Title: Programming Task for RA Position Interview
Student Name: Siva Yellepeddi

Introduction

The goal of this exercise is to identify the existence of a few dimensions in letters written by CEOs to shareholders using OpenAI's language model. The language model is accessed using OpenAI's API. The model is first fine-tuned using two training datasets of the three training datasets provided with the leftover dataset used as a validation set. The accuracy of the model is assessed using the in-sample vs out-of-sample data. For the final predictions step of the test set provided, another fine-tuned model is used where all the three training datasets are used to fine-tune this model. This fine-tuned model is used on the test set provided to populate the eight attributes for the test set.

Methodology

The step-by-step process of starting from prepping the data to getting the final output is listed below:

- **Load and Preprocess the Data:** We are using the provided train2 and train3 datasets as the *training* set and the train1 dataset as the *validation* set. In this step, we concatenated the train2 and train3 datasets to create a single *training* set.
- **Prepare Data for Fine-tuning:** In this step, we convert the data from the table format (.xlsx) provided to a Chat Completions API[1] format that is accepted by OpenAI's API. In its essence, it is a conversational format with a list of messages where each message has a role and content. In our *training* set, there will be three components for each datapoint with each playing a different *role*. The first part of the message plays the *role* of 'system'. This is where we give precise instructions to the model as to what we expect it to do. The next part of the message plays the *role* of 'user' which contains portions of the letters from CEOs. This is supposed to be the input for fine-tuning process. The last part of the message plays the *role* of 'assistant', which is the result from which we want it to fine-tune.

  Example:

  {'messages': [{'role': 'system', 'content': "Use the following step-by-step instruction to respond to the user inputs. Step 1 - In the user content which is taken from letters written by CEO to shareholders, you have to identify the existence of dimensions/qualities that are provided in this list given in brackets and that are separated by commas ['Goal', 'Activity', 'Strategy', 'Plan', 'Structure', 'Innovation', 'Tactics', 'Relevance']. Step 2 - For each of these dimensions, if the dimension exists in the user prompt based on the assistant content I provide to you in the fine-tuning data, answer Yes, otherwise answer No. After the step 2, this is an example output whose template you must use to provide your answer - ['Goal: No, Activity: Yes, Strategy: Yes, Plan: Yes, Structure: Yes, Innovation: Yes, Tactics: No, Relevance: No']"},
  {'role': 'user', 'content': 'february 24, 2011\nto our shareholders:\n2010 was another challenging year for sears holdings. our financial results .... sears holdings into a truly integrated retail company, focusing on customers first'},
  {'role': 'assistant', 'content': 'Goal: No, Activity: Yes, Strategy: Yes, Plan: Yes, Structure: Yes, Innovation: Yes, Tactics: No, Relevance: No'}]}
- **Fine-tune the Model:** We invoke an OpenAI model, feed the *training* data, and finetune it. The base model used for fine-tuning is *gpt-3.5-turbo*.
- **Make Predictions:** We now use the fine-tuned model and feed in the *validation* set and also the test dataset provided. We will use the results from the *validation* set to assess the model's performance and use the results from the test set to populate the test dataset provided.

[1] Chat Completions API format, https://platform.openai.com/docs/guides/fine-tuning/preparing-your-dataset (accessed November 18, 2023).

- **Evaluate Performance of the Fine-tuned Model:** We assess the performance of the fine-tuned model by comparing the results from the predictions step to the actual values provided in the *validation* set (out-of-sample) as well as the *training* set (in-sample).

Results

Testing the model for its accuracy in prediction

To estimate how effective the fine-tuned gpt 3.5 turbo model is, I trained the model on *training* set (train2 & train3 datasets) and tested it on the *validation* (train1) dataset. The results are tabulated below (Table 1) and are showcased in a graphical format (Figure 1). In Table 1, the In-sample Accuracy column corresponds to the attributes prediction made on the same *training* set that was used to fine-tune the model and so the model has already seen the in-sample data during the fine-tuning process. So, the in-sample results are expected to be better than the out-of-sample results.

