NUST

NATIONAL UNIVERSITY
OF SCIENCES & TECHNOLOGY

Machine Learning

Semester Project Document

Sentiment Analysis with Sarcasm and Emoji Awareness

Submitted To:

Dr. Daud Abdullah Asif.

Submitted By:

Muhammad Nabeel	413522
Umar Farooq	406481

Sentiment Analysis with Sarcasm and Emojis

Abstract:

In modern digital communication, use of sarcasm and emojis in text often invert or obscure the true sentiment of the text making sentiment analysis a complicated task. Existing solutions rely on Large Language Models (LLMs) or other methods that suffer from high computational cost having a lot of misclassifications because of a positive text leading to a negative emojis or sarcastic meaning. This paper proposes a lightweight hybrid attention mechanism that incorporates both sarcasm and emojis for sentiment analysis trained on Kaggle IMDB dataset along with custom dataset of sarcastic and emoji enriched sentences using Google Gemini API. The final model is of hybrid deep learning architecture consisting of Bi-directional Long-Term Short-Term Memory (LSTM) with Multi Head Self-Attention.

Introduction:

Modern digital communication relies largely on non-literal, sarcastic and emoji enriched text inverting the actual sentiment of the text making it difficult to identify the sentiment and making it challenging for sentiment analysis tools. For example, a sentence "*Great, another delayed flight!* ✈️🔥 "shows positivity with words but is sarcastically showing frustration and negative in tone, whereas sentiment analysis tools would predict it to be a positive sentiment sentence leading to misclassification. Transformers based models may give better results but the suffer from high computation cost making them impractical for a lot of real-time cases.

Problem Definition:

Problem Statement:

Modern digital communication includes highly sarcastic and emojis based emotions and sentiments. Sentiment Analysis of Sarcastic text with emojis is still a major challenge. Although there are few already present solutions on this project but major of them are computationally expensive and frequently misclassify the text as positive due to the surface-level positivity or sentiment inversion due to emojis. The expected outcome is a lightweight model that achieves a comparable accuracy while being computationally lesser expensive then currently present solutions.

Scope:

Develop a lightweight integrated model that helps in sarcasm detection, emoji sentiment, and text analysis to improve classification accuracy.

Research Challenges:

- **Integrated Modelling:** Most existing solution are focused on detecting either sarcasm or emojis in isolation. For example, an emoji-based sentiment analyser will ignore the sarcastic tone whereas a Sarcasm detector will only detect sarcasm not the emojis.

- **Efficient Attention**: Present mechanisms of attention (e.g., self-attention) too heavy to compute for deployment at the edge.
- **No Simple & Lightweight Solution**: Current Solutions are focused on heavy architectures like transformers and LLMs which come with a requirement of great computational power due to their large number of parameters resulting in high latency and cost.

Our Contribution:

This paper comes up with a Hybrid Deep Learning Approach consisting of Recurrent Neural Networks Specifically Bi-directional LSTMs with Multi-Head Self-Attention mechanism.

Literature Review:

