Submitted to

**The University of Roehampton**

In partial fulfilment of the requirements

for the degree of

**Master of Science**

**in**

**Data Science**

By

ASWIN PRAKASH THEKKEPURAKKAL PRAKASHAN

THE23624623

Sentiment Analysis of Airline Twitter Data Using Deep Learning for Brand Reputation Management

Declaration

I hereby certify that this report constitutes my own work, that where the language of others is used, quotation marks so indicate, and that appropriate credit is given where I have used the language, ideas, expressions, or writings of others.

I declare that this report describes the original work that has not been previously presented for the award of any other degree of any other institution.

Aswin Prakash Thekkepurakkal Prakashan

05/05/2025

Aswin Prakash Thekkepurakkal Prakashan

Acknowledgements

I would like to express my sincere gratitude to my project supervisor for their guidance, feedback, and encouragement throughout this research journey. Their support has been vital in helping me complete this work successfully.

I also wish to thank my lecturers and academic staff at the university for providing the knowledge and skills needed to carry out this project.

Lastly, I am thankful to my family and friends for their patience, motivation, and emotional support during my studies. Their belief in me kept me going, even during challenging times.

Abstract

The rise of social media platforms such as Twitter has revolutionised the way customers express feedback, particularly in the airline industry where real-time opinions can directly influence brand perception. This project explores the application of deep learning methods to automate sentiment classification of airline-related tweets. The core problem addressed was the need to accurately detect customer sentiments negative, neutral, or positive on social media, using natural language processing (NLP) techniques. A key motivation was the challenge of capturing contextual nuances, sarcasm, and informal language patterns in tweets.

The literature review covered foundational sentiment analysis techniques, starting from traditional machine learning methods like Naive Bayes and Support Vector Machines, to more recent advances in deep learning using Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM) networks, and transformer-based models such as BERT. Past research has shown the effectiveness of both LSTM and BERT, but comparative studies using the Twitter US Airline Sentiment Dataset remain limited, especially with a focus on implementation challenges and practical performance differences. The gap identified was the lack of studies that balanced contextual understanding, model accuracy, and resource efficiency in sentiment classification tasks for short social media texts.

The methodology followed a structured approach. After loading the dataset, extensive preprocessing was performed including removal of URLs, mentions, numbers, punctuation, and stopwords, followed by stemming. Exploratory data analysis revealed that 63% of tweets were negative, with common complaints about customer service and delays. A detailed visual analysis using word clouds, length histograms, and sentiment distributions provided insights into the nature of tweet content.

Two deep learning models were implemented and evaluated: an LSTM model and a fine-tuned BERT model. The LSTM model used an embedding layer followed by a bidirectional LSTM and softmax classifier. It achieved a test accuracy of 78%, performing particularly well on negative sentiment due to its ability to capture sequential patterns in longer tweets. BERT, based on the transformer architecture, was fine-tuned using the bert-base-uncased model and showed superior performance with an accuracy of 80% and higher F1-scores across all classes, especially in distinguishing neutral and positive sentiments. However, BERT required significantly more computational resources and training time.

The evaluation included confusion matrices, classification reports, and training-validation accuracy/loss curves. While BERT outperformed LSTM overall, LSTM was faster and more hardware-efficient, making it suitable for environments with limited resources. The comparison highlighted a key trade-off between performance and efficiency.

This study contributes to sentiment analysis literature by offering a clear, reproducible comparison of LSTM and BERT for tweet classification. It also demonstrates that while BERT excels in contextual understanding, LSTM remains valuable in practical, resource-constrained settings. Future work could include hybrid models, sarcasm detection, and incorporating multimodal data to further improve sentiment classification accuracy and relevance.

Table of Contents

[Declaration ii](#_Toc197393684)

[Acknowledgements iii](#_Toc197393685)

[Abstract iv](#_Toc197393686)

[Table of Contents vi](#_Toc197393687)

[List of Figures ix](#_Toc197393688)

[List of Tables x](#_Toc197393689)

[Chapter 1 Introduction 1](#_Toc197393690)

[1.1 Problem Description, Context and Motivation 1](#_Toc197393691)

[1.2 Objectives 1](#_Toc197393692)

[1.3 Methodology 2](#_Toc197393693)

[1.3.1 Dataset Selection 2](#_Toc197393694)

[1.3.2 Data Preprocessing 2](#_Toc197393695)

[1.3.3 Model Development 2](#_Toc197393696)

[1.3.4 Evaluation 2](#_Toc197393697)

[1.4 Legal, Social, Ethical and Professional Considerations 3](#_Toc197393698)

[1.4.1 Legal Considerations 3](#_Toc197393699)

[1.4.2 Social Considerations 3](#_Toc197393700)

[1.4.3 Ethical Considerations 3](#_Toc197393701)

[1.4.4 Professional Considerations 3](#_Toc197393702)

[1.5 Background 3](#_Toc197393703)

[1.6 Structure of Report 5](#_Toc197393704)

[Chapter 2 Literature – Technology Review 6](#_Toc197393705)

[2.1 Literature Review 6](#_Toc197393706)

[2.2 Technology Review 6](#_Toc197393707)

[2.3 Summary 8](#_Toc197393708)

[Chapter 3 Implementation 10](#_Toc197393709)

[3.1 System Design and Architecture Overview 10](#_Toc197393710)

[3.2 Data Preprocessing 11](#_Toc197393711)

[3.3 Tools and Libraries Used 12](#_Toc197393712)

[3.4 LSTM-Based Model Development 13](#_Toc197393713)

[3.4.1 Preprocessing and Tokenisation 13](#_Toc197393714)

[3.4.2 Model Architecture 14](#_Toc197393715)

[3.4.3 Training Configuration 14](#_Toc197393716)

[3.4.4 Step-by-Step Algorithm 14](#_Toc197393717)

[3.4.5 Mathematical Representation of LSTM 14](#_Toc197393718)

[3.4.6 Model Summary Output 15](#_Toc197393719)

[3.5 BERT-Based Model Development 16](#_Toc197393720)

[3.5.1 Tokenisation with BertTokenizer 16](#_Toc197393721)

[3.5.2 Model Architecture 16](#_Toc197393722)

[3.5.3 Training Configuration 16](#_Toc197393723)

[3.5.4 Step-by-Step Algorithm 16](#_Toc197393724)