Attributes	***In-sample Accuracy (Training)***	***Out-of-sample Accuracy (Validation)***
Goal	0.9	0.833
Activity	0.8	0.75
Strategy	0.8	0.75
Plan	0.9	0.75
Structure	0.9	0.75
Innovation	0.9	0.75
Tactics	0.8	0.333
Relevance	0.3	0.333
Total (Average)	**0.7875**	**0.6666**

Table 1: In-sample vs Out-of-sample results

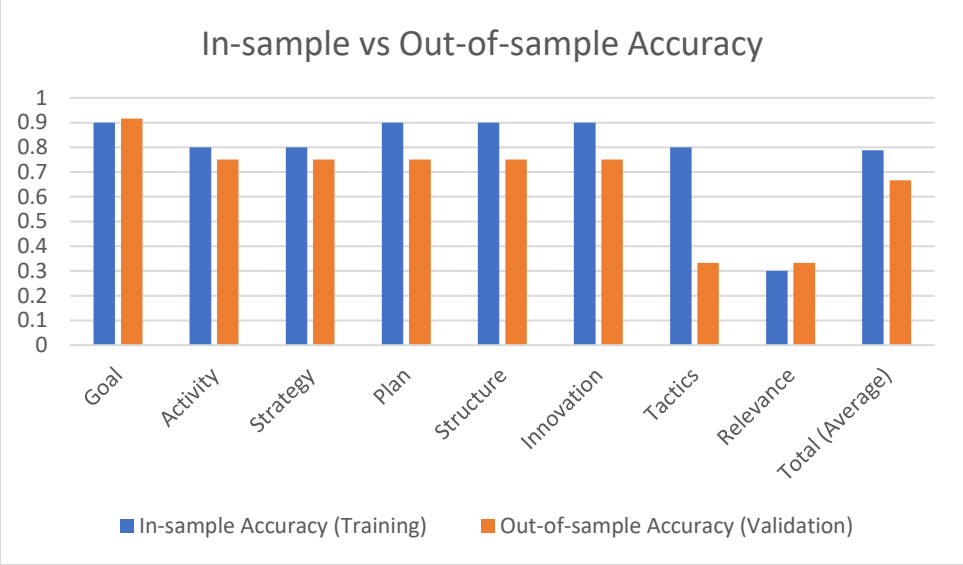

Figure 1: In-sample vs Out-of-sample results

Title: Programming Task for RA Position Interview
Student Name: Siva Yellepeddi

Predicting the Values of the Eight Dimensions of Test Set

To predict the values of the eight dimensions (Goal, Activity, Strategy, Plan, Structure, Innovation, Tactics and Relevance) I used the same gpt 3.5 turbo model. But this time, I trained it on all three train datasets (train1, train2, & train3). Based on this fine-tuned gpt 3.5 turbo model, I predicted the dimension values for the test dataset.

The predictions for the test dataset provided are populated. A preview of the results is shown for the first 5 data points in the test set (Table 2):

	paragraph	Goal	Activity	Strategy	Plan	Structure	Innovation	Tactics	Relevance
0	Chairman's LetterFebruary 26, 2015To our	No	Yes	Yes	Yes	Yes	Yes	No	No
1	For Sears and Kmart, after years of work a	No	Yes	Yes	Yes	No	Yes	No	No
2	This isn't new for Sears. An article in the O	No	No	No	No	No	No	Yes	No
3	Time and again, people have proclaimed o	No	No	No	No	No	No	No	No
4	These old stories got it partially right. Had	No	Yes	Yes	No	No	Yes	No	No
5	Without the aggressive steps we have alre	No	Yes	Yes	Yes	Yes	Yes	No	No

Table 2: Preview of Test Results

Accuracy for Test Data

Based on the True values provided and the predicted values by GPT 3.5 turbo model for the test data the accuracy for the model is shown in Table 3.