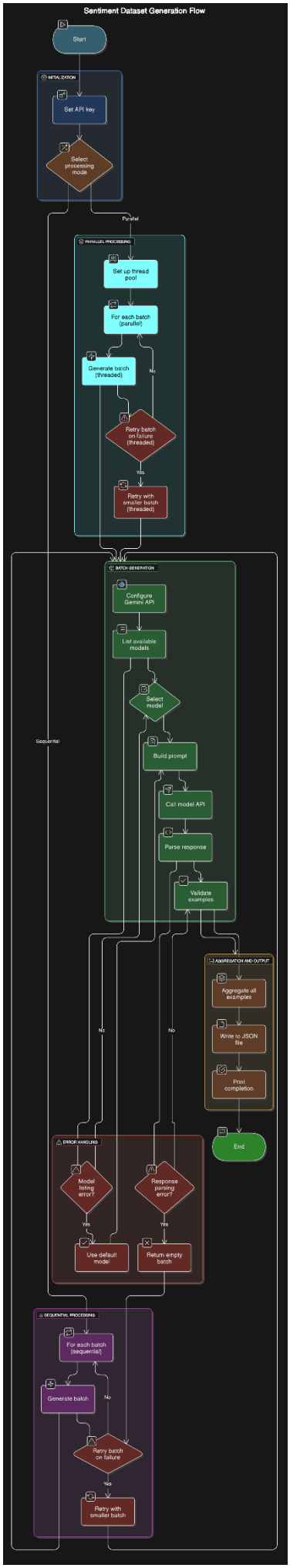
Paper	Approach	Gaps
Alternative Method for Sentiment Analysis Using Emojis and Emoticons (2020)	Emojis and Emoticons are the primary source of sentiment classification and emotion indicator. Majorly Machine learning Techniques (Support Vector Machines and Naive Bayes) are utilized.	Mapping of Sentiment only on emojis ignore the context and does not support Sarcasm in text leading to a major limitation. Static emoji mappings ignore context; no sarcasm detection; outdated for modern social media.
Predicting Multi-Label Emojis, Emotions, and Sentiments in Code-Mixed Texts (2024)	The paper proposes a new dataset SENTIMOJI. Replace self attention layer in transformer encoder with simple linear transformations and use RMS layer norm than normal layer norm. One of the major parts is the Code-Mixed RMS Fourier Transformer (CM-RFT) for prediction of multi label emojis and sentiments which outperforms the ChatGPT in our task.	Focused on multi-label classification, no sarcasm detection and heavy architecture.
Sentiment Analysis of Emoji-Fused Reviews Using Machine Learning and BERT (2025)	Sentiment Analysis of Customer Feedback specifically from social media texts consisting of different emojis. The procedure started with feature extraction (TF-IDF, Word2Vec and BERT Embeddings. Moving towards a transformation models to translate emojis to appropriate representation followed by Data Augmentation and Machine learning Classifiers alongside BERT. THe final model had 92% accuracy outperforming present techniques by 9%.	Does not include Sarcasm Detection and is a computaionally expensive solutions.
On the Impact of Language Nuances on Sentiment Analysis with LLMs (2025)	LLM based improvement in sentiment analysis was used in this paper. Starting from dataset creation of 5929 tweets to assess sarcasm, applying data augmentation, and text paraphrasing techniques to make complete text clearer (by GPT-3.5) improving	Depends on large LLM models requiring high computational cost.

	accuracy by 3-6%. After that Fine tuning of LLMs on the newly created dataset.	
Recent Advancements in NLP-Based Sentiment Analysis (2024)	Review paper on sentiment analysis techniques include Machine Learning based solutions, Deep learning methodologies including LLMs and pre-trained models. Survey paper identifying sarcasm, emojis, and data sparsity as key challenges.	No Emoji or Sarcasm detection techniques are present.
Improving Sentiment Analysis Accuracy with Emoji Embedding (2021)	Liu and others (2021) tackle the task of emotion mining within Chinese social media, where the intricate syntax and informal usage found therein seriously impede the progress of traditional sentiment models. Liu and colleagues evaluated rule-based and classification algorithms on posts from the social media platform Weibo, which is the Chinese counterpart to Twitter. As a first step, they translated the emojis found within these posts to their corresponding sentiment words. Liu and co. did not stop there, however. They enhanced their already impressive featured sets derived from emoji usage patterns by segmenting Weibo posts according to emoji usage and feeding these decisions into a Bidirectional LSTM framework—CEmo-LSTM—that achieved an astounding accuracy of ~0.95 on a dataset of 200k posts from December 1, 2019, to March 20, 2020, with the post-2020 period likely not captured due to the pandemic.	The approach used is based is on Shallow embeddings which lack contextual adaptation thus can result in ignoring of sarcasm when dealing with inverse contexts.
Sarcasm Detection of Emojis Using Machine Learning Algorithms (2024)	This 2024 research utilizes traditional ML classifiers—like SVM, Random Forests, and Naïve Bayes—to work on an emoji-augmented dataset, testing preprocessing pipelines that feature backward elimination of mellowed-strategies for missing emojis in the data. The researchers find that thoughtful selection of features coupled with a strategy for handling the absence of emoji annotations leads to the pipeline that posts the best accuracy in terms of classification, showing that even non-deep-learning pipelines gain a significant boost when it comes to features based on emojis. There's little in the way of specifics regarding the dataset in terms of size and annotation, though, and no comparison is made with deep learning baselines, which limits the direct judgment of	The main concern with this study is that it Fails to model text-emoji conflicts and further it is limited by traditional ML's contextual understanding and does not employ.