[3.5.5 Mathematical Explanation of Self-Attention in BERT 17](#_Toc197393725)

[3.5.6 Performance and Advantages 17](#_Toc197393726)

[3.6 Training and Evaluation Process 17](#_Toc197393727)

[3.6.1 LSTM Model Evaluation 18](#_Toc197393728)

[3.6.2 BERT Model Evaluation 18](#_Toc197393729)

[3.6.3 Evaluation Metrics Used 18](#_Toc197393730)

[3.6.4 Model Comparison 19](#_Toc197393731)

[3.7 Implementation Challenges and Resolutions 19](#_Toc197393732)

[Chapter 4 Evaluation and Results 21](#_Toc197393733)

[4.1 Related Works 21](#_Toc197393734)

[4.2 Dataset Evaluation and Sentiment Insights 21](#_Toc197393735)

[4.3 LSTM Model Evaluation 26](#_Toc197393736)

[4.4 BERT Model Evaluation 30](#_Toc197393737)

[4.5 Model Comparison and Discussion 32](#_Toc197393738)

[4.6 Strengths, Weaknesses, and Possible Improvements 33](#_Toc197393739)

[Chapter 5 Conclusion 35](#_Toc197393740)

[5.1 Future Work 35](#_Toc197393741)

[5.2 Reflection 37](#_Toc197393742)

[References 39](#_Toc197393743)

[Appendices I](#_Toc197393744)

[Appendix A: Project Proposal II](#_Toc197393745)

[Appendix B: Project Management XVI](#_Toc197393746)

[Appendix C: Artefact/Dataset XVII](#_Toc197393747)

[Appendix D: Screencast XXXIX](#_Toc197393748)

List of Figures

[Figure 1: System Architecture 10](#_Toc197393498)

[Figure 2: Distribution of Sentiments 22](#_Toc197393499)

[Figure 3: Tweet Length Distribution by Sentiment 24](#_Toc197393500)

[Figure 4: Word Cloud by Negative Tweets 25](#_Toc197393501)

[Figure 5: Word Cloud by Positive Tweets 25](#_Toc197393502)

[Figure 6: Number of Tweets by Hour 26](#_Toc197393503)

[Figure 7: Confusion Matrix (LSTM Model) 28](#_Toc197393504)

[Figure 8: Training and Validation Accuracy (LSTM) 29](#_Toc197393505)

[Figure 9: Training and Validation Loss (LSTM) 30](#_Toc197393506)

[Figure 10: Confusion Matrix (BERT Model) 32](#_Toc197393507)

List of Tables

[Table 1: Top 10 Reasons for Negative Tweets 22](#_Toc197440614)

[Table 2: Classification Report for LSTM Model 27](#_Toc197440615)

[Table 3: Classification Report for BERT Model 31](#_Toc197440616)

# Introduction

## Problem Description, Context and Motivation

Social media websites are important for customer engagement and brand reputation [1]. Twitter is a major platform in the formation of airlines' public image. Twitter is used by customers to share their satisfaction, dissatisfaction, and complaints about services [2]. Analysing tweets directed at airlines gives important customer sentiment information. Tracking customer comments, analysing sentiment trends, and resolving problems in real time can assist in ensuring a good public reputation [2]. Social media helps businesses engage with customers and build brand reputation. The airline industry is very vulnerable to public opinion; a single complaint going viral can ruin reputations. Airlines like Delta, United, and American are widely spoken about on Twitter, and thus it is very important to monitor brand reputation [3]. The live tweets on Twitter allow customers to give immediate feedback and hence make it necessary for airlines to employ sentiment analysis for quick responses and service improvement [2,4]. The increasing use of artificial intelligence (AI) and deep learning in natural language processing (NLP) has created opportunities for automating sentiment classification, which reduces the reliance on manual monitoring [5].

The primary motivation of this study is the increasing importance of social media sentiment analysis. Airlines encounter real-time analysis challenges because of the huge amount of data on social media [2]. Traditional approaches, such as lexicon-based methods, fail to capture the subtleties of human language. Moreover, negative sentiment propagates rapidly on the internet, impacting consumer confidence, ticket sales, and share prices [6]. There are deep learning algorithms like LSTM and BERT that provide effective solutions to sentiment classification in real time [7,8]. In this study, a deep learning sentiment analysis model is used to provide higher accuracy and efficiency in customer sentiment classification, which will help airlines manage brand reputation.

## Objectives

1. To collect and analyse airline-related tweets for sentiment classification. This involves utilizing the Twitter US Airline Sentiment Dataset, which includes customer feedback directed at major airlines.
2. To develop a deep learning-based sentiment analysis model. The study will implement models such as LSTM and BERT to classify tweets as positive, negative, or neutral.
3. To evaluate deep learning models with traditional sentiment analysis techniques. The study will evaluate the accuracy, precision, recall, and F1-score of deep learning models against classical machine learning models such as Support Vector Machines (SVM) and Naïve Bayes.

## Methodology

The research follows a structured methodology to achieve the stated objectives. The methodology comprises five key steps: dataset selection, data preprocessing, model development, evaluation, and application of insights.

### Dataset Selection

The study will use the Twitter US Airline Sentiment Dataset, a publicly available dataset on Kaggle. This dataset consists of over 14,000 tweets directed at major U.S. airlines, labelled as positive, negative, or neutral [5]. The dataset provides a real-world representation of customer sentiment in the airline industry, making it ideal for sentiment analysis research.

### Data Preprocessing

Data preprocessing is crucial for improving model performance. The following preprocessing techniques will be applied:

1. Removing stopwords, punctuation, and special characters to eliminate noise from the text.
2. Tokenization and stemming to break text into meaningful words and reduce words to their root form.
3. Handling imbalanced data by applying oversampling or under sampling techniques if necessary.

### Model Development

Deep learning models such as LSTM and BERT will be implemented for sentiment classification. The models will be trained on labelled data and fine-tuned to enhance performance. LSTM will capture sequential dependencies in text, while BERT will leverage bidirectional context to improve sentiment understanding [6].

### Evaluation

The developed models will be evaluated using standard NLP classification metrics, including accuracy, precision, recall, and F1-score. The performance of deep learning models will be compared with traditional machine learning classifiers such as SVM and Naïve Bayes.