Attributes	***Accuracy***	
Goal	0.846	
Activity	0.692	
Strategy	0.487	
Plan	0.513	
Structure	0.77	
Innovation	0.77	
Tactics	0.72	
Relevance	0.28	
Total (Average)	**0.64**	

Table 3: Test data results

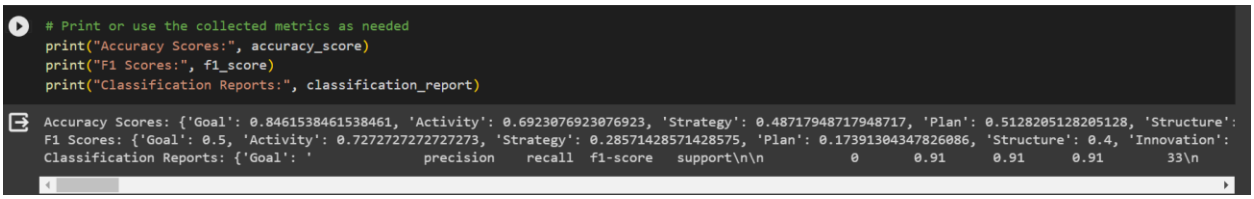

Figure 2: Accuracy results based on True values and Predicted values of Test data

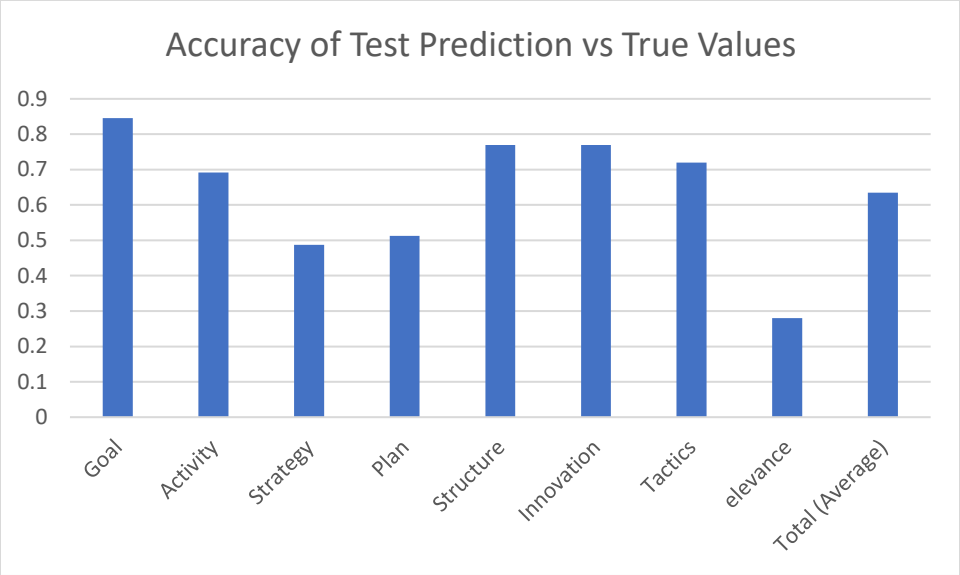

Figure 3: Test data results

Scope for Improvement

The results above showcase the extent to which we could fine-tune the model with the given datasets. There is scope for improvement and below are some pointers on how to do that:

- The fine-tuning process of the language model works best when we can map the respective components of the paragraph that led to the 'yes' or 'no' for each of the provided attributes. Not having this information will require the model to take a leap of judgment and this could bring in some ambiguity leading to poor performance. For example, for a specific paragraph, if the 'Goal' attribute is 'Yes', then adding the main keywords or phrases that led to 'Goal' being deemed as 'Yes' as another data point will make the model better fine-tuned. For example, if the keywords that make 'Goal' a 'Yes' are the existence of the words 'determined', 'catalyze', and 'transformation' in the paragraph, then adding these three keywords as another column in the training data will lead to better results.
- There could be some merit in assessing an alternative language model, such as Google's BERT, for our use case. A similar study can reveal how well a fine-tuned version of that language model is performing compared to the fine-tuned OpenAI's model.
- Lastly, having more training data can aid in making the fine-tuned model better.[2]

References

OpenAI. (n.d.). *API Reference*. Retrieved from OpenAI: https://platform.openai.com/docs/api-reference

[2] Best practices for fine-tuning GPT-3 to classify text, https://docs.google.com/document/d/1rqj7dkuvl7Byd5KQPUJRxc19BJt8wo0yHNwK84KfU3Q/edit (accessed November 18, 2023).