	how well the pipeline performs relative to potential alternatives.	
Automated Sarcasm Detection Using CCNN and ELLSTM (2024)	The authors propose a hybrid deep learning model that integrates Convolutional CNN (CCNN) layers for local n-gram feature extraction and a modified LSTM (ELLSTM) for long-range dependency modeling. In other words, this joint model specifically designed for working with text and emoji embeddings is better than its predecessors at sarcasm detection. It focuses on contextually salient tokens and emoji cues that signal sarcasm and utilizes multi-head attention in order to do so. Substantial improvements over traditional 'bag of words' or even preceding neural net models have been reported.	The main issue with this study is that the employment of Dual architectures results in computational expense thus mitigating the fact of employing a light weight solution.
Embracing Emojis in Sarcasm Detection (2024)	This master's thesis develops two tools. One is an Emoji Dictionary, which maps emojis to sentiment scores. The other is a Sarcasm Detection Approach that identifies when textual polarity and accompanying emojis are in conflict. The researchers behind the thesis evaluate the performance of both tools on three case-study datasets—COVID-19 Vaccine, Vegetarianism, and Electric Cars. They find that preprocessing tool performance significantly boosts the F1 scores of both sentiment classification and sarcasm detection.	Static emoji scoring lacks context awareness thus making the system not so effecient when dealing with contextual analysis of emotion detection. Further no efficiency analysis was done in the study.
Sarcasm Detection Using Hybrid Deep Learning with Word-Emoji Embeddings (2024)	Kumar et al. (2023) focus on Hindi tweets containing sarcasm, which is difficult to identify even for humans. Hindi is a morphologically rich, low-resource language. Sarcasm detection in tweets generally relies on deep contextual understanding, which is achieved using extensive resources like large-scale datasets and pretrained models. However, these resources are not available for Hindi and other low-resource languages. The authors propose a workaround that leverages another low resource but morphologically rich language: Arabic. By using Arabic's resources, they create a trained model that nearly reaches the performance level of a state-of-the-art Arabic model on a sarcasm detection task.	Computationally intensive; no dynamic resolution of text-emoji conflicts. The Approach suggested in the study is computationally expensive and further, no dynamic resolution of text-emoji conflicts is discussed in the study.

Key Insights:

- Most papers focus on single modalities (text or emojis) or use heavy architectures (transformers).



- Most papers are focused on a single aspect either emojis, text or sarcasm.
- Present solutions are generally of heavy architectures (LLMs) that require large computational cost.
- Current Lightweight Solutions are either outdated or not good enough to be relied on.
- None address the joint modelling of sarcasm, emojis, and text in a resource-efficient framework.

Methodology:

The solution consists of several phases as discussed below:

Dataset Preparation:

The dataset was prepared by Google Gemini API using a model creation pipeline. Along with custom generated dataset we used a publicly available dataset form Kaggle.

Dataset Generation Pipeline:

Google's Generative AI Model **Gemini's** API was used for generating the dataset consisting of Sentences with Sarcasm and Emojis achieved through Python **google.generativeai** Library. The Key components of the pipeline are given below.

Model Selection:

The scripts used automatically made sure that the most advanced model from the options is used (since Gemini have multiple models with different performance). In case of limited model access, it switches to default model.

Prompt Enigneering:

A properly designed prompt was used for generation of sentences that must had sarcasm, emojis and labels with them. The labels could be wither positive, negative or Neutral.

Sarcasm and Emoji Inclusion:

The prompts properly asks the model to include sarcasm through conflicting tone and context and enhance the tone with emojis.

Multi-threaded Generation:

The dataset was used by multi-threading techniques to generate a large dataset efficiently where each thread makes a separate API call with a fixed batch size.

concurrent.futures.ThreadPoolExecutor, A python library class was used for multi-threading.

