A Comparative Approach Using Base vs Fine-Tuned GPT-3.5 Turbo For Textual Based Sarcasm Detection

Anthony Erhire Awobasivwe

URN : 6442385

Declaration of Originality

I declare that this thesis and the research it encompasses are the outcome of my individual endeavours. Any concepts, data, images, or text derived from the work of others, whether previously published or unpublished, are explicitly acknowledged within this thesis and credited to their original sources in the text, bibliography, or footnotes. This thesis has not been presented, either wholly or partially, for any other academic qualification or professional certification. I acknowledge and consent that the University retains the right to utilise the plagiarism detection service Turnitin UK to authenticate the originality of my work. Irrespective of whether preliminary drafts have undergone such evaluations, the University reserves the right to request an electronic version of the final document (as submitted) for assessment as described above.

Anthony Erhire Awobasivwe

Acknowledgements

I would like to thank my supervisor Dr Diptesh Kanojia. Through scheduled meetings, he provided me key guidance from selecting a thesis topic to guiding me to carry out the project through his expert knowledge. I also thank my fellow classmates who encouraged me in my down times during our university stay. Lastly and most importantly I would like to thank my parents who have stood by me and provided me with the financial and emotional support to keep going throughout my education and all the way to the completion of my master's degree.

Abstract

Detecting sarcasm in a text-based setting is a persistently challenging task. Identifying sarcasm solely through written text adds an extra layer of difficulty since it lacks the vocal tone, facial expressions, and other nonverbal cues that aid in sarcasm detection during face-to-face conversations. As a result, recognizing sarcasm in text often requires a deeper understanding of the context, semantics, and subtle linguistic patterns, which can be highly nuanced and context dependent. Researchers are continuously working to improve sarcasm detection algorithms, leveraging techniques from natural language processing, machine learning, and sentiment analysis to enhance the accuracy of automated sarcasm detection systems. However, despite advancements, the task remains an ongoing and complex challenge due to the limited understanding of what constitutes irony in a sentence. This project investigates different GPT models provided by OpenAI capable of detecting sarcasm. This dissertation presents a novel approach to sarcasm detection utilising GPT-3.5 turbo to leverage techniques such as zero-shot and few-shot learning, combined with prompt tuning. The performance of the model will be evaluated across four unique datasets containing a range of sarcastic and non-sarcastic text samples. Different evaluation metrics such as accuracy, precision, recall, and F1-score will be used to determine the effectiveness of each approach. This dissertation contributes to the advancement of sarcasm detection by leveraging the immense language understanding capabilities of OpenAI GPT models combined with techniques such zero-shot, few-shot learning, fine-tuning and prompt tuning, to better enable the model to discern sarcasm in diverse contexts and outperform existing methods. Furthermore, the recent inauguration of fine-tuning to the GPT-3.5 turbo model presents a milestone not just for sarcasm detection but for other natural language tasks.

Table of Contents

Table of Figures

List of Abbreviations

Abbreviation	Word	Abbreviation	Word
NLP	Natural Language Processing	ML	Machine Learning
ZSL	Zero-Shot Learning	GPT	Generative Pre-trained Transformer
FSL	Few-Shot Learning	LLM	Large Language Model
ICL	In-Context Learning	API	Application Programming Interface

Project Structure

This thesis is divided into 6 chapters. Chapter 1 serves as an introduction to the presented topic and provides the background of the project as well as setting the motivation behind the project and defining the objectives, hypotheses, and scope of the project. Building on Chapter 1, Chapter 2 reviews different approaches used in the past to detect sarcasm in textual data as well as introducing our main focus of OpenAI GPT models to provide the reader with the requisite understanding for subsequent chapters. Chapter 3 presents the proposed model to be used for experimentation and the approach used for prompt engineering. This chapter also provides an overview of the experimentation setups for zero-shot learning, few-shot learning, and fine-tuning. Chapter 4 showcases the implementation of the project including steps taken for pre-processing, fine-tuning, and model building along with some supporting code. Chapter 5 presents the evaluation techniques to be used as well as the results of the experiments conducted and presents high-level findings from said experiments. Chapter 6 reflects on the results of the experimentations and gives a critical overview of the project objectives in relation to the derived results. The chapter concludes by outlining potential avenues for future research and improvements on this thesis.

Chapter 1 - Introduction

1.1 - Project Background

Sarcasm originating from the Greek word "sarkasmós", is a form of irony which occurs when there is a discrepancy between the literal meaning of an utterance and its intended meaning (Oprea & Magdy, ACL 2019). This can occur in both textual and multimodal forms. In textual form, sarcasm is usually conveyed through means such as the use of capital letters, exclamatory marks, etc. This disparity between the literal meaning and intended meaning of a word has posed a significant challenge to both humans and more so to machines (Bing Liu, 2010). This is mainly due to sarcasm enabling the speaker or writer to conceal their true intention of contempt and negativity under the guise of overt positive representation (Yaghoobian Hamed et al., 2021). Several different methods have been introduced to solve this problem which range from machine learning approaches to the more recent use of large language models such as OpenAI GPT-3 and its successors. These large language models have demonstrated the ability to solve NLP tasks such as sentiment analysis and next-sentence prediction (Ehsan Hosseini-Asl et al., 2022).

1.2 – Motivation

With the continuous growth of artificial intelligence and the emphasis on making machine learning models capable of mirroring human cognition and understanding, drawing predictions from little to no data has become a more popular theme in natural language tasks. Methods such as zero-shot and few-shot learning introduce a pathway for models to mirror this human-like adaptability, allowing them to generalize across tasks and scenarios with minimal prior exposure. Historically, machine learning and NLP models have been heavily reliant on vast, labelled datasets for training. In the case of sarcasm detection, this would mean having a considerable collection of text data manually labelled as 'sarcastic' or 'non-sarcastic'. With the rise of large language models such as OpenAI GPT models, techniques such as zero-shot and few-shot learning introduce a new pathway for making classifications within the paradigm. Drawing similarities from human cognition, these learning methodologies enable models to

make informed decisions based on extremely limited data or any example data provided. This closely resembles how human beings often make inferences, and decisive conclusions without the need for a plethora of examples to learn from. However, the OpenAI models still struggle in both zero and few-shot settings for tasks that heavily rely on context and inherent subtleties such as sarcasm detection. The recent introduction of finetuning for GPT-3.5 turbo opens a potential new pathway to sarcasm detection using zero-shot and few-show learning. By leveraging domain-specific data for fine-tuning paired with the huge knowledge base of the OpenAI model, the model now has the potential ability to be tailored to specific data to better recognize the nuanced cues and contextual indicators of sarcasm. This suggests a more effective approach to sarcasm detection, by fine-tuning models to better address the inherent challenges with sarcasm detection i.e., the deep-rooted and context-sensitive nature of textual-based sarcasm.

1.3 Project Objectives

The objectives of the project are separated into 3 sections. Each objective will be critically reviewed after the conclusion of all experimentations with each objective having its respective hypothesis.

Objective 1

The first objective aims to critically evaluate the performance of the chosen model (GPT-3.5 turbo) for two different scenarios (zero-shot and few-shot learning (10, 20, 30, 50 examples provided) across four datasets containing different types of textual data.

From objective 1, an initial hypothesis is made which states an increase in the number of examples provided in each prompt to lead to better performance. This hypothesis can be broken down into two parts:

1. Few-shot learning to outperform the zero-shot learning.

2. Few-shot setting with more prompt examples provided to outperform few-shot setting with fewer prompt examples provided.

Objective 2

The second objective builds on objective 1 by critically evaluating the performance of the base GPT-3.5 turbo model for each dataset compared with a fine-tuned version of the model on each dataset respectively (350 examples) across the same scenarios outlined in objective 1.

For objective 2, we hypothesise the fine-tuned version of the GPT-3.5 model to outperform the base GPT-3.5 model.

Objective 3

The third objective builds on objective 2 by critically evaluating the performance of the fine-tuned GPT-3.5 turbo model with 350 examples to a much larger fine-tune sample of 2000 example points across the same scenarios outlined in objective 1.

For objective 3, we hypothesise increasing the amount of data examples to be used for fine-tuning each model would lead to an increase in model performance.

Chapter 2 - Literature Review

2.1 – History

Sarcasm detection has been a topic extensively explored by NLP researchers. Over the years, several different methods have been proposed to address this challenge. Some of the earliest attempts at sarcasm detection by machines involved rule-based approaches where specific patterns or linguistic cues were manually defined to identify sarcastic statements (Joshi A et el., 2015). Early on, the availability of annotated data for training and evaluation was a significant challenge, hence researchers relied on small manually annotated datasets. As a result, early models struggled to generalise well to unseen data. As research progressed, sarcasm detection increasingly gained prominence through the application of machine learning techniques. This involved training classifiers on large annotated datasets, where human annotators labelled instances of sarcasm. The classifier then learnt patterns and features indicative of sarcastic language (Abu Farha et al., Findings 2022). Traditional machine learning methods however struggled to grasp the full context in the text as they couldn't handle sequential data effectively. With the emergence of deep neural networks such as RNNs and CNNs, researchers were now able to capture the sequential aspects of language and extract contextual information to enhance sarcasm detection capabilities. However, these deep neural networks still had certain limitations with performance bottlenecks for longer sequences of data (Stefanos Tsimenidis, 2020). With the release of the paper "Attention is all you need" in 2017, this changed the landscape for both sarcasm detection and other NLP tasks. The attention mechanism transformer model proposed in the paper, allowed the model to focus on different parts of an input sequence when producing an output sequence, leading to better model performance and making the transformer more adept at handling long-range dependencies. Building on the success of transformers, pre-trained large language models such as Open AI's series of GPT models have become a mainstay for modern natural language processing tasks including sarcasm detection. While these LLMs for the most part have the same architecture as the original transformer model, they differ in scale having been trained on huge amounts of data sourced from the web, giving these models the ability to grasp intricate linguistic

structures, discern subtle cultural references, better understand context and in general provide a vast expanse of knowledge into coherent and contextually aware responses (Dilmegani C, 2021). Furthermore, these LLM'S can be fine-tuned for specific tasks such as sarcasm detection enhancing their performance compared to previous approaches.

Below, we will further discuss in more detail various approaches to sarcasm detection that have been developed over the years.

2.2 Approaches To Sarcasm Detection

2.2.1 - Rule Based Approach

Rule-based methods for sarcasm detection involve the use of distinct pieces of evidence which can be expressed as rules. The use of hashtags, punctuations and capitalization are common methods which are used to identify sarcasm in text. (Diana Maynard and Mark Greenwood, 2014) highlighted sentiment which is conveyed through hashtags as a potential means of detecting sarcasm. By recognizing the use of hashtags by tweet authors when conveying sarcasm, they came up with a methodology for a rule-based detection system which primarily involved tokenizing concatenated words within hashtags to analyse their sentiment. This was then compared to the content in the rest of the tweet and sarcasm was identified when a conflict between the two cases occurred. While their presented approach had certain limitations, particularly in the depth of discourse analysis for effectively handling negation, sarcasm, and inherent opinions, it still provided some interesting observations. (Bharti et al., 2015) present another approach to rule-based sarcasm detection. They proposed a two-way approach, which initially employs a parsing algorithm to identify sentiment-bearing situations within the text and discern sarcasm based on contradictions of a negative (or positive) sentiment and a positive (or negative) situation. Subsequently, the approach leverages linguistic cues, like hyperbolic interjections such as "wow" or "yay", and intensifiers such as "absolutely" or "huge", to detect sarcasm in the text. Although they were able to draw some conclusions from their work such as tweets beginning with interjections and immediately followed by either an adjective or adverb tending to denote sarcasm, this approach still had its limitations as the reliance on

explicit contradictions and specific linguistic cues could potentially overlook subtler forms of sarcasm deeply rooted in context.