## Legal, Social, Ethical and Professional Considerations

### Legal Considerations

This research adheres to data privacy regulations, including the General Data Protection Regulation (GDPR) and Twitter’s API policies. Since the dataset is publicly available, there are minimal privacy concerns. However, ethical considerations must be maintained in data handling and usage.

### Social Considerations

Sentiment analysis impacts customer service strategies in the airline industry. Automated sentiment classification can help airlines improve passenger experiences and respond to service complaints effectively.

### Ethical Considerations

Bias in sentiment analysis models can lead to unfair or inaccurate classification. To mitigate bias, the study will use diverse training datasets and apply fairness-aware AI principles.

### Professional Considerations

Findings from this study should assist airlines in decision-making while maintaining ethical AI practices. Sentiment analysis should complement human judgment rather than replace it entirely.

## Background

Opinion mining, or sentiment analysis, is a computational technique that uses natural language processing (NLP) and machine learning to extract and categorize sentiments from text data. Sentiment analysis enables businesses, policymakers, and researchers to quantify public sentiment, providing them with crucial insights into consumer behaviour and trends. Sentiment analysis in social media allows organizations to quantify customer sentiments around products, brands, and events and is key to marketing and strategy [9]. Companies can utilize computational models to measure consumer sentiment, detect emerging trends, and take proactive measures regarding customer grievances [10]. Further, sentiment analysis is essential for crisis management since companies can detect negative sentiment even before it turns into a catastrophe. Deriving knowledge from user opinions is very important for brand image and customer relationship management [11].

Early sentiment analysis methods depended on lexicon-based and rule-based approaches, which used predefined word lists and syntactic rules to determine sentiment. These approaches excelled at simple tasks but had problems handling context-dependent sentiment expressions and sarcasm [12]. Machine learning enhanced sentiment classification by allowing models to learn from labeled data instead of relying on word lists. Traditional classifiers like SVM and Naïve Bayes offered improved generalization and domain adaptability, transitioning from rule-based manual methods to data-driven learning and decreasing reliance on specific lexicons [13].

Deep learning led to a paradigm shift in sentiment analysis with Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks outperforming traditional approaches. Deep learning-based models improved the task of sentiment classification by modelling complex dependencies within text data and utilizing word embeddings to interpret the context [14]. The usage of transformer models like BERT has improved the performance of sentiment classification through learning the subtleties of language in social media [15]. Deep learning-based models transformed sentiment analysis by encoding contextualized meanings in text and enhancing classification performance. Unlike machine learning methods, deep learning models do not need extensive feature engineering since they can learn hierarchical representations from raw text data [16].

Recurrent Neural Networks (RNNs) are widely used in sentiment analysis since they are suitable for handling sequential text data. They suffer from vanishing gradients that make capturing long-range dependency difficult. To address this, Long Short-Term Memory (LSTM) networks were designed to solve it using gate control mechanisms to retain information in longer sequences to improve the effectiveness of sentiment classification [17]. CNNs, originally for image processing, are being applied to text classification using local feature extraction. They work well for sentiment analysis as they can pick up important phrases associated with sentiment polarity. They do not, however, handle sequential dependencies as compared to RNN models [18].

BERT and GPT models have significantly advanced sentiment analysis with self-attention models that represent global text dependencies. They outperform traditional approaches by considering the context of words and enabling transfer learning, making them effective for sentiment classification tasks [19]. Deep learning models are more precise than traditional approaches but need much computational power and large data. On the other hand, models like Support Vector Machines (SVM) and Naïve Bayes are more appropriate for small sentiment analysis tasks since they have lower resource demands [20]. Social networking websites offer plenty of sources of consumer opinions, allowing companies to analyse customers' sentiments in real time [21]. Some companies have integrated sentiment analysis into their reputation management system. For instance, major airlines use sentiment analysis to track passenger satisfaction and solve complaints pro-actively. Airlines track social media messages to recognize service issues and improve [22]. Real-time sentiment tracking allows firms to react quickly to crises. For instance, negative social media spikes may signal PR issues, prompting timely responses to reduce reputational harm. Advanced models improve crisis management through the detection of sentiment shifts and the prediction of backlash [23].

## Structure of Report

1. **Chapter 1: Introduction** – Defines the problem, objectives, methodology, and background.
2. **Chapter 2: Literature Review** – Reviews existing sentiment analysis research and methods.
3. **Chapter 3: Implementation** – Details dataset selection, model development, and evaluation.
4. **Chapter 4: Evaluation and Results** – Presents findings from sentiment analysis.
5. **Chapter 5: Conclusion** – Summarizes key takeaways of the research analysis.

# Literature – Technology Review

## Literature Review

Social media platforms like Twitter have become critical channels for consumers to voice opinions, making sentiment analysis an important tool for businesses​ [24]. In the airline industry, millions of tweets reflect passenger experiences, ranging from praise to complaints, directly impacting public perception of an airline’s brand. Companies increasingly leverage sentiment analysis to gauge customer satisfaction and manage brand reputation​ [25]. By automatically classifying tweets as positive, negative or neutral, airlines can monitor public sentiment in real-time, identify emerging issues, and address negative feedback promptly to protect their brand image​ [26]. This section reviews the technical advancements in applying deep learning, especially Long Short-Term Memory (LSTM) networks and Bidirectional Encoder Representations from Transformers (BERT) for sentiment analysis on airline-related Twitter data. It also highlights the performance of these models, compares them with traditional approaches, and discusses key challenges, limitations, and future research directions in this domain.

## Technology Review

Natural language processing (NLP) has evolved from basic keyword spotting and lexicon-based methods to machine learning and now deep learning approaches. Traditional machine learning classifiers (e.g. Support Vector Machines, Naïve Bayes) with handcrafted features or bag-of-words representations have been widely used for sentiment analysis, but they often struggle with the nuanced and context-dependent nature of language​ [27]. Deep learning models have gained popularity because of their ability to automatically learn feature representations and capture sequential context in text data​ [27]. Recurrent neural networks, particularly LSTMs, introduced the ability to remember long-term dependencies in sentence structure, which is crucial for understanding sentiment in longer or complex tweets. LSTM-based models incorporate word embeddings (dense vector representations of words like Word2Vec or GloVe) to capture semantic relationships and context beyond what n-gram features can provide​ [27]. The recent introduction of transformer-based models has been a breakthrough in NLP​. Unlike RNNs, transformers (such as BERT) use self-attention mechanisms to process an entire sentence simultaneously, enabling them to capture complex word relationships and contextual meaning from both preceding and following words. This bidirectional context understanding gives transformers a significant advantage in interpreting sentiment. For example, BERT will assign different contextual embeddings to the word “delay” in “flight delay was terrible” versus “delay the flight if weather is bad,” whereas a standard LSTM might not distinguish context as effectively​ [28].