In case of failure in generation, batch size was automatically reduced.

IMDB Dataset:

Along with the custom created dataset, IMDB Sentiment Classification dataset was also used. This dataset was merged with our custom generated dataset in order to get different kind of inputs giving more exposure to the model which is a preferred step for generalization purposes.

Link to dataset: https://www.kaggle.com/datasets/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews

Dataset Schema and Format:

The dataset is generated as json text that have a list of individual dataset samples. Each dataset sample consisted of two objects as:

Key	Value
sentence	The sentence having sarcasm and emojis
polarity	{-1,0,1}

Each data point was validated to have both the keys and polarity strictly from any one of {-1,0,1}.

The dataset was exported as JSON file.

Conclusion:

A total of around 41,000 samples were present in the dataset which is a good enough number to train a deep learning-based sentiment classifier. This dataset was in form of JSON file and further pre-processed explain later on in this paper.

Dataset Preprocessing:

Importing the dataset:

The dataset used was in JSON format as a list of sentences with individual polarity labelled with them. Starting with dataset loading and Denoting the labels as in table below:

Label	**Sentiment**
-1	Negative
0	Neutral
1	Positive

No emojis were removed or translated initially, preserving their importance for further steps.

Tokenization and Emoji Embeddings:

Pre-trained tokenizer **bert-base-uncased** was loaded and its vocabulary was extended on our emojis.

Emoji Extraction:

All the unique emojis were extracted from our dataset as a set using Python **emoji** library.

Vocabulary Extension:

The BERT tokenizer was then extended on this set of unique emojis, where each emoji was assigned, a unique token ensuring their representation as a single unit rather then broken down into sub tokens or ignored.

Dataset Construction:

A custom **PyTorch** Dataset class **SentimentDataset** was implemented to handle dataset operations. It consisted of following major steps:

Tokenization:

Each Sentences was tokenized by the custom BERT tokenizer to get the vector representation of dataset for training purposes

Attention mask Generation:

Along with vectors an attention mask was also generated for helping the model on which tokens should be given attention during training.

Label Assignment:

Generation of PyTorch Tensor representation of the Sentiment labels for the model training.

Dataset Splitting and Loading:

The dataset was partitioned into train and test sets as described below:

Train Set	80%
Test Set	20%

The train and test sets were stored as PyTorch **Dataloader** with following configurations:

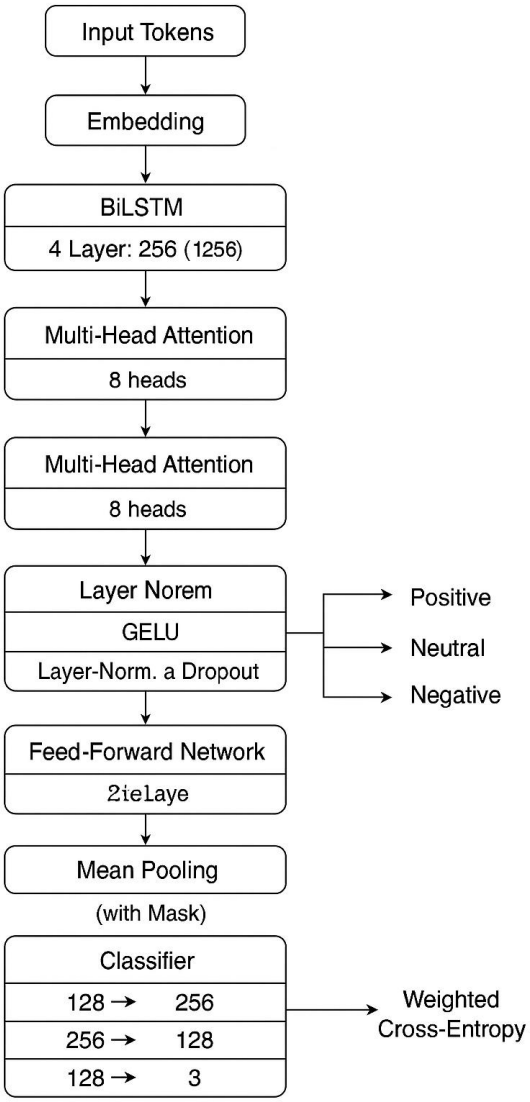
- Batch Size: 16
- Shuffling: Enabled for training Dataset to ensure varied input labels for optimization while training.
- Padding Mask: Handles internally with each batch.