2.2.2 – Classical Machine Learning Based Approach

Classical learning-based methods can be categorized into three primary approaches namely supervised, semi-supervised and unsupervised learning. Although these approaches have differing methodologies, they typically involve the use of large datasets for training with the training data needing to undergo pre-processing steps through methods such as stemming, lemmatization, tokenization, and stop word removal to improve both quality and usability.

Supervised Learning

Supervised learning is a subfield of machine learning that utilizes labelled datasets for training models, enabling them to make predictions and extract insights from the data. In the context of sarcasm detection, this involves classifying sarcastic text through a model trained on textual data typically sourced from platforms such as Twitter and Reddit and yielding corresponding predictions. The dataset usually consists of human-annotated labels indicating if a given text is sarcastic or non-sarcastic. . This approach typically encompasses the use of methods such as support vector machines, instance-based learning, ensemble-learning , tree-based learning, and more recently deep learning approaches such as recurrent and convolutional neural networks. A unique approach using support vector machines for sarcasm detection was proposed by (Gupta & Yang, 2017). In their work, they trained an SVM classifier for detecting sarcastic expressions from tweets. To do this they developed a tool called crystalnet capable of analysing features by combining the sarcasm score derived, sentiment scores, word embedding vectors, and part-of-speech features. Although these models performed relatively well, overall performance was still not ideal due to these classical machine learning models for the most part lacking the ability to effectively extract intricate patterns, sequences, and representations from the data, a problem deep learning models such as Convolutional and Recurrent Neural Networks don't possess due to their ability to identify sequence patterns in data. Several deep learning approaches have been proposed for the problem of sarcasm detection with the view of improving the performance of classical machine learning models with the most prominent

being the use of convolutional neural networks (CNNs) and long short-term memory neural networks (LSTMs). These deep learning models possess multiple layers, allowing them to extract more intricate information from the data. This is readily seen in the approach proposed by (Ali, R. et al., 2023). In their work, they integrated a GlobalMaxPooling layer into the original LSTM model in order to capture higher-level patterns while handling longer data sequences more effectively. The model was evaluated on two distinct datasets and showed improved performance compared to the normal LSTM model and non-deep learning models. It is however likely the model proposed was overfitting to the data given the extremely high accuracies achieved (0.999 and 0.925 for the train and test set respectively).

Unsupervised Learning

Unsupervised learning is a subfield of machine learning where patterns are inferred from the data without the use of labels. This is usually done through methods such as anomaly detection, clustering, density estimation, and association rule learning. In the context of sarcasm detection, this would mean classifying sarcastic data without any provided labels. This is particularly challenging given sarcasm heavily depends on context; meaning two different phrases could potentially be interpreted as sarcastic based on their surrounding content. Although still in its infancy, Researchers have used unsupervised learning to solve different types of NLP tasks closely related to sarcasm detection such as text classification, hate speech classification, sentiment analysis etc. Of the relatively few approaches to sarcasm detection via unsupervised learning, one interesting framework which stood out was by (Nozza, Debora et al., 2016). They propose an unsupervised framework for domain-independent irony detection. They did this by building upon an existing probabilistic topic model previously used for sentiment analysis to identify the hidden semantic structures in large archives of ironic texts paired with word embeddings to obtain domain-aware ironic orientation of words. The model performance yielded from their work performed significantly better than existing models within the domain. The proposed model however, had some limitations concerning its assumptions on word independence with latent relationships currently not existing among different sentences.

Semi-Supervised Learning

Semi-supervised learning combines elements of both supervised and unsupervised learning often used when there is a limited amount of labelled data. The model is trained on a few labelled samples and then on large amounts of unlabelled data. As such it is often a popular choice among researchers due to the high cost involved and time consumed with manually annotating labels for full supervised learning. Different types of semi-supervised learning methods such as self-training, co-training, and multi-view learning have been used to solve natural language tasks including sarcasm detection with each giving varying degrees of success and effectiveness. One of which was by proposed by (Davidov et al., 2010). Making use of large samples of unlabelled data gotten online from Amazon reviews and tweets from twitter, they proposed a semi-supervised technique using pattern-based recognition and punctuation-based features for model building. The proposed model showed good results across all metrics even in the context of cross-domain training and exhibited proficiency without the need for domain adaptation. This approach however required large proportions of data for training in order to be effective.

2.2.3 Transformer Based Approach

The transformer model first proposed in the 2017 paper ‘'Attention Is All You Need’ by (Vaswani, A et al., 2017), consists of an encoder-decoder architecture with both components making use of the well-known self-attention mechanism. The encoder processes the input sequence while the decoder generates the output sequence. The architecture can be seen in Figure 1.

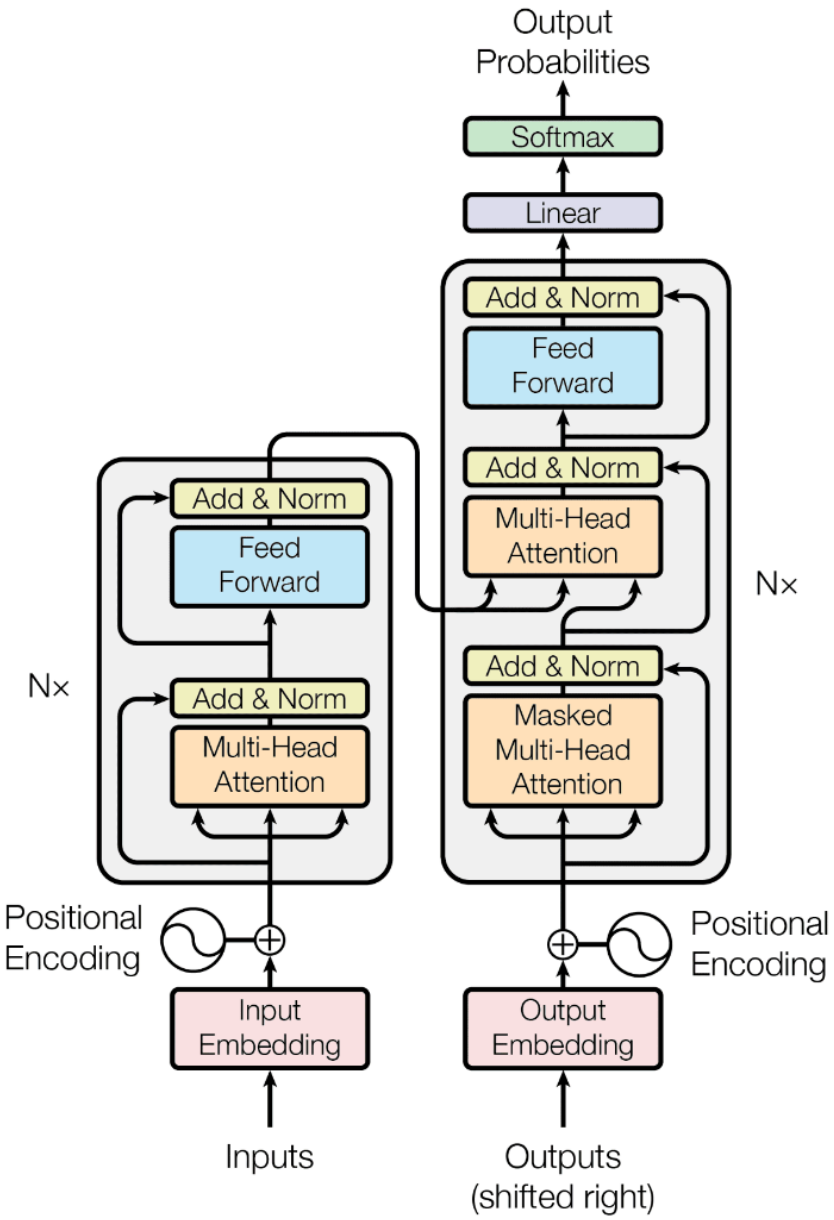


Figure 1 : Transformer Architecture "Attention is all you need"

Since the introduction of the original transformer model, various types of transformers have been developed with the architecture being adapted and expanded upon leading to different forms of transformer models such as auto-regressive transformers, auto-encoding transformers, and sequence-to-sequence based transformers.

Sequence-to-Sequence Transformers

Sequence-to-sequence transformers are the most similar to the original transformer model. The sequences are embedded to a higher dimension and translated by the decoder. These models such as Text-to-Text Transfer Transformer (T5) can be used for different types of natural language tasks although they are best served for language translation tasks due to their excellence at mapping the sequences between languages. However, researchers have still investigated using models such as T5 for sarcasm detection. (Maryam Najafi and Ehsan Tavan, 2022) propose a model using T5 for sarcasm detection in English and Arabic tweets. Paired with T5, the proposed model also made use of an LSTM layer to capture and utilize sequential information in the tweets. The results obtained were surprisingly successful leading them to use the same architecture for other subtasks. Their results highlighted the most significant effects on predictions being sentiment and emojis in text samples. A limitation, however, was the model's struggles with shorter text samples and text samples which lacked the presence of sentiments or emotion. While there's still room for improved performance, their work showcases the potential for using sequence-to-sequence models like T5 for sarcasm detection.

Auto-regressive Transformers

Auto-regressive models use probabilistic inference by leveraging prior tokens in order to generate text and predictions and hence rely more on the decoder of the transformer. These models are better suited for text generation tasks as they don't require an explicit input sequence, unlike sequence-to-sequence models. Popularised by the GPT series of models, other types of auto-regressive models are also used for classification tasks such as the Bidirectional Auto-Regressive Transformer (BART) and XLNET. Although these models haven't been widely adopted to solve tasks such as sarcasm detection, they showcase their capability in solving closely related tasks such as sentiment analysis. (Gong, Xinrong et al.,2019) propose a Broad Autoregressive Language Model (BroXLNet) for improving contextual representation for textual classification tasks such as sentiment analysis by searching additional features in broad space. Although they acknowledged the current setup used less complicated textual examples for modelling, the performance yielded a high accuracy, showcasing the model's

ability to improve contextual representations, a setting potentially useful for tasks which rely heavily on context such as sarcasm detection.