LSTM networks have been a cornerstone of deep learning for sentiment analysis on Twitter data. Their gated memory cells allow them to maintain context over long sequences and mitigate the vanishing gradient problem that vanilla RNNs face. In practice, LSTM-based models have shown strong performance in classifying tweet sentiments. Researchers often employ an embedding layer (using pre-trained word embeddings like GloVe) feeding into one or more LSTM layers, sometimes with dropout regularization and fully connected output layers, to predict sentiment classes. Applied to airline Twitter data, such architectures have achieved high accuracy. For instance, a recent study on an airline customer feedback dataset found that an LSTM model outperformed other deep learning models (including CNNs, GRUs, and even BERT) with about **91%** classification accuracy​ [29]. This indicates that when provided sufficient data and tuning, LSTMs can effectively learn the language patterns of praise or complaints in tweets. In many cases, LSTM-based approaches also outperform traditional machine learning classifiers on Twitter sentiment tasks​ [27]. One publication reported an LSTM achieving around **82%** accuracy on the popular Twitter US Airline sentiment dataset, notably higher than the accuracies achieved by classical models like SVM or Naïve Bayes on the same data​ [27]. The sequential modelling capability of LSTMs enables them to catch subtle cues in informal text (for example, negation or tone indicated by preceding words) that simpler models might miss.

Transformer models like BERT have rapidly become the state-of-the-art for sentiment analysis due to their superior ability to understand context. BERT is a pre-trained language model that was trained on enormous text corpora and then fine-tuned for specific tasks like sentiment classification. Unlike LSTMs which process sequences step by step, BERT’s transformer encoder looks at the entire tweet at once through multi-head attention, considering all words in parallel. This enables BERT to grasp the meaning of a word based on both its left and right context. As a result, BERT can disambiguate words and capture sarcasm or negation more reliably than earlier models. One technical advantage is that BERT will produce different vector representations for the same word used in different contexts (e.g., “bad” in a phrase indicating a negative experience versus in “not bad” indicating a positive sentiment), whereas an LSTM relying only on preceding context might misinterpret these​ [28]. This contextual sensitivity often leads to improved sentiment classification performance.

Empirical results have indeed shown BERT’s effectiveness on Twitter sentiment tasks. For example, a study analysing COVID-19 related tweets compared an LSTM model with a fine-tuned BERT model and found that BERT significantly outperformed the LSTM, with accuracy improvements of 5–15% across various tweet topics​ [30]. The BERT model achieved around 85–92% accuracy on different subsets of the COVID tweet data, whereas the LSTM peaked around 76–81%​ [30]. This gap illustrates how BERT’s deeper language understanding yields more accurate sentiment predictions. Other research has likewise highlighted the superior performance of BERT in sentiment analysis tasks​ [27]. In many benchmark evaluations (for example, product or movie review sentiment datasets), fine-tuned transformers now set the highest accuracy scores, often several points above RNN-based models. On airline Twitter data, transformers have also shown excellent results. One approach combined BERT’s embeddings with a BiLSTM classifier and reported an accuracy over 93% in classifying tweets into positive, negative, and neutral​ [28].

## Summary

Ongoing research is addressing the above challenges and exploring new directions to enhance sentiment analysis on social media. One active area is the integration of auxiliary tasks like sarcasm detection or emotion recognition alongside sentiment classification. Multi-task learning frameworks have been proposed where a model is jointly trained to identify sarcastic tone and sentiment, with the goal of improving overall accuracy​ [30, 31]. Early results indicate that when a model “knows” to flag sarcasm, it can adjust its sentiment prediction accordingly, thereby handling sarcastic airline tweets more effectively. Future models might routinely incorporate sarcasm detection or use metadata (such as whether a tweet contains a sarcasm-related hashtag or emoji) to inform sentiment decisions.

Another promising direction is the development of hybrid models that combine the strengths of different architectures. For example, researchers have experimented with models that feed transformer-based embeddings (like BERT or its variants) into LSTM or GRU layers, aiming to capture both global context and sequential dependencies​. Such hybrid models have yielded very high accuracies; one study reported a RoBERTa–BiLSTM hybrid achieving about 91.3% accuracy on the airline tweet dataset, outperforming either model alone​ [27]. This suggests that there is still room to improve by creatively fusing architectures. In the future, we may see more combinations (e.g., CNNs for local phrase detection feeding into transformers, or ensemble approaches blending BERT with simpler classifiers) to boost robustness. Additionally, as transformer models are resource-intensive, there is ongoing work on model distillation and optimization to make them faster and more feasible for real-time monitoring of social media streams – an important consideration for airlines responding to viral issues or crises. Lighter distilled models or efficient architectures (like DistilBERT or TinyBERT) could deliver near BERT-level performance with less computational cost, enabling deployment in live systems that track brand sentiment continuously.

Beyond text-only analysis, a notable gap in current airline sentiment analysis is the lack of multimodal integration​ [25]. Passengers often share images (photos of cramped seating, meal pictures, etc.) or videos along with text in their tweets. These could carry sentiment information (a photo of a mishandled bag strongly signals a negative experience) that pure text analysis might miss. Future sentiment analysis systems might incorporate computer vision to analyse accompanying images or use metadata (like the number of likes/retweets a complaint tweet gets) to weight the impact of each item. Incorporating multiple data modalities and sources can provide a more holistic view of public sentiment. For instance, combining Twitter data with Facebook comments or Airline quality review sites could improve coverage of brand sentiment. Current research highlights this as an open area, noting a lack of comprehensive studies that fuse text with other data sources for sentiment in real-time contexts​ [25].

# Implementation

## System Design and Architecture Overview

The architecture of this research project is built around a multi-step deep learning pipeline designed for sentiment analysis of airline-related tweets. The system begins by loading the Twitter US Airline Sentiment Dataset, which includes thousands of real tweets directed at major US airlines. These tweets form the raw input for the entire system. From there, the data moves through a preprocessing phase that involves cleaning, tokenising, and transforming the text into a usable format. After this stage, the text is converted into sequences suitable for input into the deep learning models. These models Long Short-Term Memory (LSTM) and Bidirectional Encoder Representations from Transformers (BERT) are then trained on the processed data to learn how to classify the sentiment of tweets. Finally, the performance of each model is evaluated using accuracy, precision, recall, F1-score, and visualisations such as confusion matrices.

