# Introduction

The task aimed to identify dimensions within CEO letters to shareholders using OpenAI's language model, leveraging fine-tuning on provided datasets for prediction purposes.

# Methodology

**Data Loading and Preprocessing**:

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]](#footnote-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.
* **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

The results are tabulated below and also showcased in a graphical format (Table 1) (Graph 1). Here, the in-sample values correspond to the predictions made on the *training* set which was used to fine-tune the model in the first place. The in-sample results are expected to be better than the out-of-sample results, as the model has already seen the in-sample data during the fine-tuning process.

|  |  |  |
| --- | --- | --- |
| Attributes | In-sample Accuracy (Training) | Out-of-sample Accuracy (Validation) |
| Goal | 0.9 | 0.916 |
| 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**

**Graph 1: In-sample vs Out-of-sample results**

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):



**Table 2: Preview of Test 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 lead to ‘Goal’ being deemed as ‘Yes’ as another data point will definitely 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.

# References

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

Introduction

The task aimed to identify dimensions within CEO letters to shareholders using OpenAI's language model, leveraging fine-tuning on provided datasets for prediction purposes.

Methodology

1. **Data Loading and Preprocessing**:
   * Concatenated **train2** and **train3** as the training set, utilizing **train1** for validation.
   * Data preprocessing involved lowercasing text and formatting into Chat Completions API format for OpenAI's API.
2. **Fine-tuning the Model**:
   * Utilized the **gpt-3.5-turbo** model for fine-tuning.
3. **Making Predictions**:
   * Employed the fine-tuned model on both validation and test datasets.
4. **Performance Evaluation**:
   * Assessed model performance comparing predictions to actual values from the validation set.

Results

* Tabulated in-sample vs out-of-sample accuracy values for the dimensions.
* Demonstrated results in both tabular and graphical formats.

Test Set Predictions

* Provided a preview of predictions for the test dataset.

Scope for Improvement

Outlined potential enhancements:

* Data enrichment with specific keywords or phrases leading to 'Yes' or 'No' attributes.
* Exploring alternative models like Google's BERT.
* Expanding the training data to enhance the fine-tuned model's performance.

References

Cited OpenAI's API Reference for further details.

Look inside a current textbook on software architecture, and you’ll find few patterns that we don’t apply at

Amazon. We use high-performance transactions systems, complex rendering and object caching, workflow and

queuing systems, business intelligence and data analytics, machine learning and pattern recognition, neural

networks and probabilistic decision making, and a wide variety of other techniques. And while many of our

systems are based on the latest in computer science research, this often hasn’t been sufficient: our architects and

engineers have had to advance research in directions that no academic had yet taken. Many of the problems we

face have no textbook solutions, and so we -- happily -- invent new approaches

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