Auto-Encoding Transformers

Auto-encoding models such as BERT (Bidirectional Encoder Representations from Transformers) and RoBERTa (Robustly Optimised BERT) capture meaningful representations of input data in their encoded form and hence rely more on the encoder side of the transformer. To better capture dependencies and relationships within the input text, these models use a masked language modelling approach during pre-training, the model is trained to predict some words in a sentence that are hidden deliberately. This allows them to process both forward and backward contexts of an input sequence, thus acting as bidirectional encoders. Through fine-tuning, models such as BERT and RoBERTa have shown good performance for various types of natural language tasks including sarcasm detection. (Khatri & P, Fig-Lang 2020) use BERT in conjunction with GloVe embeddings to detect sarcasm in tweets. The approach involved analysing both the tweets and its contextual information. Paired with the GloVe embeddings, the BERT model was better able to better capture the subtle contextual nuances necessary for tasks such as sarcasm detection that heavily rely on context. However, the model fell short in terms of its ability to generalize. (Maciej Hercog et al., 2022) use a pre-trained Roberta model augmented with a 3-layer feed-forward fully connected neural network to detect sarcastic tweets in the “iSarcasm dataset”. The results obtained from their approach yielded an F1 score of 0.526. The lower performance was attributed to sarcasm's inherent reliance on context, for which they considered additional pre-training for the model to potentially improve performance.

2.3 GPT Models (OpenAI)

OpenAI was founded in December of the year 2015 to advance AI in a way that benefits humanity as a whole. Shortly after the release of the paper "Attention is all you need", OpenAi released GPT-1 which was based on the decoder architecture of the original transformer model. The model was able to take 12 layers of the decoder stacks and was trained on roughly 117 million parameters. After the success of GPT-1, OpenAi released GPT-2 which built up on

GPT-1 and was trained on a larger 1.5 billion parameters with a total of 48 layers. These models when fine-tuned were able to perform tasks such as question answering, text classification, language translation etc. In June 2020, OpenAi released the third generation of GPT (GPT-3) models. The model was trained on a staggering 175 billion parameters equivalent to 300 billion tokens of text, a capacity which is ten times larger than that of Microsoft's Turing NLG, the next largest NLP model known at the time (Wikipedia, 2023), setting up a new era for LLM's. The majority of the data, about 60 percent, was obtained from common crawl which had a total of 410 billion tokens. The other training data was obtained from WebText2(22%), Books1 and 2 (16%), and Wikipedia (3%). In terms of its architecture, the main architecture of the GPT-3 model is similar to that of the transformer mentioned in the original "attention is all you need" paper although the GPT-3 model integrates positional encodings at the embeddings of each input layer to address a problem the original transformer had where it lacked an innate understanding of the sequence or order of words. The GPT-3 model also has a far greater number of layers than any previous transformer model previously created. It consists of an encoder, decoder, and attention layers with multiple layers of transformer blocks utilizing a series of feed-forward neural networks to process and transform the data from the attention layers with each layer being able to learn different patterns and information from the data by using an activation function.

The GPT-3 models consist of various types depending on the parameter's size and task. Some of these include GPT-3 Curie, GPT-3 Babbage, and GPT-3 Davinci. OpenAI's GPT models are also capable of fine-tuning i.e., the weights of the pre-trained model are trained on new data. GPT-3.5 is a fine-tuned version of the GPT-3 model released by OpenAI in March 2022. In August 2023, OpenAI provisioned the GPT-3.5 model for fine-tuning on custom datasets aiming to add a new layer of optimised performance to the model. Similar to the GPT-3 model, GPT-3.5 comes in different variations based on parameter size and tasks. One of which is the GPT-3.5 turbo model. GPT-3 and its successors are capable of performing the same tasks as previous releases however due to their significantly larger number of parameters trained on them, these models also can perform tasks such as zero-shot and few-shot learning.

2.3.1 Zero Shot Learning

Zero-shot learning represents a model's ability in our case GPT-3.5 to classify data it hasn't been explicitly trained on. The model is given a task description and a defined prompt as seen in Figure 2 (Tom B. Brown et al , 2020). From Figure 2, a task is given for the model to translate English to French with no prior example translations provided. The zero-shot setting leverages the huge knowledge base of the OpenAI pre-trained model (GPT-3.5), enabling it to make classifications across different tasks without the need for training, hence this setting has been used to solve different types of natural language tasks with limited data. However, the zero-shot setting is also the most demanding environment. At times, even humans might struggle to grasp the task's structure without prior samples, making this scenario occasionally excessively tough (Tom B. Brown et al , 2020).

Figure 2 : Zero Shot Learning Example Prompt

2.3.2 Few Shot (In-Context Learning)

In-context learning, also known as few-shot learning, represents a model's ability, typically a pre-trained model, to learn context and classify data it hasn't been explicitly trained on based on a few provided examples. Compared with fine-tuning, in context learning does not update the weights of the pre-trained model, instead adapts a model to a task by conditioning it on a sequence of demonstrations (Tom B. Brown et al , 2020). The model is given a task description, and a suitable prompt along with a few examples as seen in Figure 3 (Tom B. Brown et al ,

2020). In this scenario, the model is provisioned for a language translation task. A task description is given in this case to translate English to French and some examples consisting of input-output pairs representing some language conversions are presented to the model. The model is then fed a new example which it autoregressive completes with what it considers to be the most likely tokens to generate a prediction. The number of examples given to the model is normally within a reasonable range (10 to 100 examples) as too many examples i.e., in the thousands would lean more towards the supervised learning paradigm. (Pezzotti N, 2021) proposed a few shot learning solution for dialogue state tracking. The work suggests a framework which selected in context learning examples based on similarity, a suggestion which outperformed the original random in-context selection framework. The work however showed some interesting observations with increasing model sizes not correlating to increased performance for the task of Dialogue State Tracking.

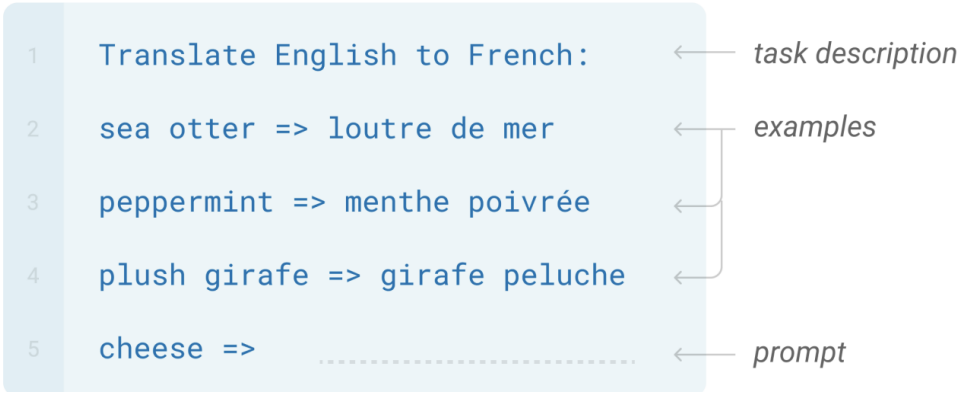


Figure 3 : Few Shot Learning Example Prompt

2 .4 Chosen Datasets

To conduct experiments a total of four datasets were chosen with each having its unique qualities. We look at past work done on these datasets as well as explaining the contents of each.

2.4.1 SARC Dataset

The Self-Annotated Reddit Corpus (SARC) created by (Khodak et al , 2017) and originally designed for contextual investigations consists of a total of 1.3 million comments obtained from Reddit with sarcastic entries being self-annotated by authors through the use of the \s token which Reddit users often use to denote sarcasm. The dataset consists of both a parent comment and the child comment making the dataset a gold standard for sarcasm detection as sarcasm in most cases hinges heavily on the contextual foundation provided by preceding statements. As a result, the dataset has gained significant popularity among researchers who utilize a range of algorithms in the hope of improving model performance for sarcasm detection on the SARC dataset. Some notable work done on this dataset include (Kolchinski & Potts, EMNLP 2018), (Hazarika et al., 2018), and Painter et al.,2022).

2.4.2 SAD Dataset

The SAD dataset was curated by Painter et al.,2022) by scraping tweets off Twitter using the TWINT library. The tweets were gathered by tracking the '#sarcasm' hashtag using the TWINT2 library and manually annotated by three annotators. The dataset contains a total of 2,340 data points containing an equal split of sarcastic and non-sarcastic tweets. The tweets are openly accessible using the Twitter API via the ID of the tweets. It is indicated that the dataset may contain bias, given the data originates from raw social media content. Previous work has been done on this dataset by Painter et al.,2022), in their work, they explored model performance using a machine learning approach using decision trees and logistic regression with text representation of Word2vec and GloVe Embeddings. The second approach was using deep learning algorithms such as Bert and RoBERTa. The results as seen in Figure 4 show acceptable model performance across all experiments, with the auto-encoding transformers significantly outperforming the classical machine learning models across all metrics.

	Word2Vec+LR			Word2Vec+DT			GloVe+LR			GloVe+DT		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
SARC	62.93	61.06	**61.98**	57.59	55.57	56.56	62.06	56.56	58.63	56.69	55.34	56.02
SAD	62.14	55.56	58.67	63.38	58.58	**60.89**	60.87	56.57	58.64	65.48	55.56	60.11

	BERT			BERTweet			RoBERTa$_{base}$			Twitter-RoBERTa			RoBERTa$_{large}$		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
SARC	73.91	79.47	76.59	76.52	80.35	**78.39**	76.23	78.35	77.30	74.89	80.52	77.61	77.65	77.57	77.61
SAD	65.89	71.21	68.45	77.36	62.12	68.91	81.49	93.43	**87.06**	82.19	90.90	86.33	86.84	83.33	85.05

Figure 4 : Evaluation Results Painter et al,2022

2.4.3 News Headlines Dataset

The news headlines dataset is collected from two news websites known as "*TheOnion*.3" and "*HuffPost4"*. The first website Onion3 produces sarcastic events of real time current events and the second *HuffPost4,* an American news media company produces non-sarcastic news headlines. The news headlines dataset gives a different feel to textual sarcasm detection from the traditional means of web scraping text from social media websites such as Twitter and Reddit as these aforementioned websites tend have lots of label and language noise, considering many tweets curated are replies to other tweets and hence lack proper context (Misra, Rishabh, 2022) in comparison to the news headlines datasets which is curated from websites solely designated for producing sarcastic and non-sarcastic comments, hence giving the news dataset an overall higher quality with less biased data for the task of sarcasm detection. The dataset consists of a total of 28k rows with a total of 11724 sarcastic comments and 14985 non-sarcastic comments. The dataset has been used extensively by researchers for tackling sarcasm detection problems ever since its introduction in 2022. (Santosh K.B et al , 2022) use the news headlines dataset for a bag of words analysis using both n-grams and term frequency to compare performance across different classifiers as seen in Figure 5 From the results we see

good performance for both accuracy and F1-score across all models in cases of default and hyper tuned settings.