A diagram of a program

AI-generated content may be incorrect.

Figure : System Architecture

The key tools used in this project are as follows: Python as the main programming language; Pandas and NumPy for data manipulation; NLTK for natural language processing tasks like tokenisation and stopword removal; TensorFlow and Keras for implementing the LSTM model; Hugging Face Transformers for BERT; and Scikit-learn for evaluation metrics such as classification reports and confusion matrices. Visualisations were done using Matplotlib and Seaborn libraries to generate informative plots for sentiment distribution and performance analysis.

Deep learning methods such as LSTM and BERT were chosen over traditional approaches like Support Vector Machines (SVM) and Naïve Bayes because they offer better performance in handling the complexities of natural language. Traditional methods often depend on handcrafted features and fail to understand context or word order properly. In contrast, deep learning models can automatically learn feature representations and retain sequential information (LSTM) or even capture the meaning of a word in its bidirectional context (BERT), which improves sentiment classification accuracy [32].

## Data Preprocessing

Data preprocessing is a crucial step in building any text classification system because it helps in removing noise and preparing the data for model consumption. For this study, the **Twitter US Airline Sentiment Dataset** was used. This dataset is publicly available on Kaggle and consists of over 14,000 tweets, each labelled as positive, negative, or neutralbased on customer opinions about different airlines like United, Delta, and American Airlines. Each tweet also includes metadata such as the tweet ID, the airline name, the reason for negative sentiment (if applicable), and the time of creation [33].

The first step in the preprocessing pipeline involved **cleaning the tweet text**. This included removing URLs, usernames (mentions), hashtags, numbers, and special characters. Such elements do not contribute to sentiment meaning and can introduce noise that affects model accuracy. Regular expressions (regex) were used for pattern matching and removal. Tweets were then converted to lowercase to ensure uniformity, as "Bad" and "bad" should be treated the same during training.

After cleaning, **stopwords were removed** using NLTK’s stopword list. Stopwords are commonly used words like “is”, “and”, and “the”, which do not carry significant meaning and can dilute the impact of important words in the sentence. Removing them helps the model focus on sentiment-bearing words. This was followed by tokenisation, where each sentence was broken down into individual words or tokens. A stemming process was also applied using the Porter Stemmer algorithm, which reduces words to their root form (e.g., "delayed" becomes "delay"). This reduces word redundancy and improves generalisation during training [34].

Another essential step involved handling class imbalance. An initial analysis of sentiment distribution showed that negative tweets were more frequent than positive or neutral ones. This could bias the model towards predicting negative sentiment. To address this, a stratified split was used when dividing data into training and test sets, ensuring all classes were represented proportionally. Future work may involve applying more advanced balancing techniques like SMOTE or class weighting to further address this issue [35].

To understand the data better, several visualisation techniques were applied. A count plot showed that most tweets were negative, which aligns with the natural tendency of users to complain more than praise on platforms like Twitter. Additionally, a word cloud was generated for each sentiment class. The word cloud for positive tweets contained terms like “great”, “thanks”, and “amazing”, while negative ones included “delayed”, “lost”, and “worst”. These visuals helped confirm the dataset's reliability and offered early insight into frequently used expressions of sentiment.

A histogram of tweet lengths was also created to show how long users’ tweets typically are. It was observed that neutral tweets were often shorter and more factual, while negative ones tended to be longer, possibly because users were explaining their dissatisfaction in detail. This observation can be important when setting padding lengths during sequence preparation for models like LSTM and BERT [36].

The preprocessing stage helped transform messy and inconsistent tweet text into a structured and clean format, making it suitable for deep learning models. The processed data retained sentiment-rich expressions while removing noise and inconsistencies, ensuring the models could focus on learning meaningful patterns. This step is foundational to the success of the sentiment analysis system and directly influences the model’s performance.

## Tools and Libraries Used

To implement the sentiment analysis system, several tools and libraries were used, each playing a specific role in the processing, modelling, and evaluation pipeline. The entire implementation was developed using **Python**, which is widely used in data science and machine learning for its simplicity and the vast ecosystem of libraries available.

For deep learning, **TensorFlow** and **Keras** were employed. Keras, being a high-level API running on top of TensorFlow, simplified model construction and training. It was used for building the LSTM model, defining layers, and compiling the architecture. Keras also provided useful utilities for tokenisation and padding of text data [36].

The **Hugging Face Transformers** library was used to load and fine-tune the pre-trained BERT model. This library offered a simple interface to access state-of-the-art transformer architectures and was essential for leveraging BERT’s contextual understanding of text [37].

**NLTK (Natural Language Toolkit)** supported the preprocessing stage. It was used for stopword removal, stemming, and tokenisation. These steps helped standardise and clean the text before training [38].

Visualisation of data and training results was performed using **Matplotlib** and **Seaborn**, which allowed clear plotting of sentiment distributions, word clouds, and model performance metrics. For evaluating model accuracy, precision, recall, F1-scores, and confusion matrices, **Scikit-learn** provided essential functions [39]. This combination of tools ensured a robust and flexible development environment suited to both traditional and modern NLP techniques.

## LSTM-Based Model Development

The Long Short-Term Memory (LSTM) network was selected as one of the primary deep learning models for this research due to its proven strength in handling sequential data, such as text. LSTM is an improved type of Recurrent Neural Network (RNN) that overcomes the vanishing gradient problem by using memory cells to store information over long time steps [40].

### Preprocessing and Tokenisation

After the text data had been cleaned, tokenisation and sequence padding were applied. Tokenisation involved converting each processed tweet into a list of tokens (words). A Keras **Tokenizer** was configured with a vocabulary size of 5,000 most frequent words and an out-of-vocabulary (OOV) token for handling unseen words. This tokenizer was fitted on the training dataset and used to transform both training and testing tweets into sequences of integers.

The next step was **padding**. Since LSTM networks expect input sequences of equal length, all sequences were padded (or truncated) to a maximum length of 100 tokens. This step ensured uniform input shapes and prevented shape mismatch during model training.