Model Architecture:

The model is based on a hybrid deep learning architecture with RNN and Self- Attention mechanism. The model consists of following layers:

- Embedding Layer.
- Bi-Directional LSTM.
- Layer Normalization.
- Multi-Head Self-Attention.
- Feed Forward Network.
- Global Average Pooling.
- Classification Head.

- Class-Weighted Loss Function.



Embedding Layer:

The model starts with a trainable embedding layer that maps inputs token into as 768-dimensional dense vector. The weights of this layer are initialized randomly and optimized during training.

Bi-Directional LSTM:

Next is a 4-layer Bi-directional LSTM that captures both the forward and backward dependencies in the inputs. To allow the model to retain contextual information, the hidden size is set to 256 per direction. The LSTM captures both the past and future context that is very important to identify the polarity of sentence in case of sarcasm.

LayerNormalization:

Then a Layer Normalization step is applied to stabilize the training and ensure better generalization.

Multi-Head Self-Attention:

The Bi-LSTMs output is then operated by a custom Multi-Head Self Attention module consisting of 8 heads that are responsible to:

- Focus on relevant part of Inputs.
- Focus on multiple relationships in parallel between words and emojis.

Feed Forward Network:

The output of attention module is then processed by a two-layer feed-forward network with GELU activation further abstracting the semantic representation.

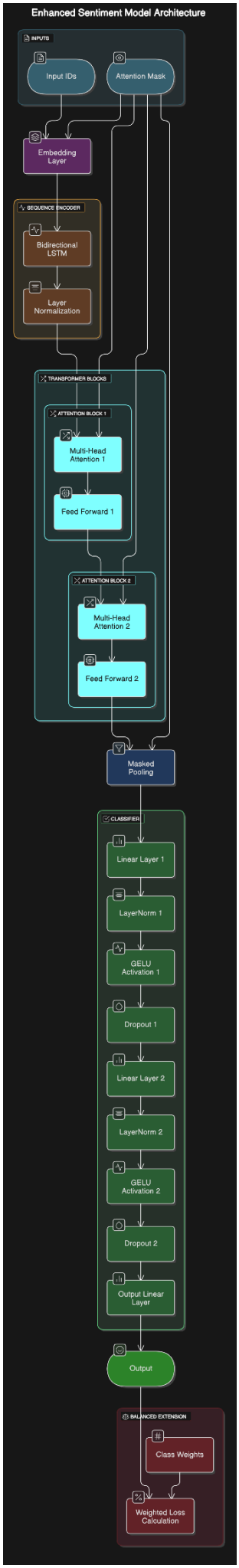
Global Average Pooling:

To reduce output to a fixed size, a global average pooling is performed so the padding tokens in the inputs do not affect the resulting sentiment.

Classification Head:

A muti-layer classification then refines the output of the pooling layer by:

- Linear Layers with LayerNorm and GELU activations.
- Dropout Regularization layer.



- Final Softmax layer generally used for classification purposes for 3 outputs (Positive, Negative or Neutral)

Class-Weighted Loss Function:

The Architecture is extended into a **BalanacedSentimentModel**, to handle class imbalance which computes the weighted cross-entropy loss based on the sentiment classes' distributions.

Model Training:

Optimizer:

AdamW optimizer was used with scheduling learning rate and weight decay.

Batch Size:

32 for Memory Efficiency.

Epochs:

15 epochs based on train and validation loss which may lead to early stopping.

The model was actually trained till 15 epochs but the 11th epoch version was the one chosen for our task since it was lead overfitted and best for generalization purposes.