Classifiers	**Hyperparameters**	**Accuracy**(%)	**F1-measure**
Naive Bayes	default	78.45	0.769
Multinomial Naive Bayes	default	77.63	0.761
Bernoulli Naive Bayes	default	78.31	0.751
Logistic Regression	penalty = l2	79.1	0.765
SGD Classifier	loss = 'modified-huber', penalty = 'elasticnet'	78	0.759
LinearSVC	default	77.7	0.757
NuSVC	default	78.6	0.753
Voted Classifier	none	**86.4**	**0.841**

Figure 5 : Evaluation Results (Santosh K.B et al , 2022)

2.4.4 iSarcasmEval Dataset

The iSarcasm dataset was curated in 2022 by (Abu Farha et al., 2022)._ In contrast to some alternative datasets used for sarcasm detection, during the data collection phase, the authors provided the sarcasm labels for each text. which eliminated the need for labelling intermediaries, such as pre-defined tags or annotators from third-party sources. The dataset consists of sarcastic text in English and Arabic however only English text will be considered for the purpose of this project. The dataset consists of 3 sub-tasks although the only subtask which fits the scope of the project is subtask one which determines if a given text is sarcastic or non-sarcastic. Subtask one was handed out to a total of 43 teams, with each team coming up with its unique approach for making classifications performance. From Figure 6 we see the top 20 highest-scoring teams with most top scoring teams using variations of BERT in their approach, from Figure 6 we also see the corresponding performances in terms of F1-score with the highest F1-score achieved being 60.5 and the lowest score achieved having an F1-score of 0.341. The average F1 score achieved from the top 20 scores was 0.441. Although the

performance is not ideal, the performance is reflective of the challenge in accurately predicting sarcasm detection current methods face.

r	Team Name	Affiliation	$F_1^{\text{sarcastic}}$
1	stce	PALI Inc., China	0.605
2	X-PuDu	Baidu & Shanghai Pudong Development Bank, China	0.569
3	TUG-CIC	TU Graz, Austria	0.530
4	Plumeria	Indian Institute of Technology Kanpur, India	0.477
5	John Thomson	University of Alberta, Canada	0.456
6	Naive	Dalian University of Technology, China	0.452
7	MarSan_AI	Part AI Research Center, Iran	0.434
8	LISACTeam	Sidi Mohamed Ben Abdellah University, Morocco	0.429
9	LT3	Ghent University, Belgium	0.424
10	niksss	-	0.402
11	Amobee	-	0.401
12	YNU-HPCC	Yunnan University, China	0.392
13	Dartmouth	Dartmouth College, USA	0.386
14	underfined	Ping An Life Insurance Company of China, China	0.383
15	CS-UM6P	Mohammed VI Polytechnic University, Morocco	0.371
16	UTNLP	University of Tehran, Iran	0.369
17	Jumana-Safa	-	0.356
18	cnxup	University of Chinese Academy of Sciences, China	0.351
-	baseline-bert	-	0.348
19	IISERB Brains	Indian Institute of Science Education and Research, Bhopal, India	0.345
20	rematchka	Cairo University, Egypt	0.341

Figure 6 : Top 20 F1-scores achieved (Abu Farha et al., 2022)

Previous work on these 4 datasets showcases their suitability for use in this project as well as their adaptability for both fine-tuning and performing tasks such as zero-shot and few-shot learning and also serve as a benchmark for evaluating our model's performance.

Chapter 3 – Approach

3.1 Proposed Model

OpenAI possesses a vast range of models each with its strengths and weaknesses. To select the best model to be used for experimentation, firstly different factors such as the rate maximum tokens per request, cost per token, model parameter size, and the ability of the model to be fine-tuned were considered. After review of these factors and taking into consideration the GPT-3.5 turbo model has been recommended by OpenAI due to its lower cost(1/10th the cost of text-davinci-003 model) and its improved performance, the GPT-3.5 turbo model was the selected model for experimentation. GPT-3.5 turbo is a language model trained through a combination of advanced machine learning techniques, including deep neural networks and reinforcement learning from Human Feedback (RLHF) which has been optimised for chat and is capable of understanding as well as generating natural language or code. Figure 7 by (Alan D. Thompson, 2022) showcases the reinforcement learning process for the model.

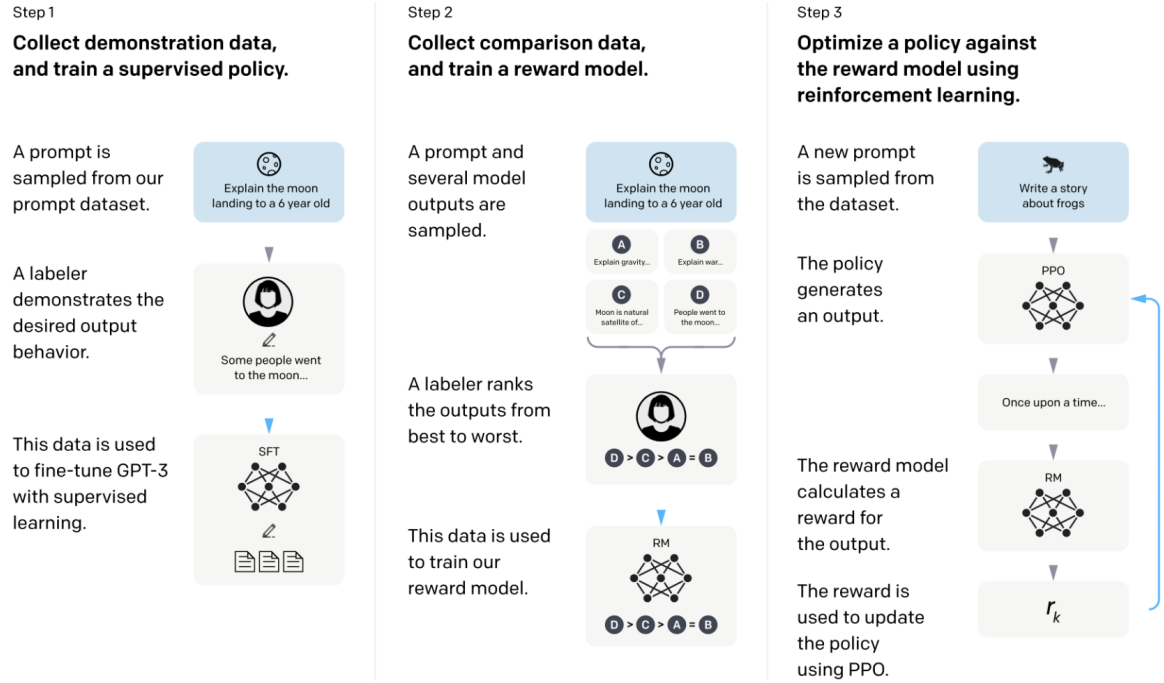


Figure 7 : Reinforcement Learning Cycle GPT-3.5 Turbo

3.2 Prompt Engineering

Prompt engineering aims to design well-crafted prompts to get better performance from a model. While models like GPT-3.5 Turbo are advanced, they require clear direction to deliver highly accurate responses. Hence for a prompt to be effective it needs to be both clear and precise. We see an example of an unclear prompt provided by (Pezzotti N, 2021) in Figure 8. The prompt is provisioned and asks the model to classify a sentiment without specifying all possible answers within the sample space. Although the model correctly classifies the Positive instance, we see from the list of possible answers the model is not completely sure what answers should be in the sample space which could lead to overall lower model accuracy down the line. This is a problem more prevalent in models which have not been fine-tuned for a specific task as the model does not possess specific knowledge of the exact label space for its responses.

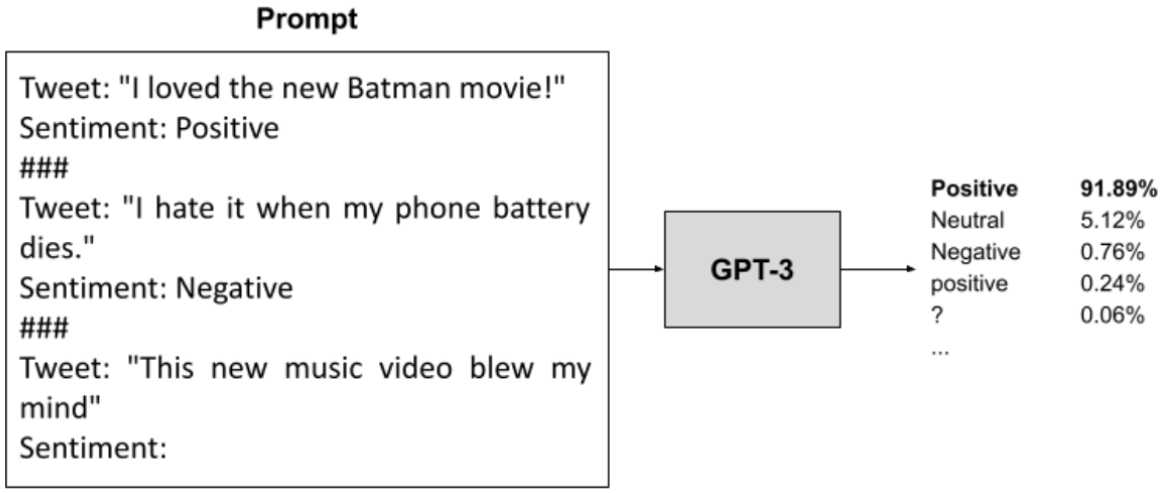


Figure 8 : Example Of Unclear Prompt

To design a suitable prompt for the zero show and a few shot models, we looked into several articles on prompting design. (Robinson R, 2023) shows prompts could be both precise and simple. Using keywords such as strictly also better helps the model know only a particular set of answers is wanted. It was also decided not to make use of polite words such as please and thank you as from past work use of these words seems to not affect performance while

increasing the token size of the prompt which increases cost and potentially reduces clarity (Jones S, 2023). The prompts for each of the three datasets are also slightly tweaked based on the context of the data i.e., the prompt for tweets data is “Strictly classify the following text as either sarcastic or non-sarcastic” while the prompt for the news headlines dataset is slightly tweaked to incorporate the data being news headlines i.e., “Strictly classify the following news headline as either sarcastic or non-sarcastic”. For the fine-tuned models, less care is paid to keywords such as “strict" and “either” hence these can be removed from the prompt leading to simpler and shorter prompts.

3.3 Experimentation Setup

To ensure optimal project flow, the project makes use of the CRISP-DM methodology. The project will make use of 4 datasets for experimentation as mentioned in section 2.4. The smallest dataset is the SAD dataset with a total of 2340 rows which is much smaller than the other three datasets. Hence to remain consistent with results throughout the project all experimentations will have the same sample size for train and test sets with a total of 1,500 data points allocated for testing and 500 examples allocated for the train set. The amount of fine-tuning will vary across different experiments. Each dataset will be fine-tuned independently, meaning each dataset will have its separate model. For the few shot scenarios, all experiments will make use of 4 settings with 10, 20, 30 and 50 examples provided. These examples are randomly selected from the train set. For each few shot-setting, an equal number of sarcastic and non-sarcastic examples will be used i.e., 5 sarcastic and 5 non-sarcastic examples for the 10-example setting. This is important as few-shot learning model’s only take a small number of examples for training, hence a small imbalance in label representation could have a pronounced effect on the model's learning ability and subsequent performance.