### Model Architecture

The LSTM model was built using Keras's Sequential API. It began with an **Embedding layer** with input dimension 5,000 and output dimension 64. This layer converted integer-encoded words into dense vectors, which allowed the network to learn word relationships during training.

Next was a **Bidirectional LSTM layer** with 64 units. This allowed the model to process input sequences in both forward and backward directions, thereby capturing dependencies from both past and future contexts in the tweet. This feature is particularly useful in sentiment analysis, where the meaning of a word can depends on both its previous and following words [41].

Following the LSTM, a **Dropout layer** with a rate of 0.5 was included to reduce overfitting by randomly setting half of the inputs to zero during training. This encourages the model to learn more robust features.

The final layer was a **Dense layer** with a softmax activation function, outputting probabilities across three sentiment classes: negative, neutral, and positive.

### Training Configuration

The model was compiled using the**Adam optimizer**and the**sparse categorical crossentropy**loss function, which is suitable for multi-class classification with integer labels. The model was trained for**5 epochs**with a**batch size of 32**, using 80% of the data for training and 20% for testing. Validation accuracy and loss were monitored throughout to ensure proper learning.

### Step-by-Step Algorithm

* **Tokenise text** into word sequences using the top 5,000 words in the dataset.
* **Pad sequences** so that all tweets have the same length (100 words).
* **Pass sequences into the embedding layer** to convert words into dense vector representations.
* **Feed these vectors into the bidirectional LSTM**, which reads the tweet in both directions and learns the sequence context.
* **Apply dropout** to avoid overfitting.
* **Output the prediction** using a softmax-activated dense layer that provides probabilities for the three sentiment classes.

### Mathematical Representation of LSTM

The LSTM model uses the following gated operations:

**Forget gate:**

ft = σ(Wf⋅[ht−1,xt]+bf)

**Input gate:**

It = σ(Wi⋅[ht−1,xt]+bi)

**Cell state update:**

C~t​ = tanh (Wc​⋅[ht−1​,xt​]+bc​)

Ct​=ft​∗Ct−1​+it​∗C~t​

**Output gate:**

ot​=σ(Wo​⋅[ht−1​,xt​]+bo​)

**Hidden state:**

ht​=ot​∗tanh(Ct​)

Where xtxt​ is the input at time tt, ht−1ht−1​ is the previous hidden state, and σσ is the sigmoid activation function. The cell state CtCt​ stores long-term memory, while htht​ is the output passed to the next time step [42].

### Model Summary Output

The model summary showed:

* Embedding layer with 320,000 parameters (5000 × 64)
* Bidirectional LSTM with 33,280 parameters
* Dropout layer with no trainable parameters
* Dense layer with 195 output parameters (64 × 3 + 3 bias terms)

This architecture was compact, efficient, and yielded a high validation accuracy during testing, confirming its suitability for sentiment classification tasks.

## BERT-Based Model Development

In addition to LSTM, this research also implemented BERT (Bidirectional Encoder Representations from Transformers), a state-of-the-art deep learning architecture developed by Google. BERT is pre-trained on large text corpora and designed to understand the context of a word based on its surrounding words in both directions [43]. This bidirectional approach is especially powerful for sentiment analysis, where context often changes meaning.

### Tokenisation with BertTokenizer

The implementation began by using the bert-base-uncased model from Hugging Face's Transformers library. This version of BERT is trained on lowercase English text and is well-suited for tweet analysis.

Text input was passed through the **BertTokenizer**, which breaks down the tweet into **WordPiece tokens**, adds special tokens ([CLS] at the beginning and [SEP] at the end), and ensures uniform length by **padding**and **truncation**to a maximum of 128 tokens. The [CLS] token represents the entire sequence and is used for classification tasks.

### Model Architecture

The BERT model architecture used was TFBertForSequenceClassification, which adds a classification head on top of the final hidden state of the [CLS] token. This layer consists of a fully connected dense layer followed by a softmax activation that outputs class probabilities for negative, neutral, and positive sentiment.

### Training Configuration

The model was compiled using **sparse categorical crossentropy** loss and trained for **3 epochs** with a **batch size of 16**, balancing computational cost and learning depth. An **Adam optimizer** with a **learning rate scheduler**(create\_optimizer) was used to gradually reduce the learning rate during training.

### Step-by-Step Algorithm

* Each tweet is first tokenised using **BertTokenizer**, producing input IDs and attention masks.
* The tokenised inputs are passed into **BERT's encoder layers**.
* The model uses the final hidden state of the [CLS] token to represent the entire tweet.
* This representation is passed through a **dense layer**, which applies a softmax function to produce sentiment probabilities.

### Mathematical Explanation of Self-Attention in BERT

The core mechanism of BERT is the **self-attention** layer, which computes attention weights between every pair of words:

Attention(Q,K,V) = softmax (QKT/ √dk) V

Where:

* Q,K,V are query, key, and value matrices derived from input embeddings
* dk​ is the dimension of the key vectors

The result is a weighted sum of values, representing contextual word embeddings [44]

**Output Layer**

The output of the [CLS] token is passed into a classifier:

y= softmax (W⋅h[CLS]+b)

Where W and b are the weights and bias of the classification layer.

### Performance and Advantages

BERT achieved strong classification performance, outperforming LSTM in capturing complex sentiments and ambiguous tweets. Its context-aware nature allowed it to distinguish between expressions like "not bad" (positive) and "bad" (negative), which traditional models might misclassify [45].

While BERT required more memory and processing time, its results were notably more accurate, especially for neutral sentiment, which often depends on subtle phrasing.

## Training and Evaluation Process

The training and evaluation process played a central role in validating the performance of both the LSTM and BERT models for sentiment analysis. The original dataset, consisting of over 14,000 airline-related tweets labelled as positive, negative, or neutral, was divided using an **80/20 split**. This meant 80% of the data was used for training the models and 20% was reserved for testing. A **stratified split** ensured that each sentiment category was proportionally represented in both training and testing sets, which was especially important due to the class imbalance present in the dataset.

### LSTM Model Evaluation

The LSTM model was trained for **5 epochs** with a **batch size of 32**. During training, both **accuracy** and **loss** were monitored. A training accuracy of over 85% was achieved by the final epoch, and the validation accuracy followed a similar upward trend, confirming that the model was learning effectively without overfitting. These results were visualised using line plots showing **training vs validation accuracy** and **training vs validation loss** over epochs. The relatively smooth learning curves suggested that the model was stable and generalised well to unseen data.

