TASK 2 (PROMPTING TECHNIQUES / RAG)

0.1 Problem Statement

0.1.1 Dataset: train_40k .csv

Design and implement a classifier using any LLM to classify the data in Column name "Text" with Column name "Cat2" and Column name "Cat3".

- Input to your prompt will be a text from column name Text, and output should be class name from Column Name Cat 2
- Input to your prompt will be a text from column name Text, and output should be class name from Column Name Cat 3

Report the Accuracy on a sample test set split from 40k samples.

SOLUTION

1 DATASET PREPARATION

In addressing this classification problem and aiming to construct a classifier using a Large Language Model (LLM), the data must be formatted in a specific manner for fine-tuning. In this case, I am opting to prepare the data in Alpaca format.The dataset has been segregated into inference data and training data. The initial 1000 datapoints are designated as inference data, while the remaining datapoints are allocated for training data.

1.1 Input to your prompt will be a text from column name Text, and output should be class name from Column Name Cat 2

Following steps are performed for data preparation.

- Taking column Text and Cat2 from the dataset.
- Creating a column called Instruction and add the prompt :**Classify the text into categories given in output column. Reply with only the words given in output column.**
- Rename the columns Text and Cat2 as input and output.
- Finally create a column called **text** and add the content as below:

  ### InstructionInstruction.### Input:input+### Output:output

  For more details ,please refer the notebook :**training_data_preparation2 .ipynb**

1.2 Input to your prompt will be a text from column name Text, and output should be class name from Column Name Cat 3

Following steps are performed for data preparation.

- Taking column Text and Cat2 from the dataset.
- Creating a column called Instruction and add the prompt :**Classify the text into categories given in output column. Reply with only the words given in output column.**
- Rename the columns Text and Cat3 as input and output.
- Finally create a column called **text** and add the content as below:

  ### InstructionInstruction.### Input:input+### Output:output

  For more details ,please refer the notebook :**training_data_preparation2 .ipynb**

2 MODEL FINE TUNING

For our task, we're employing the Llama 2 model. Given that Llama 2 lacks specific knowledge about our data domain, we plan to enhance its performance by fine-tuning the model. This approach aims to yield improved results tailored to our specific domain.

2.1 Input to your prompt will be a text from column name Text, and output should be class name from Column Name Cat 2

Following steps are used for fine tuning the Model:

- Load the data which is prepared from data preparation step.
- Load the model Llama 2 from hugging face(Note: We require Access tokens from hugging face to access this model)
- Load the tokenizer
- We are going to use PEFT technique for fine tuning the model to save the computation and cost.
- Finally training has been started .
- I have pushed the fine tuned model to hugging face hub : Model

2.2 Input to your prompt will be a text from column name Text, and output should be class name from Column Name Cat 3

- Load the data which is prepared from data preparation step.
- Load the model Llama 2 from hugging face(Note: We require Access tokens from hugging face to access this model)
- Load the tokenizer
- We are going to use PEFT technique for fine tuning the model to save the computation and cost.
- Finally training has been started .
- I have pushed the fine tuned model to hugging face hub : Model

3 EXPERIMENT WITH FINE TUNED MODEL AND PROMPTING TECHNIQUES

We utilized Langchain along with our fine-tuned model to build the classifier.

3.1 Input to your prompt will be a text from column name Text, and output should be class name from Column Name Cat 2

Following steps are performed to build the system:

- Load the inference data which was done in data preparation step
- Import PromptTemplate, LLMChain from Langchain
- Create the list of unique categories from the column Cat 2 from the inference data.
- Develop a prompt
- I have followed few short prompting for the purpose our task.
- For more details : Please refer the Notebook : **training_cat2_llama2.ipynb**

3.2 Input to your prompt will be a text from column name Text, and output should be class name from Column Name Cat 3

Following steps are performed to build the system:

- Load the inference data which was done in data preparation step
- Import PromptTemplate, LLMChain from Langchain
- Create the list of unique categories from the column Cat 3 from the inference data.
- Develop a prompt
- I have followed few short prompting for the purpose our task.
- For more details : Please refer the Notebook : **training_cat3_llama2.ipynb**

4 RESULT & METRICS

4.1 Input to your prompt will be a text from column name Text, and output should be class name from Column Name Cat 2

For simplicity, I have selected 100 data points for inference, and within this sample, there are 24 categories.

```
[18] inference_data_cat2['Cat2'].unique()

array(['meat poultry', 'games', 'puzzles', 'beverages', 'makeup',
       'arts crafts', 'action toy figures', 'dolls accessories',
       'baby toddler toys', 'personal care', 'nutrition wellness',
       'learning education', 'electronics for kids', 'household supplies',
       'stuffed animals plush', 'tricycles', 'health care', 'gear',
       'skin care', 'grown up toys', 'dress up pretend play',
       'novelty gag toys', 'bath body', 'tools accessories'], dtype=object)
```

Figure 1. Inference Categories for Cat 2

Out of 100 records , the model is able to predict 46 records correctly .
Note : The inference result can be found in csv file entitled : **inference_data_cat2_with_accuracy.csv**

4.2 Input to your prompt will be a text from column name Text, and output should be class name from Column Name Cat 3

For simplicity, I have selected 100 data points for inference, and within this sample, there are 34 categories.

```
[95] finalCategory_three['Cat3'].unique()

array(['jerky', 'unknown', 'jigsaw puzzles', 'board games', 'juices',
       'nails', 'drawing painting supplies', 'figures', 'dolls',
       'card games', 'drawing sketching tablets', 'shape sorters',
       'deodorants antiperspirants', 'nutrition bars drinks', 'habitats',
       'household batteries', 'push pull toys', 'scooters wagons',
       'clay dough', 'allergy', 'baby gyms playmats',
       'shaving hair removal', 'face', 'animals figures', 'feminine care',
       'music sound', 'oral hygiene', 'pretend play', 'cleansers',
       'playsets', 'd puzzles', 'dollhouses', 'lip care products',
       'nail tools'], dtype=object)
```

Figure 2. Inference Categories for Cat 3

Out of 100 records , the model is able to predict 14 records correctly .
Note : The inference result can be found in csv file entitled : **inference_data_cat3_with_accuracy.csv**

5 IMPROVEMENT SUGGESTIONS

- The model is demonstrating satisfactory performance for the category Cat2 .
- Enhancing the results is possible by incorporating diverse sample categories and experimenting with various prompts.
- Fine-tuning with specific prompts has the potential to enhance the results for cat3.
- Implementing the RAG technique could potentially enhance the outcomes for both cat2 and cat3.

MY PORTFOLIO

Please feel free to explore my projects:

- Github