Experiment 1 compares model performances across all four datasets for base GPT-3.5 turbo across both zero-shot and few-shot settings. Experiment 2 compares performances for the base GPT-3.5 turbo versus the fine-tuned GPT-3.5 turbo. The selected sample for fine-tuning was 340 examples as this was the remaining data points left in the SAD dataset after the train test splits were taken from the initial 2k sample rows. Hence, to maintain consistency in results each of the models for the other three datasets will be fine-tuned using 340 examples from their

respective datasets. Experiment 3 builds on the second experiment and compares model performance across the datasets for the fine-tuned models with 340 examples compared to a much larger fine-tuned sample(2000 examples). As the Sad dataset only contains 2340 data points, experiment 3 will compare performances between the other three datasets (Sarc, News Headlines, iSarcasm).

Chapter 4 - Implementation

4.1 Initial Pre-processing

The OpenAI GPT-3.5 turbo model is designed to understand and process raw input data hence common text pre-processing techniques such as stemming, lemmatization, tokenization, stop word removal, etc. are not considered. With context already limited for tasks such as sarcasm detection, and capitalization being an important cue for denoting sarcasm (Diana Maynard and Mark Greenwood, 2014), paired with the GPT-3.5 turbo model having been trained on a mixture of lowercase and capitalised text, it was decided not to convert all text to lower case when passing it through the model.

In terms of pre-processing done for each dataset, common checks such as missing value checks were done to check the usability of the datasets. Some alterations were made in the columns with unclear label names changed for better clarity i.e., the Sad dataset label was changed from ‘sarcastic’ to ‘Label’. Any columns not being used were also dropped. For the Sarc dataset, the parent comment and child comment columns were merged using data transformation techniques in order for both parent and child comments to be used in its model's prompt later. Similar pre-processing was done for the news headlines and iSarcasm datasets when applicable.

All data entries were screened for the presence of words such as 'sarcasm', and 'non-sarcastic', and their variants with these keywords being removed from the rows containing them. This was done to reduce bias in results and to ensure the model does not make predictions solely on the word sarcasm being present in a sentence. The datasets were then split into training and testing sets.

4.2 Exploratory Data Analysis

Exploratory data analysis was carried out during and after pre-processing for each dataset to gain some insights from the data to be used. These included checking the label distributions, text character counts, unigram analysis, etc. The label distribution check can be seen in Figure 9. From the figure, we see the label distribution across the test sets for each dataset. The SAD, NEWS, and SARC datasets show a good distribution between the sarcastic and non-sarcastic classes. The iSarcasm dataset shows a heavy imbalance to the non-sarcastic class label with this level of imbalance also similar to its training set. Normally with this level of imbalance techniques such as over-sampling and under-sampling would be considered. For this project however, care has been taken to make sure an equivalent amount of sarcastic and non-sarcastic examples are contained within each prompt for the few-shot scenarios, hence the imbalance in the label distribution is not further considered. This class imbalance, however, could potentially

skew the F1-scores yielded during evaluation meaning closer care will be paid to the results yielded during the evaluation phase.

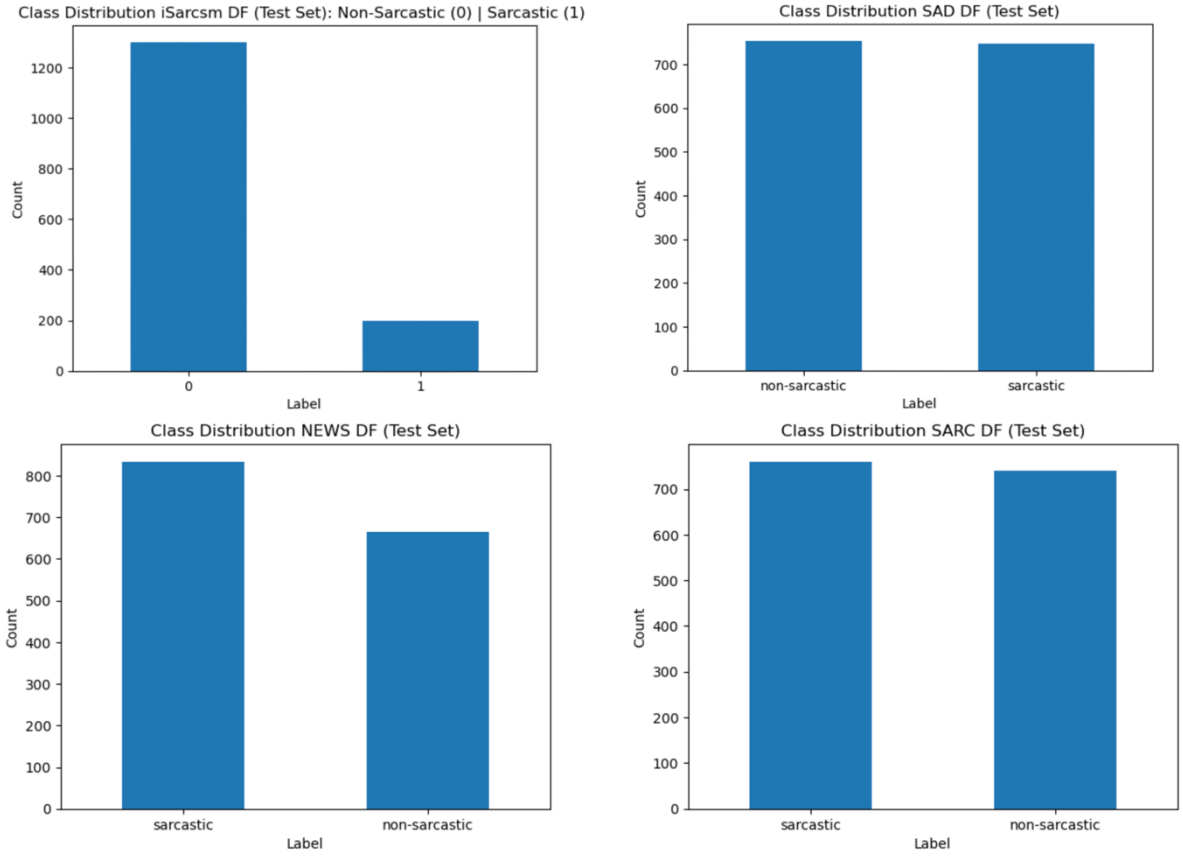


Figure 9 : Label Distributions Test Sets

4.3 Fine-Tuning Process

Unlike GPT completion models which respond based on singular prompts, fine-tuning the GPT-3.5 turbo model requires feeding the model 3 role scenarios, one for the system, one for the user, and one for the assistant as seen from an example role provided by OpenAI in Figure 9. These roles provide specific types of information to the model and dictate model behaviour during fine-tuning. The system role provides high-level instructions to the model.

For the proposed sarcasm detection model, the system's role was distinctly defined as, "You are a chatbot capable of detecting sarcasm in text". Although this is slightly tweaked based on the dataset i.e., for the news dataset this would be "You are a chatbot capable of detecting sarcasm in news headlines". This provides a clear and precise understanding of the system's

expected functionality. The user role represents the input passed to the model in the form of prompts. These prompts contain the example text data which is to be fine-tuned. The assistant role contains the response from each user role prompt. In this use case, this would be the labels of the text data from user input prompts i.e. [‘sarcastic’ or ‘non-sarcastic’].

```
{"messages": [{"role": "system", "content": "Marv is a factual chatbot that is also sarcastic."},
{"role": "user", "content": "What's the capital of France?"}, {"role": "assistant", "content": "Paris,
as if everyone doesn't know that already."}]}
```

Figure 10 : Example Role For Fine-tuning GPT-3.5 Turbo

After defining the roles, the data is formatted and saved to a JSONL((JSON Lines) file which is the recommended format for fine-tuning. The file is then uploaded to OpenAI for fine-tuning. The model is then fine-tuned by calling the Fine-Tuning Job method (Figure 10). From this, the response can then be printed showing details such as the progress of the fine-tuning job, the number of epochs trained on, etc. For fine-tuning across all datasets, the default number of epochs (3) was used. This choice considered the overall cost and time for fine-tuning on larger numbers of epochs.

```
finetune = openai.FineTuningJob.create(training_file="file-iOFAgX5QeL2ikmxqb70k0CIf", model="gpt-3.5-turbo")
```

```
print(finetune)
```

```
{
  "object": "fine_tuning.job",
  "id": "ftjob-TI9iYSRNZ0TTzznfDA0oazIH",
  "model": "gpt-3.5-turbo-0613",
  "created_at": 1692834808,
  "finished_at": null,
  "fine_tuned_model": null,
  "organization_id": "org-CYjHEhmYPFZM91JdgazGfxVF",
  "result_files": [],
  "status": "created",
  "validation_file": null,
  "training_file": "file-iOFAgX5QeL2ikmxqb70k0CIf",
  "hyperparameters": {
    "n_epochs": 3
  },
  "trained_tokens": null
}
```

Figure 11 : Fine-tune Job Information

Once a fine-tuning job has started it normally takes a considerable amount of time to finish with the time taken to finish fine-tuning increasing as the dataset sample to be fine-tuned increases. Once the fine-tuning job is completed a notification is sent to the corresponding

email associated with the OpenAI account (Figure 11) and the fine-tuned model is then ready for use.

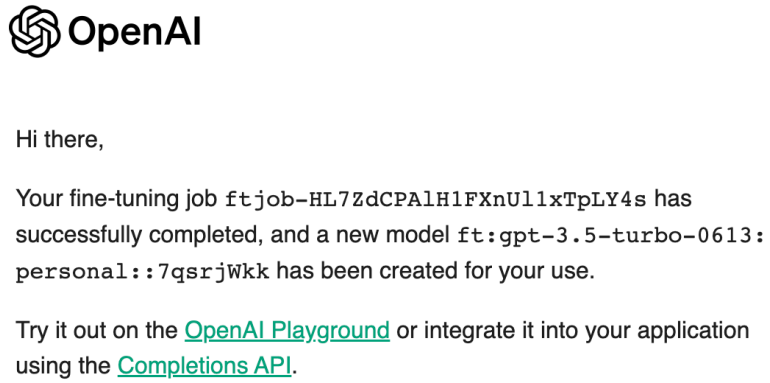


Figure 12: Example Confirmation Of Fine-tuned Model

4.4 Model Building

All models were constructed utilizing the OpenAI API, which provides access to a range of models developed by OpenAI, including the chosen model for experimentation GPT-3.5 Turbo. GPT3.5 turbo is a chat completion model i.e., the model takes a list of text as input and returns a model-generated message as output, hence the chat completion API provisioned by OpenAI is used. Two functions were defined, one for the zero-shot scenario and the second for the few-shot scenario (Figure 12). As the task is for zero/few-shot learning, both functions use the system role only with the content containing the defined prompts. The maximum token response length is set to 100 in the unlikely case of the model providing ambiguously long answers. The few-shot formula is similar to the zero-shot formula however it takes an extra parameter for the category examples. This is also reflected in its prompt, which instructs the model to factor in the given examples when making predictions.

```python
def zero_shot_learning_gpt3_turbo(model, input_texts):
    classifications = []
    for input_text in input_texts:
        prompt = f"\nStrictly classify the following news headline as either sarcastic or non-sarcastic .  No Extra information is required. : {input_text}\n"
        response = openai.ChatCompletion.create(
            model=model,
            messages=[
                {
                    "role": "system",
                    "content": prompt
                }
            ],
            max_tokens=100,
            stop=None
        )

        classification = response.choices[0].message['content'].strip().lower()
        classifications.append(classification)


    return classifications

def few_shot_learning(model, input_texts, category_examples):
    classifications = []

    for input_text in input_texts:
        prompt = f"Possible Categories: {', '.join(categories)}\n"

        # Few-shot learning prompts with your examples
        for category in categories:
            prompt += f"{category.capitalize()} Examples: {', '.join(category_examples[category])}\n"


        prompt += f"\Based on the provided examples, strictly classify the following news headline as either sarcastic or non-sarcastic. No extra information is required. : {input_text}\n"
        response = openai.ChatCompletion.create(  # Use the correct chat completion method
            model=model,
            messages=[
                {
                    "role": "system",
                    "content": prompt
                }
            ],
            max_tokens=100,
            stop=None
        )

        classification = response.choices[0].message['content'].strip().lower()
        classifications.append(classification)

    return classifications
```