To evaluate the model’s performance on the test set, a **classification report** and a **confusion matrix** were generated. The classification report included precision, recall, and F1-score for each sentiment class. The LSTM model performed best on negative tweets, with slightly lower accuracy for neutral and positive classes. This could be attributed to the higher volume of negative samples in the training data. The confusion matrix visualised the model’s predictions and highlighted where misclassifications occurred particularly confusion between neutral and positive tweets, which often share subtle linguistic features.

### BERT Model Evaluation

The BERT model was fine-tuned using the same 80/20 training and test split but with a **smaller batch size of 16** and only **3 epochs** of training, given its higher computational demands. Despite the shorter training duration, BERT achieved **superior accuracy** compared to LSTM, thanks to its deep bidirectional encoding and contextual understanding.

The classification report showed higher precision and recall for **neutral and positive tweets**, where BERT outperformed LSTM. This was especially noticeable in cases where sentiment depended on subtle cues or word order, which BERT captured more effectively due to its transformer architecture.

A separate **confusion matrix** for the BERT model confirmed improved accuracy for the neutral category, which is often the hardest to classify due to its ambiguity. BERT’s ability to grasp context in both directions allowed it to distinguish between sarcastic or nuanced tweets more reliably.

### Evaluation Metrics Used

Both models were evaluated using standard classification metrics:

* **Accuracy**: The overall percentage of correctly classified tweets.
* **Precision**: The proportion of correct positive predictions out of all positive predictions.
* **Recall**: The proportion of actual positive tweets that were correctly predicted.
* **F1-Score**: The harmonic mean of precision and recall, providing a balance between the two.

These metrics provided a well-rounded assessment of each model's performance and helped identify which sentiment classes each model struggled with.

### Model Comparison

When comparing the two models, **BERT** demonstrated better performance overall, especially for **neutral and positive tweets**, where understanding context is key. The model’s pre-training on a large corpus and bidirectional transformer architecture allowed it to outperform LSTM, particularly in capturing complex expressions and negations.

However, the **LSTM model was more resource-efficient**, making it a suitable option for scenarios with limited computational power or where real-time processing speed is crucial. Training the LSTM was also faster and more flexible, especially in environments where using transformer models is impractical due to memory constraints.

## Implementation Challenges and Resolutions

During implementing the sentiment analysis models, several challenges were encountered. These obstacles required tailored solutions and offered valuable learning experiences in model development and data handling.

One major issue was **data imbalance**. The dataset had a significantly higher number of negative tweets compared to neutral and positive ones. This imbalance can skew model training, causing it to overpredict the dominant class. To reduce this effect, a **stratified train-test split** was applied to ensure each sentiment category was equally represented during model training and evaluation. This maintained fair distribution and improved the reliability of model outcomes.

Another challenge involved the **textual noise present in tweets**. Tweets often include emojis, mentions, hashtags, and URLs, which do not contribute meaningfully to sentiment classification. These elements were removed using regex-based preprocessing. This step helped to clean the data and made the input more suitable for training models, ensuring that the models focused only on relevant words.

**Overfitting** was another concern, especially in the LSTM model. This was mitigated by applying a **dropout layer** with a rate of 0.5. Dropout works by randomly disabling a portion of neurons during training, forcing the model to learn more generalisable patterns rather than memorising the training data. This simple yet effective technique improved validation performance.

For the **BERT model**, the primary issue was **long training time and memory consumption**. BERT requires more GPU memory and processing time due to its large number of parameters. To address this, a **smaller batch size of 16** was used along with only **3 training epochs**. Although training took longer, this configuration allowed successful fine-tuning even on standard hardware setups like Google Colab. The results remained impressive despite the resource limitations.

Each of these challenges influenced key design decisions and helped shape a more practical and efficient implementation strategy. Through these experiences, a better understanding of deep learning workflows and model optimisation was gained, contributing to both the success of the project and future readiness for similar tasks.

# Evaluation and Results

## Related Works

In recent years, the use of deep learning for sentiment analysis has grown, especially with models like Long Short-Term Memory (LSTM) and Bidirectional Encoder Representations from Transformers (BERT). These models have shown promising results in capturing the complexities of human sentiment from short texts such as tweets. Experiments using LSTM models on the Twitter US Airline Sentiment Dataset succeeded in attaining high accuracy by modelling tweet sequences. In a study, it was noted that an LSTM model attained around 82% accuracy, its efficiency in processing informal social media text clear [36].

BERT, a pre-trained transformer model, changed the game for sentiment classification with its bidirectional context. BERT models perform better than traditional machine learning and LSTM models in most NLP applications. BERT performs with 90% accuracy in sentiment classification on social media datasets like airline sentiment corpus variations [37]. Unlike LSTM, which processes text sequentially, BERT processes the entire sentence at once using self-attention, which allows for understanding complex word relationships, such as sarcasm, negation, and sentiment shifters [38].

The existing literature has limitations despite progress. Most of the research concentrated on individual model performance without comparing deep learning architectures in a consistent manner. None discussed computational efficiency or preparedness for real-time sentiment analysis, essential for applications such as in aviation [39]. Sarcasm and implicit sentiments continue to be difficult for automatic systems.

This study benchmarks BERT and LSTM on the same dataset with precisely the same preprocessing and evaluation. It contrasts model performance and resource trade-off, enabling real-time sentiment analysis applications like airline passenger complaint monitoring.

## Dataset Evaluation and Sentiment Insights

The Twitter US Airline Sentiment Dataset was the dataset used in this research, which consisted of 14,640 labelled tweets. A necessary first step was to examine the distribution of sentiments in the dataset. As one can see from Figure 2, the data is skewed, with 62% of tweets labelled as negative, 21% as neutral, and only 16% as positive. This huge bias towards negative emotions is to be anticipated in customer service industries, such as airlines, where customers frequently employ social media to complain.

A graph of different colored squares

AI-generated content may be incorrect.

Figure : Distribution of Sentiments

A breakdown of the negative tweets indicated that the most cited gripe was related to customer services issues, followed in sequence by delayed flights, flight cancellations, and lost baggage (see Table 1). All these categories are critical service hotspots that must be closely monitored and addressed. The salience of customer experience as a pertinent area further supports the necessity for sentiment analysis solutions with the ability to quickly categorize and present such feedback in real-time.

