , which is often preachy and poorly acted	0 negative
by far the worst movie of the year	0 negative

The Stanford Sentiment Treebank is a corpus with fully labeled parse trees that allows for a complete analysis of the compositional effects of sentiment in language. The corpus is based on the dataset introduced by Pang and Lee (2005) and consists of 11,855 single sentences extracted from movie reviews. It was parsed with the Stanford parser and includes a total of 215,154 unique phrases from those parse trees, each annotated by 3 human judges.

Binary classification experiments on full sentences (negative or somewhat negative vs somewhat positive or positive with neutral sentences discarded) refer to the dataset as SST-2 or SST binary.

Sample random **20 validation** set examples for evaluation (use train set only when using In-context Learning).

Models to consider:

a. `llama-3.1-8b-instant`
b. `Gemma2-9b-it`
c. `deepseek-r1-distill-llama-70b`
d. `qwen-qwq-32b`

Prompting strategies to consider:

a. *Zero-Shot Prompting*

In this technique, your input to the model will only be the sentence for which you want to obtain the sentiment.

…

```
messages=[
        # Set an optional system message. This sets the behavior of the
        # assistant and can be used to provide specific instructions for
        # how it should behave throughout the conversation.
        {
            "role": "system",
            "content": "You are a helpful assistant."
        },
        # Set a user message for the assistant to respond to.
        {
            "role": "user",
            "content": f"Given a phrase as input, please classify whether the
                        input sentence has positive or negative sentiment.
                        Output only Positive or Negative.

                        Sentence: {sentence}
                        Sentiment:",
        }
    ],
```

…

Note: You can change the instruction format if needed, this is just for sake of demonstration.

b. *Prompting —Task explanation*

In this technique, your input to the model will only be the sentence and two demonstrations specific to the task at hand sampled at random one specific to each category. Next, we append the test point.

…

```
messages=[
        # Set an optional system message. This sets the behavior of the
        # assistant and can be used to provide specific instructions for
        # how it should behave throughout the conversation.
        {
            "role": "system",
            "content": "You are a helpful assistant."
        },
        # Set a user message for the assistant to respond to.
        {
            "role": "user",
             "content": f"Given a phrase as input, please classify whether the
                        input sentence has positive or negative sentiment.
                        Please consider examples below to understand the
                        Sentiment category. Output only Positive or Negative.
```

```
                        Sentence: {Positive Category Sentence}
                        Sentiment: Positive

                        Sentence: {Negative Category Sentence}
                        Sentiment: Negative

                        Sentence: {sentence}
                        Sentiment:",
            }
        ],
...
```

Note: You can change the instruction format if needed, this is just for the sake of demonstration.

c. *In-Context Prompting*

In this technique, your input to the model will be the task explanation along with two examples similar to the test input.

In order to find the similar sentences, sample sentences from the train set using a sentence similarity procedure. We leave the design choice of procedure to you. You could set up tf-idf, BM25, sentence-transformers, BERT etc.

```
...
messages=[
        # Set an optional system message. This sets the behavior of the
        # assistant and can be used to provide specific instructions for
        # how it should behave throughout the conversation.
        {
            "role": "system",
            "content": "You are a helpful assistant."
        },
        # Set a user message for the assistant to respond to.
        {
            "role": "user",
             "content": f"Given a phrase as input, please classify whether the
                        input sentence has positive or negative sentiment.
                        Please consider the examples below to guide your
                        decision. Output only Positive or Negative.

                        Sentence-1: {Positive Category Sentence}
                        Sentiment-1: {Positive/Negative}

                        Sentence-2: {Positive Category Sentence}
                        Sentiment-2: {Positive/Negative}

                        Sentence: {sentence}
                        Sentiment:",
        }
    ],
...
```

Note: You can change the instruction format if needed, this is just for the sake of demonstration.

d. *Zero-Shot Chain-of-Thought Prompting*

In this technique, you will force the model to imitate a thought process that will help the model to predict the class more clearly. This is typically done by putting the phrase *'think step-by-step*' in the prompt.'

```
…
messages=[
        # Set an optional system message. This sets the behavior of the
        # assistant and can be used to provide specific instructions for
        # how it should behave throughout the conversation.
        {
            "role": "system",
            "content": "You are a helpful assistant."
        },
        # Set a user message for the assistant to respond to.
        {
            "role": "user",
             "content": f"Given a phrase as input, please classify whether the
                         input sentence has positive or negative sentiment.
                         Please think step-by-step while processing the
                         sentiment. Output only Positive or Negative.

                         Sentence: {sentence}
                         Sentiment:",
        }
    ],
…
```

Note: You can change the instruction format if needed, this is just for the sake of demonstration.

e. *Few-Shot Chain-of-Thought Prompting*

In this technique, you will force the model to imitate a thought process that will help the model to predict the class more clearly.

For the input sample you will do the following task:

1. Find two similar examples from the train set (you can use the ones already found in Task-c).
2. Create a short explanation of why you feel the sentence should belong to category (formally called as Chain-of-Thought)
3. Prompt the model with example and explanation in-context.

```
…
messages=[
        # Set an optional system message. This sets the behavior of the
        # assistant and can be used to provide specific instructions for
        # how it should behave throughout the conversation.
        {
            "role": "system",
            "content": "You are a helpful assistant."
        },
        # Set a user message for the assistant to respond to.
```

```
            {
                "role": "user",
                 "content": f"Given a phrase as input, please classify whether the
                             input sentence has positive or negative sentiment.
                             Please consider the examples below to guide your
                             decision. Please think step-by-step while processing
                             your decision. Output only Positive or Negative.

                             Sentence-1: {Positive Category Sentence}
                             Explanation: {explanation}
                             Sentiment-1: {Positive/Negative}

                             Sentence-2: {Positive Category Sentence}
                             Explanation: {Explanation}
                             Sentiment-2: {Positive/Negative}

                             Sentence: {sentence}
                             Sentiment:",
            }
        ],
    …
```

Note: You can change the instruction format if needed, this is just for the sake of demonstration.

Deliverables

1. 1 .ipynb file sections corresponding to the tasks and prompting strategy.
2. 1 .md file containing:
    a. Prompt structure you wrote specific to each prompt strategy —clearly marking the system prompt and user prompt.
    b. Evaluation report: classification accuracy, precision, recall, f1 score for each model using each of the prompting strategies.

Format:

<Roll_No>_Prompts.zip

|- <Roll_No>.ipynb

|- <Roll_No>.md