Figure 13 : Python Code Using OpenAI API For Zero And Few-Shot Learning

Due to rate limits with GPT3.5 turbo, the data is sent in chunks with each chunk containing 100 data points for classification making a total of 15 chunks. In some cases however, when dealing with larger prompt sizes the chunk size was adjusted accordingly. To run the model the zero shot or few shot formula is called passing in the respective parameters including the model's name *"gpt-3.5-turbo"*. For the fine-tuned models this parameter would be the equivalent to the name of the fine-tuned model for example *"ft:gpt-3.5-turbo-*

0613:personal::7qrtFyuT ". Once all chunks have been run, all predictions are results are appended to a list for later pre-processing.

```
input_texts = [item for item in all_chunks[0]['text']]
classifications = zero_shot_learning_gpt3_turbo("gpt-3.5-turbo" , input_texts)
zeroshot_pred0 = pd.DataFrame(classifications)

input_texts = [item for item in all_chunks[0]['text']]
classifications = few_shot_learning("gpt-3.5-turbo", input_texts, category_examples)
fewshot_10ex_pred0 = pd.DataFrame(classifications)
```

Figure 14: Example Run For Zero And Few Shot Settings

4.5 Results Processing

Results from each chunk are concatenated once experiments are concluded. Upon checking the value counts of the predicted (Figure 15). From the Sad dataset, we see even after careful prompt selection the base GPT3.5 turbo model without fine-tuning still had problems classifying labels strictly as sarcastic or non-sarcastic. Some predictions contain variants of the desired predictions such as "sarcasm" instead of "sarcastic" while some contain a sentence with "sarcasm" at the end. We also see the model made some predictions and added a full stop at the end. To fix this we first define a function to check the number of tokens per prediction and filter out predictions with token lengths greater than a threshold number. These filtered predictions are then replaced with None and later discarded from the prediction's dataset. For the predictions within the given threshold, we use a regular expression search to find predictions that match defined keywords such as 'sarcasm', and 'non-sarcasm'. The predictions containing these keywords are then converted to the desired format using a replacement method call for example "sarcasm" to "sarcastic" and "non-sarcasm ." to "non-sarcastic". Any predictions not containing the defined keywords are set to "None" and discarded from the prediction's dataset. Some challenges however do exist when pre-processing the results, for example, some predictions contained both keywords ("sarcastic"/ "non-sarcastic") which could lead to confusion hence this was also set to "None" considering there were only a few instances

where this happened and for all instances the prediction was inconclusive. Another challenge from the predictions can again be seen in Figure 15, we see one prediction that says in quote “not enough information to determine sarcasm”. This prediction's result is inconclusive however with the presence of the word sarcasm in it using a regex search and classifying any instances with the defined keywords such as “sarcasm” without proper inspection of the results would lead to a misclassification in this scenario. Hence care was taken when pre-processing results. Each result pre-processing was handled on a case-by-case basis although most of the earlier steps mentioned can be applied. From Figure 15, we see the predictions for the few-shot scenario had different keywords and different token lengths for acceptable predictions compared to the zero-shot scenario, hence careful consideration was paid to the token length threshold set.

```
sad_df_zeroshot['predicted'].value_counts()
```

```
predicted
non-sarcastic
1043
sarcastic
394
non-sarcastic.
33
sarcasm
25
sarcastic.
2
not enough information to determine sarcasm.
1
since the provided link is invalid, i cannot access the text to classify it as sarcastic or non-sarcastic.
1
the provided text does not contain any meaningful content, so it cannot be classified as sarcastic or non-sarcastic.
1
Name: count, dtype: int64
```

```
sad_df_fewshot10ex['predicted'].value_counts()
```

```
predicted
sarcastic                                                                                    767
non-sarcastic                                                                                709
non-sarcastic.                                                                                 8
sarcastic.                                                                                     4
category: non-sarcastic                                                                        3
sarcastic example                                                                              2
sarcasm                                                                                        2
category: sarcastic                                                                            1
sarcasitc                                                                                      1
possible category: non-sarcastic                                                               1
classification: non-sarcastic                                                                  1
without any specific text to analyze, it is impossible to determine whether it is sarcastic or non-sarcastic.    1
Name: count, dtype: int64
```

Figure 15 : Unclear Predictions SAD Dataset

After completing the results pre-processing, the outcomes containing model predictions are saved into their respective CSV files, ready for subsequent evaluation.

CHAPTER 5 - Results and Evaluation

5.1 Evaluation Metrics

Different evaluation metrics such as accuracy , precision , recall , F1-Score, and AUC are used to judge the performance of the models. For each metric the macro average is used. Previous work done on our chosen datasets have made use of F1-Score as the priority metric to judge overall model performance. Hence, to accurate compare overall model performance to previous works done on these datasets, the F1-Score is the chosen priority metric. From each best performing model across each dataset, an ROC curve is visualised to better understand the trade-off between the true positive rate and the false positive rate for the predictions made.

5.2 Base GPT-3.5 Turbo

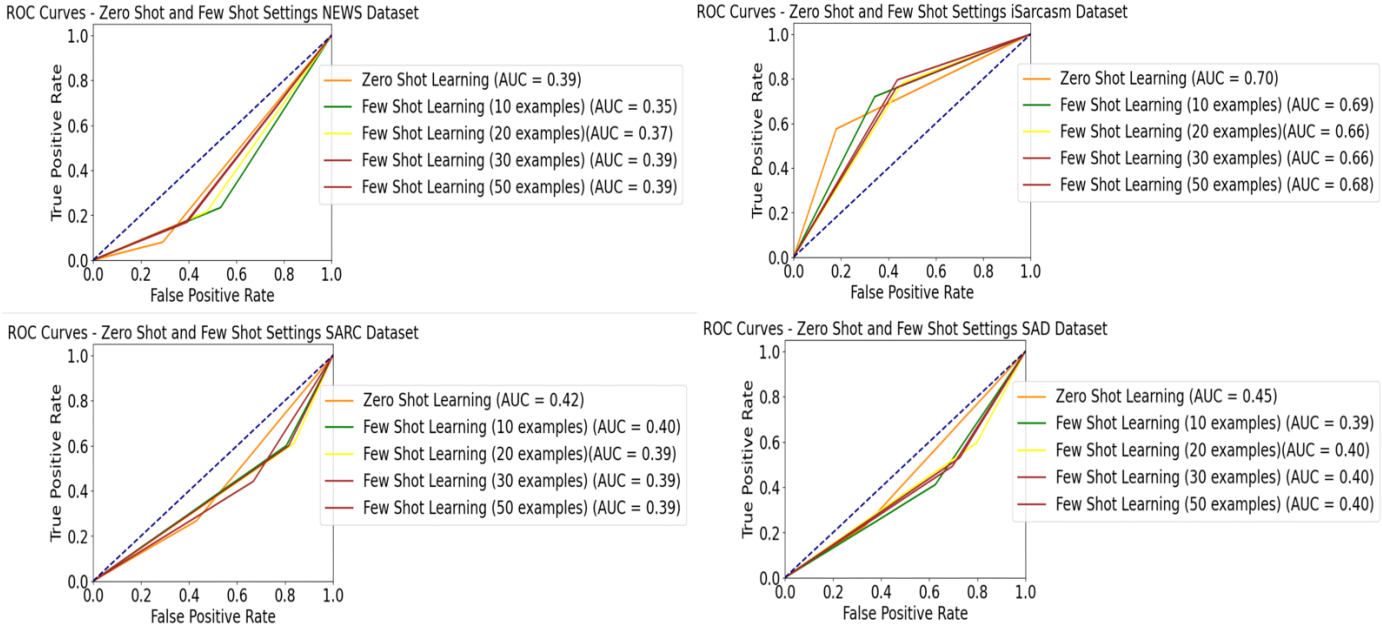


Figure 16 : ROC Curves Base GPT-3.5 Turbo

Dataset	Setting	Accuracy	Precision	Recall	F1-Score	AUC
News Headlines	Zero Shot	0.36	0.32	0.39	0.31	0.39
	Few Shot (10 ex)	0.34	0.34	0.35	0.33	0.35
	Few Shot (20 ex)	0.35	0.35	0.37	0.34	0.37
	Few Shot (30 ex)	0.37	0.36	0.39	0.35	0.39
	Few Shot (50 ex)	0.36	0.36	0.39	0.34	0.39
	Average	0.356	0.346	0.378	0.334	0.378
iSarcasm	Zero Shot	0.79	0.63	0.70	0.64	0.70
	Few Shot (10 ex)	0.67	0.59	0.69	0.57	0.69
	Few Shot (20 ex)	0.57	0.57	0.66	0.51	0.66
	Few Shot (30 ex)	0.59	0.58	0.66	0.52	0.66
	Few Shot (50 ex)	0.59	0.58	0.68	0.52	0.68
	Average	0.642	0.59	0.678	0.552	0.678
SARC	Zero Shot	0.42	0.41	0.42	0.40	0.42
	Few Shot (10 ex)	0.40	0.38	0.40	0.37	0.40
	Few Shot (20 ex)	0.39	0.36	0.39	0.36	0.39
	Few Shot (30 ex)	0.39	0.37	0.39	0.39	0.39
	Few Shot (50 ex)	0.39	0.39	0.39	0.39	0.39
	Average	0.398	0.382	0.398	0.382	0.398
SAD	Zero Shot	0.45	0.44	0.45	0.42	0.45

	Few Shot (10 ex)	0.39	0.39	0.39	0.39	0.39
	Few Shot (20 ex)	0.40	0.38	0.40	0.37	0.40
	Few Shot (30 ex)	0.40	0.39	0.40	0.39	0.40
	Few Shot (50 ex)	0.40	0.39	0.40	0.39	0.40
	Average	0.408	0.398	0.408	0.392	0.408

Table 1 : Model Performances Experiment 1 (Base GPT-3.5 Turbo)

The results from the first experiment yield low performances as seen in Table 1, across all settings for the News, Sad, and Sarc datasets with performances across all metrics close to the 40-percentile range. The iSarcasm dataset shows significantly better performance compared to the three aforementioned datasets across each variation with an average F1-Score of 0.552. However, this could be attributed to the imbalance previously mentioned in this dataset. For the News dataset, the F1 score increases as the number of examples fed to the model increases barring the few-shot 30 examples and 50 examples scenarios, however, this is not a trend that is followed by the other 3 datasets.