Loss Function:

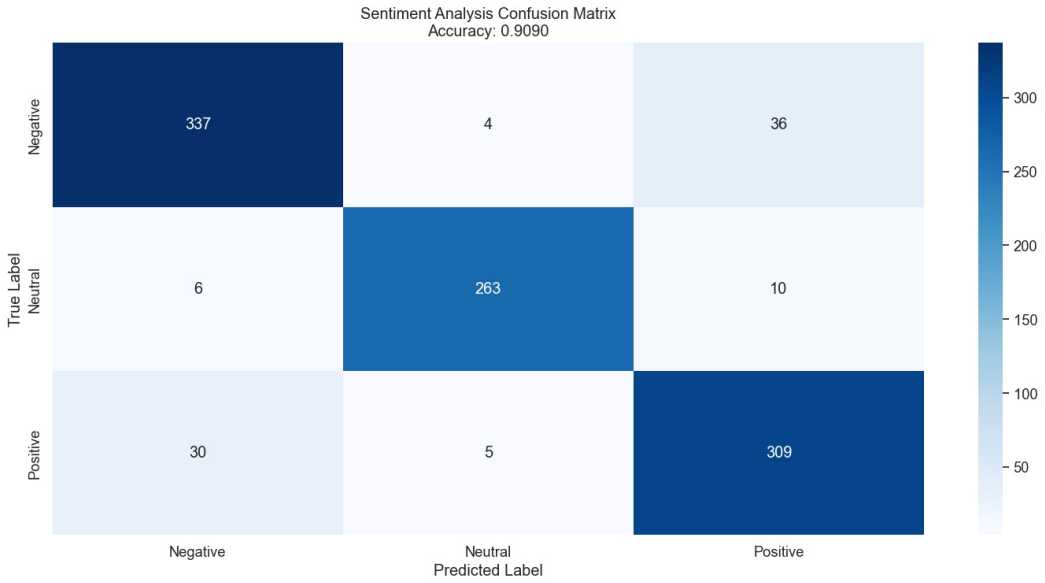
Weighted Cross Entropy Loss Function is used.

Evaluation metrics:

Accuracy, Precision, Recall, F1-Score.

Results:

TO evaluate the performance of our model, we tested it on a set of 1000 samples whose confusion matrix is given below:



The above confusion matrix shows that the model was successful in correct classification of:

- 89.39% of Negative Samples.
- 94.27% of Neutral Samples.
- 89.83% of Positive Samples.

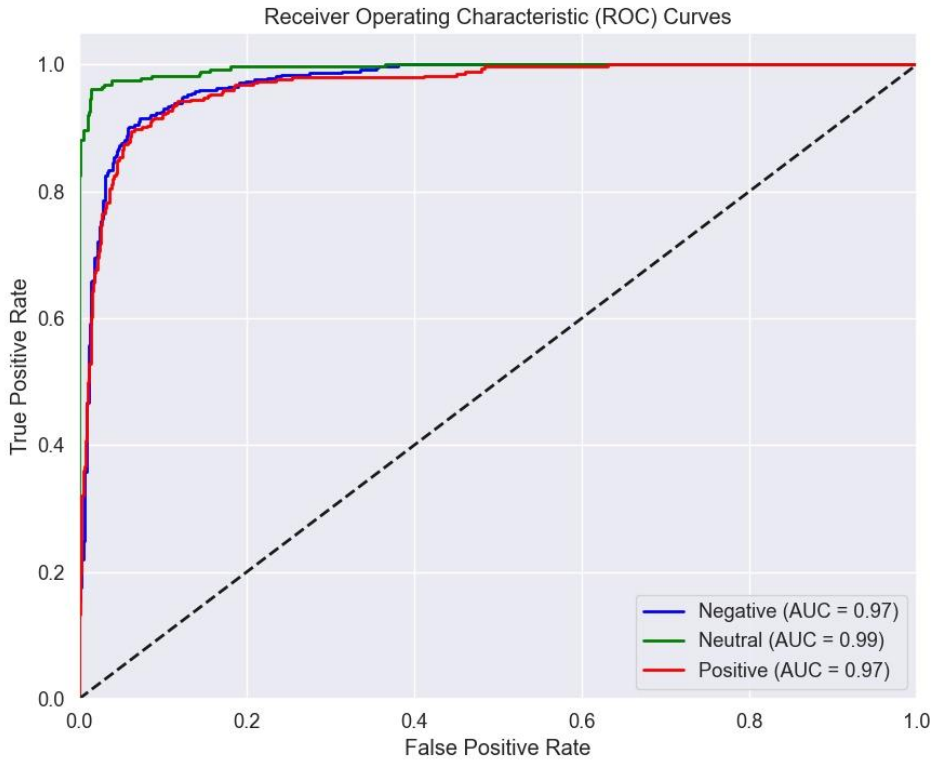
The evaluation metrics and results are given in the below table:

Metrics	Score
Accuracy	90.90 %
Macro F1-Score	0.9125
Weighted F1-Score	0.9093

Class wise Performance:

Class	Precision	Recall	F1-Score	ROC AUC
Negative	0.9235	0.8939	0.8987	0.9715
Neutral	0.9669	0.9427	0.9546	0.9945
Positive	0.8704	0.8983	0.8841	0.9662

The Receiver operating Characteristic Graph (ROC) is given below:



Conclusion:

The Study presents a novel Sentiment Analysis Model designed to effectively analyse the sentiment polarity of textual data consisting of sarcastic tones and emojis, the two elements that are a major challenge to traditional sentiment classification systems. By integration of a LSTM based model with muti-head attention mechanism followed by feed forward layers, the model captures both the contextual dependencies and nuanced emotional cues mostly present in informal language.

Extensive Experimentation on the custom-built dataset was successful in achieving an accuracy of 90.90% with an F1-Score of 0.9125. These results validate the effective of the designed architecture in enhancing the present sentiment analysis solutions which can be lightweight that heavy transformers architectures leading to a high computational cost.

Links and Attachments:

GitHub Repository:

https://github.com/NemroNeno/Sentiment_analysis_Semester_proj.git

Hugging Face Model Endpoint:

https://huggingface.co/mnabeel12/sentiment_analysis/tree/main

Presentation:

https://www.kaggle.com/datasets/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews

Kaggle IMDB Dataset:

https://www.kaggle.com/datasets/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews

References:

Anatoliy Surikov, Evgeniia Egorova, “Alternative method sentiment analysis using emojis and emoticons”, Procedia Computer Science, Volume 178, 2020, Pages 182-193

Gopendra Vikram Singh, Soumitra Ghosh, Mauajama Firdaus, Asif Ekbal and Pushpak Bhattacharyya, “Predicting multi-label emojis, emotions, and sentiments in code-mixed texts using an emojifying sentiments framework”

Amit Khan, Dipankar Majumdar and Bikromadittya Mondal, "Sentiment Analysis of Emoji-Fused Reviews Using Machine Learning and BERT"

Naman Bhargava, Mohammed I. Radaideh, O Hwang Kwon, Aditi Verma, Majdi I. Radaideh, “On the Impact of Language Nuances on Sentiment Analysis with Large Language Models Paraphrasing, Sarcasm, and Emojis”

Jamin Rahman Jim, Md Apon Riaz Talukder, Partha Malakar, Md Mohsin Kabir, Kamruddin Nur, M.F. Mridha, “Recent advancements and challenges of NLP- based sentiment analysis: A state-of-the-art review”

Chuchu Liu, Fan Fang, Xu Lin, Tie Cai, Xu Tan, Jianguo Liu, Xin Lu, “Improving Sentiment Analysis Accuracy with Emoji Embedding”

Yash Tingre, Rohit Pingale, Nirmal Choudhary, Anirudh Kale, Neha Hajare, “Sarcasm Detection of Emojis using Machine learning Algorithm”

Shaikh Ambreen Mohd Ibrahim, Manoj M. Deshpande, Vijaykumar N. Pawar, “Automated Sarcasm Detection in English Tweets Using CCNN and ELLSTM with Text and Emoji Embeddings”

Malak Abdullah Alsabban, Mark Weal, Wendy Hall, “Embracing Emojis in Sarcasm Detection to Enhance Sentiment Analysis”

Vidyullatha Sukhavasi, Venkatesulu Dondeti, “Sarcasm Detection using Hybrid Deep Learning framework based on Word-Emoji Embeddings”