5.3 Fine-tuned GPT-3.5 Turbo (340 examples)

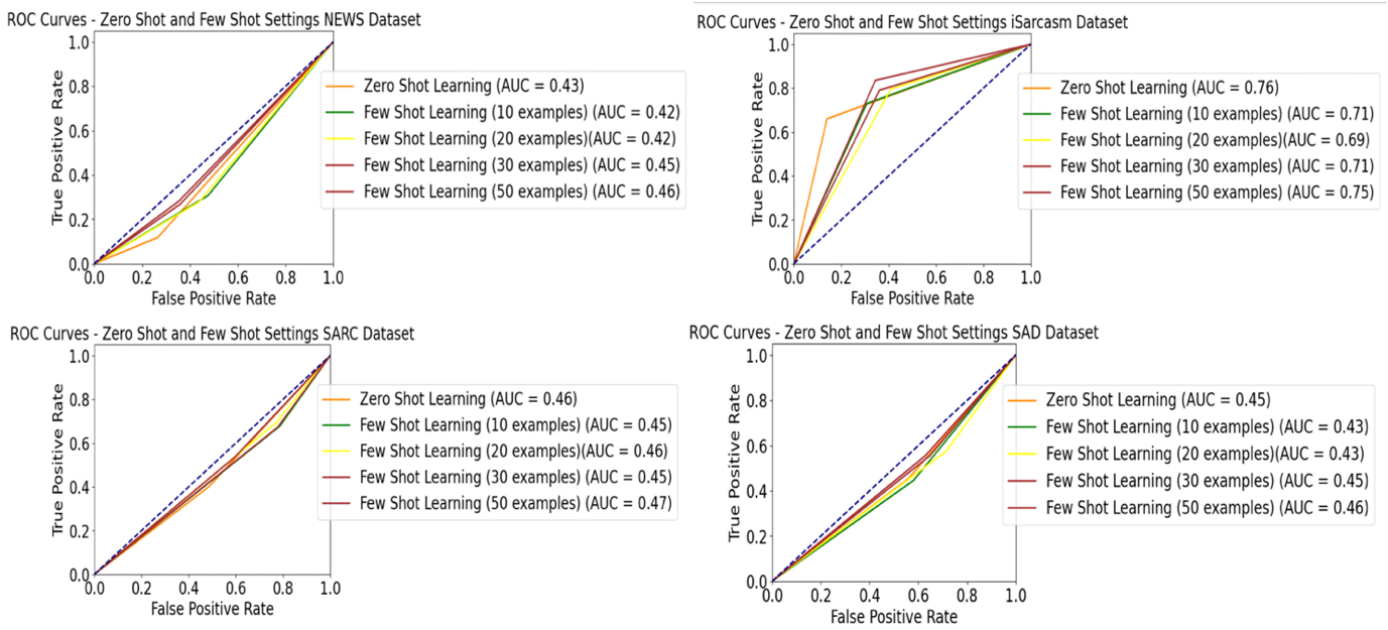


Figure 17 : ROC Curves Fine-tuned GPT-3.5 Turbo (340 examples)

Dataset	Setting	Accuracy	Precision	Recall	F1-Score	AUC
News Headlines	Zero Shot	0.39	0.38	0.43	0.35	0.43
	Few Shot (10 ex)	0.40	0.41	0.42	0.40	0.42
	Few Shot (20 ex)	0.40	0.41	0.42	0.40	0.42
	Few Shot (30 ex)	0.43	0.43	0.45	0.42	0.45
	Few Shot (50 ex)	0.44	0.46	0.46	0.43	0.46
	Average	0.412	0.418	0.436	0.40	0.436
iSarcasm	Zero Shot	0.83	0.68	0.76	0.71	0.76
	Few Shot (10 ex)	0.70	0.61	0.71	0.60	0.71

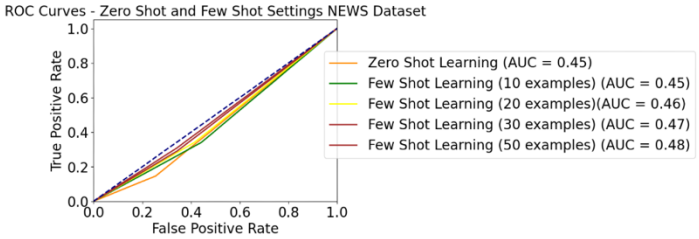
	Few Shot (20 ex)	0.62	0.59	0.69	0.54	0.69
	Few Shot (30 ex)	0.66	0.60	0.71	0.57	0.71
	Few Shot (50 ex)	0.68	0.62	0.75	0.60	0.75
	Average	0.698	0.62	0.724	0.604	0.724
SARC	Zero Shot	0.46	0.46	0.46	0.46	0.46
	Few Shot (10 ex)	0.45	0.43	0.45	0.42	0.45
	Few Shot (20 ex)	0.46	0.45	0.46	0.43	0.46
	Few Shot (30 ex)	0.45	0.43	0.45	0.42	0.45
	Few Shot (50 ex)	0.47	0.47	0.47	0.47	0.47
	Average	0.458	0.448	0.458	0.44	0.458
SAD	Zero Shot	0.45	0.45	0.45	0.45	0.45
	Few Shot (10 ex)	0.43	0.43	0.43	0.43	0.43
	Few Shot (20 ex)	0.43	0.42	0.43	0.42	0.43
	Few Shot (30 ex)	0.45	0.45	0.45	0.45	0.45
	Few Shot (50 ex)	0.46	0.45	0.46	0.45	0.46
	Average	0.444	0.44	0.444	0.44	0.444

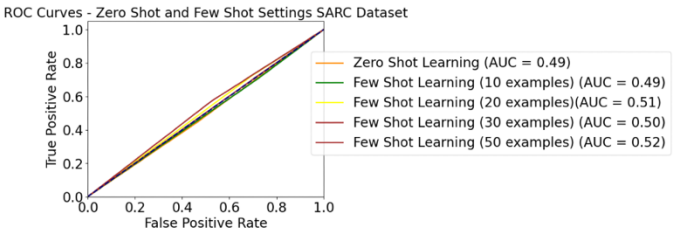
Table 2 : Model Performances Experiment 2 (Fine-tuned GPT-3.5 Turbo - 350 examples)

The results from experiment 2 show improved performance compared to the base GPT-3.5 turbo model with the F1-score increasing roughly by 10 - 20 percent across each dataset. The best-performing dataset was the iSarcasm dataset with an average F1-score of 0.604. The other

3 datasets had F1 scores in the 40-percentile range with the News dataset having the lowest performance out of the 3 (0.40).

5.4 Fine-tuned GPT-3.5 Turbo 2000 examples





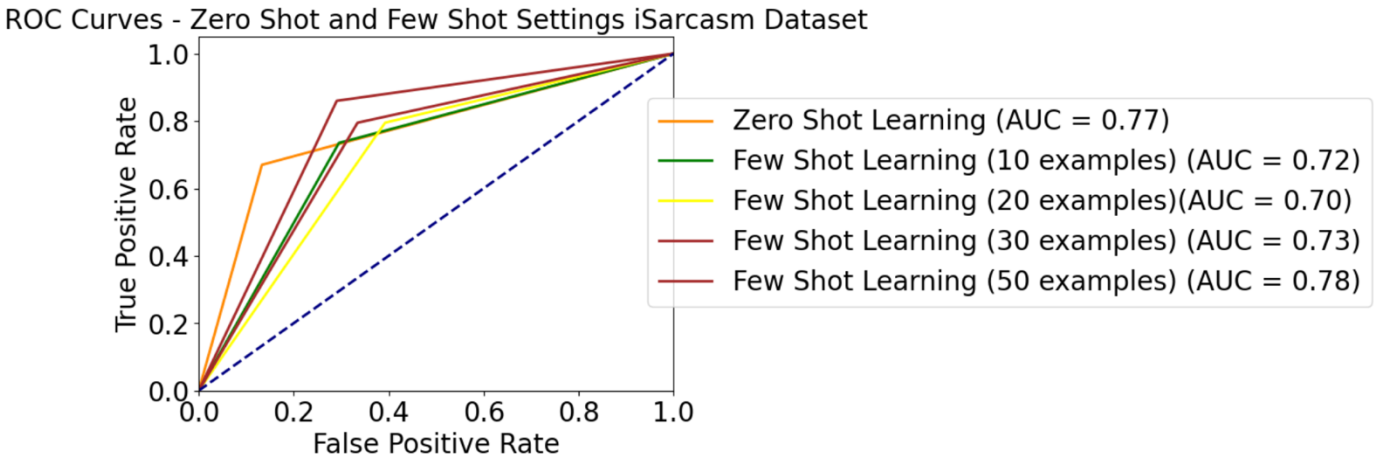


Figure 18 : Model Performances Experiment 2 (Fine-tuned GPT-3.5 Turbo - 2000 examples

Dataset	Setting	Accuracy	Precision	Recall	F1-Score	AUC
News Headlines	Zero Shot	0.41	0.41	0.45	0.37	0.45
	Few Shot (10 ex)	0.44	0.45	0.45	0.43	0.45
	Few Shot (20 ex)	0.45	0.46	0.46	0.45	0.46
	Few Shot (30 ex)	0.45	0.47	0.47	0.44	0.47

	Few Shot (50 ex)	0.46	0.48	0.48	0.46	0.48
	Average	0.442	0.454	0.462	0.43	0.462
iSarcasm	Zero Shot	0.84	0.69	0.77	0.72	0.77
	Few Shot (10 ex)	0.71	0.62	0.72	0.60	0.70
	Few Shot (20 ex)	0.63	0.59	0.70	0.55	0.72
	Few Shot (30 ex)	0.68	0.61	0.73	0.59	0.73
	Few Shot (50 ex)	0.73	0.64	0.78	0.64	0.78
	Average	0.718	0.63	0.74	0.62	0.74
SARC	Zero Shot	0.49	0.49	0.49	0.49	0.49
	Few Shot (10 ex)	0.50	0.49	0.49	0.47	0.49
	Few Shot (20 ex)	0.52	0.52	0.51	0.49	0.51
	Few Shot (30 ex)	0.50	0.50	0.50	0.48	0.50
	Few Shot (50 ex)	0.52	0.52	0.52	0.52	0.52
	Average	0.506	0.504	0.502	0.49	0.502

Table 3 : Model Performances Experiment 2 (Fine-tuned GPT-3.5 Turbo - 2000 examples)

The results from the third experiment again show an improved performance compared to the models fine-tuned with a sample set of 350 examples for each dataset. The iSarcasm dataset remained the best-performing dataset across all metrics, although it showed the lowest percentage increase in F1-score with an average F1-score of 0.62 (2.65% increase). The Sarc dataset shows improved performance across all metrics with its F1-score close to the 50% range (0.49), an 11.36 percentage increase compared to its previous average from experiment 2. The

News dataset shows improved performance (7.5% increase) across all metrics, although it was the lowest-performing dataset with an F1-score of 0.43.

CHAPTER 6 - Conclusion and Future Work

From the experiments conducted, no clear pattern could be observed between the performance of zero-shot learning and few-shot learning. Hence the initial hypothesis which stated that an increase in the number of examples provided in a prompt would lead to better model performance is rejected. Although it initially appears logical to presume that providing more examples would translate to enhanced performance, a possible reason for the unexpected correlation in performance could be due to too much noise in the prompt. The second and third experiments involved comparisons of the base GPT-3.5 turbo to fine-tuned versions of the model. The performance increased on all datasets when initially fine-tuned on just 340 examples. For experiment 3, although there was an increase in performance across each dataset compared to the models fine-tuned with 340 examples, the increase was not as substantial as the increase previously noted for base vs. initial fine-tuned models. As a result of the improved performance when fine-tuned and the subsequent increase when further fine-tuned on more examples, we can accept the hypothesis initially set for both objectives 2 and 3 respectively. The initial model performances with no fine-tuning yielded poor results across the News, SAD, and SARC datasets. compared to previous works done on each dataset. Although performance increased with fine-tuning, the F1-Scores yielded performances close to the 50-percentile range i.e., equivalent to a random guess for all the aforementioned datasets which unsurprisingly was below the performance threshold set in previous works done on said datasets. However, this low performance could be attributed to factors such as a bad quality sample used, unclear prompts, and further refinement during the pre-processing stage for better quality data overall. The iSarcasm dataset showed promise with a maximum average F1-score of 0.62 achieved when fine-tuned with 2000 examples. This score in comparison exceeded the top scoring score

achieved (0.605) during the team invitational by (Abu Farha et al., 2022) mentioned in section 2.4.4. The yielded performance also surpassed the benchmark for human recognition (0.616) set by (Oprea Silviu & Magdy Walid, 2019) using third-party human annotation for coming up with predictions for sarcastic text. This performance in the iSarcasm dataset shows promise in using techniques that mirror human recognition such as zero and few-shot learning for NLP tasks such as sarcasm detection.

In terms of future work, currently, the OpenAI fine-tuning is only provisioned on the GPT-3.5 Turbo model. In the case fine-tuning is made available for the GPT-4 model this is an area that could potentially be explored. Currently, fine-tuning is being done on each dataset, As of the time of writing, fine-tuning models are also fairly expensive to run so in the case this is reduced in the future, we could investigate fine-tuning on similar tasks for example concatenating different Twitter datasets and fine-tuning on them and comparing performance. As of writing OpenAI GPT-3.5 turbo model is rate-limited with a request per limit (rpm) of 3,500 and tokens per limit (tpm) of 90,000. This makes testing on entire datasets, for example, the entire SARC dataset comprising of one million example data points both extremely time-consuming and costly. If this rate limit is either removed or drastically reduced in the future, rather than testing on samples of each dataset, testing model performance on the entire dataset would be a potential consideration for future work. Lastly, assuming, a contributing factor to the lower model performances for the few-shot models in comparison to the zero-shot models was due to inadequate prompts, a potential future work would be to explore better strategies for prompt engineering.

Ethics Statement

Due to the sensitive nature of the datasets and to enforce a sense of anonymity for curators of the sarcastic texts for the Reddit and Twitter datasets, it was ensured that the chosen datasets already had information such as the originator of the tweets / Reddit comments removed as well as personal information relating to the user such as their username, user ID, email address, phone number etc. removed before the acquisition of each dataset.

References

[1] Silviu Oprea and Walid Magdy. 2019. Exploring Author Context for Detecting Intended vs Perceived Sarcasm. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2854–2859, Florence, Italy. Association for Computational Linguistics.

[2] Bing Liu , 2010. Sentiment analysis and subjectivity. Handbook of natural language processing, 2(2010):627–666.

[3] Yaghoobian H, Arabnia HR, Rasheed K. Sarcasm Detection: A Comparative Study. arXiv preprint arXiv:2107.02276. 2021.

[4] Ehsan Hosseini-Asl, Wenhao Liu, and Caiming Xiong. 2022. A Generative Language Model for Few-shot Aspect-Based Sentiment Analysis. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 770–787, Seattle, United States. Association for Computational Linguistics.

[5] Aditya Joshi, Vinita Sharma, and Pushpak Bhattacharyya. 2015. Harnessing Context Incongruity for Sarcasm Detection. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 757–762, Beijing, China. Association for Computational Linguistics.

[6] Ibrahim Abu Farha, Steven Wilson, Silviu Oprea, and Walid Magdy. 2022. Sarcasm Detection is Way Too Easy! An Empirical Comparison of Human and Machine Sarcasm Detection. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 5284–5295, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

[7] Tsimenidis S. Limitations of Deep Neural Networks: a discussion of G. Marcus' critical appraisal of deep learning. arXiv preprint arXiv:2012.15754. 2020.

[8] Dilmegani C. Large Language Models: Complete Guide in 2023. AIMultiple. [Updated 2023 June 21]. Available from: https://research.aimultiple.com/large-language-models/. Accessed August 10, 2023.

[9] Diana Maynard and Mark Greenwood. 2014. Who cares about Sarcastic Tweets? Investigating the Impact of Sarcasm on Sentiment Analysis.. In *Proceedings of the Ninth International Conference on Language Resources*

and Evaluation (LREC'14), pages 4238–4243, Reykjavik, Iceland. European Language Resources Association (ELRA).

[10] Santosh Kumar Bharti, Korra Sathya Babu, and Sanjay Kumar Jena. 2015. Parsing-based sarcasm sentiment recognition in twitter data. In Advances in Social Networks Analysis and Mining (ASONAM), 2015 IEEE/ACM International Conference on, pages 1373–1380. IEEE.

[11] Raj Kumar Gupta and Yinping Yang. 2017. CrystalNest at SemEval-2017 Task 4: Using Sarcasm Detection for Enhancing Sentiment Classification and Quantification. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 626–633, Vancouver, Canada. Association for Computational Linguistics.

[12] Ali, R.; Farhat, T.; Abdullah, S.; Akram, S.; Alhajlah, M.; Mahmood, A.; Iqbal, M.A. Deep Learning for Sarcasm Identification in News Headlines. *Appl. Sci.* **2023**, *13*, 5586. https://doi.org/10.3390/app13095586.

[13] Nozza, Debora & Fersini, Elisabetta & Messina, Vincenzina. (2016). Unsupervised Irony Detection: A Probabilistic Model with Word Embeddings. 10.5220/0006052000680076.

[14] Dmitry Davidov, Oren Tsur, and Ari Rappoport. 2010. Semi-Supervised Recognition of Sarcasm in Twitter and Amazon. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, pages 107–116, Uppsala, Sweden. Association for Computational Linguistics.

[15] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I. Attention is all you need. arXiv preprint arXiv:1706.03762. 2017.

[16] Maryam Najafi and Ehsan Tavan. 2022. MarSan at SemEval-2022 Task 6: iSarcasm Detection via T5 and Sequence Learners. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, pages 978–986, Seattle, United States. Association for Computational Linguistics.

[17] Gong, X., Jin, J., & Zhang, T. (2019). Sentiment Analysis Using Autoregressive Language Modeling and Broad Learning System. *2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 1130-1134.

[18] Akshay Khatri and Pranav P. 2020. Sarcasm Detection in Tweets with BERT and GloVe Embeddings. In *Proceedings of the Second Workshop on Figurative Language Processing*, pages 56–60, Online. Association for Computational Linguistics.

[19] *Maciej Hercog, Piotr Jaroński, Jan Kolanowski, Paweł Mieczyński, Dawid Wiśniewski, and Jedrzej Potoniec. 2022. Sarcastic RoBERTa: A RoBERTa-Based Deep Neural Network Detecting Sarcasm on Twitter. In Big Data Analytics and Knowledge Discovery: 24th International Conference, DaWaK 2022, Vienna, Austria, August 22–24, 2022, Proceedings. Springer-Verlag, Berlin, Heidelberg, 46–52. https://doi.org/10.1007/978-3-031-12670-3_4.*

[20] Wikipedia. GPT-3. Available from: https://en.wikipedia.org/wiki/GPT-3. Accessed August 10, 2023.

[21] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, Dario Amodei:
Language Models are Few-Shot Learners. CoRR abs/2005.14165 (2020).

[22] Pezzotti N. GPT-3 for Few-Shot Dialogue State Tracking [dissertation]. Cambridge: University of Cambridge, Department of Engineering; 2021.

[23] Khodak M, Saunshi N, Vodrahalli K. A Large Self-Annotated Corpus for Sarcasm. arXiv preprint arXiv:1704.05579. 2018.

[24] Y. Alex Kolchinski and Christopher Potts. 2018. Representing Social Media Users for Sarcasm Detection. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1115–1121, Brussels, Belgium. Association for Computational Linguistics.

[25] Devamanyu Hazarika, Soujanya Poria, Sruthi Gorantla, Erik Cambria, Roger Zimmermann, and Rada Mihalcea. 2018. CASCADE: Contextual Sarcasm Detection in Online Discussion Forums. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1837–1848, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

[26] Jordan Painter, Helen Treharne, and Diptesh Kanojia. 2022. Utilizing Weak Supervision to Create S3D: A Sarcasm Annotated Dataset. In *Proceedings of the Fifth Workshop on Natural Language Processing and Computational Social Science (NLP+CSS)*, pages 197–206, Abu Dhabi, UAE. Association for Computational Linguistics.

[27] Misra, Rishabh. (2022). News Headlines Dataset For Sarcasm Detection. 10.48550/arXiv.2212.06035.

[28] Santosh Kumar Bharti, Rajeev Kumar Gupta, Nikhlesh Pathik, and Ashish Mishra. 2022. Sarcasm Detection in News Headlines using Voted Classification. In Proceedings of the 2022 Fourteenth International Conference on Contemporary Computing (IC3-2022). Association for Computing Machinery, New York, NY, USA, 208–212. https://doi.org/10.1145/3549206.3549245.

[29] Ibrahim Abu Farha, Silviu Vlad Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, pages 802–814, Seattle, United States. Association for Computational Linguistics.

[30] Robinson R. How to write an effective GPT-3 or GPT-4 prompt. Zapier. 2023 Aug 3. Available from: https://zapier.com/blog/gpt-prompt/. Accessed August 10, 2023.

[31] Jones S. The cost of politeness in a GPT world. MetaMirror. [Publication Date if available]. Available from: https://blog.metamirror.io/the-cost-of-politeness-in-a-gpt-world-3cbc8cb13c39. Accessed August 11, 2023.

[32] Oprea, Silviu & Magdy, Walid. (2019). iSarcasm: A Dataset of Intended Sarcasm.

[33] Thompson AD. ChatGPT. Life Architect. 2022 Dec. Available from: https://lifearchitect.ai/chatgpt/. Accessed August 29, 2023.