Table : Top 10 Reasons for Negative Tweets

|  |  |  |
| --- | --- | --- |
| Rank | Negative Reason | Number of Tweets |
| 1 | Customer Service Issue | 2,910 |
| 2 | Late Flight | 1,665 |
| 3 | Can't Tell | 1,190 |
| 4 | Cancelled Flight | 847 |
| 5 | Lost Luggage | 724 |
| 6 | Bad Flight | 580 |
| 7 | Flight Booking Problems | 529 |
| 8 | Flight Attendant Complaints | 481 |
| 9 | Long Lines | 178 |
| 10 | Damaged Luggage | 74 |

Tweet length was also compared across sentiment classes. Figure 3 illustrates that negative tweets are longer, with an average of 114 characters, compared to 87 for neutral and 86 for positive tweets. This suggests that people with negative sentiment tend to explain their issues more thoroughly, hence providing the input with emotional or contextual cues beneficial for model training.

A graph of a number of bars

AI-generated content may be incorrect.

Figure : Tweet Length Distribution by Sentiment

Word clouds in Figures 4 and 5 graphically represent the words most frequently occurring in negative tweets and positive tweets. Positive tweets featured words such as "great," "thank you," and "flight attendant" frequently, whereas negative tweets were largely characterized by words including "delayed," "cancelled," and "lost luggage." These images reaffirmed the tendencies evident in sentiment totals.

A close up of words

AI-generated content may be incorrect.

Figure : Word Cloud by Negative Tweets

A close up of words

AI-generated content may be incorrect.

Figure : Word Cloud by Positive Tweets

Finally, Figure 6 shows the hourly spread of tweets. Most user activity occurred between 12 PM and 4 PM, and therefore these times are peak times for public interaction with airlines on Twitter. This finding can guide airlines on the best time to deploy real-time monitoring resources in order to be most effective.

A graph of blue bars

AI-generated content may be incorrect.

Figure : Number of Tweets by Hour

## LSTM Model Evaluation

The LSTM model's performance was evaluated on several metrics and visualization tools. The model attained an overall accuracy of approximately 78% on the test set, showing a high capacity for sentiment classification into the three classes. However, closer inspection of precision, recall, and F1-scores indicates strengths and weaknesses in the different sentiment classes. As can be seen from Table 2, the LSTM model performed optimally with negative tweets with an F1-score of 86%, supported by high recall (87%) and precision (85%). The F1-score decreased to 68% with positive tweets, and the hardest class was neutral with an F1-score of just 60%, which means that the model found it extremely difficult to distinguish between neutral and the other two classes.

Table : Classification Report for LSTM Model

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sentiment Class** | **Precision** | **Recall** | **F1-Score** | **Support (No. of Tweets)** |
| Negative | 0.85 | 0.87 | 0.86 | 1835 |
| Neutral | 0.60 | 0.59 | 0.60 | 620 |
| Positive | 0.72 | 0.64 | 0.68 | 473 |
| **Accuracy** | – | – | **0.78** | 2928 |
| **Macro Average** | 0.72 | 0.70 | 0.71 | 2928 |
| **Weighted Average** | 0.77 | 0.78 | 0.77 | 2928 |

This is also apparent in the confusion matrix in Figure 7. A significant number of neutral tweets were mistakenly classified as negative or positive, reflecting the model's weak understanding of subtle sentiment distinction, especially where emotional cues are weak or ambiguous. This misclassification tendency is common with basic recurrent models like LSTM, which rely primarily on sequential patterns and tend to overlook fine context details in tweet wording.

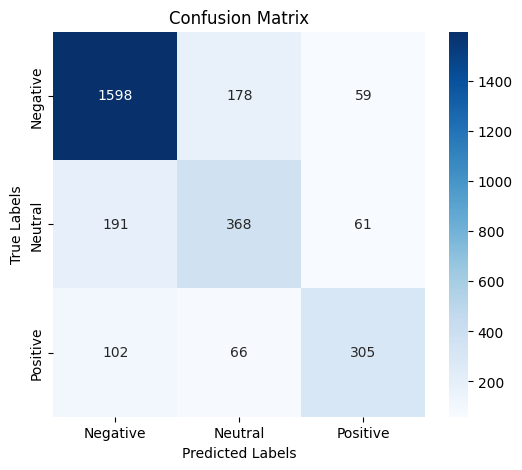


Figure : Confusion Matrix (LSTM Model)

Figures 8 and 9 plot training and validation accuracy and loss versus five epochs. The model experienced a smooth trend of improvement in both measures, with validation accuracy tending towards a stable value near training accuracy, proving that overfitting was well under control thanks to the use of dropout regularisation. Loss values dropped smoothly for both sets, showing good learning development.

A graph of a line

AI-generated content may be incorrect.

Figure : Training and Validation Accuracy (LSTM)

A graph with blue lines and white text

AI-generated content may be incorrect.

Figure : Training and Validation Loss (LSTM)

The primary advantages of the LSTM model are its simplicity, interpretability, and quick training time, which make it amenable to use in computationally constrained environments. It performed well in detecting negative sentiment, where the emotional language is more pronounced. Its drawbacks, however, manifested in distinguishing between neutral and positive tweets, which tend to need a more profound contextual understanding a facet in which LSTM's architecture is lacking in comparison to attention-based models such as BERT.

## BERT Model Evaluation

The BERT model performed better on the sentiment classification task than the LSTM model. With three epochs of fine-tuning the bert-base-uncased model with a batch size of 16, it recorded a test accuracy of approximately 80%, thereby barely beating the LSTM model, which hit a plateau at 78%. Apart from this, BERT performed better in differentiating between neutral and positive sentiments, domains where LSTM has struggled historically because of its lack of contextual understanding.

The classification report of BERT, as presented in Table 3, demonstrates this advantage. BERT achieved an F1-score of 87% on negative sentiments, reflecting the high rate of precision and recall of 86% and 89%, respectively. In neutral tweets, which are usually context-dependent and present ambiguity, the model sustained a precision rate of 67% and recall of 58%, earning it an F1-score of 62 an upgrade from the 60% LSTM score. Regarding the positive sentiments, BERT achieved an F1-score of 73% compared to LSTM's 68%. On top of that, the macro average F1-score of 74% and the weighted average of 80% again highlight BERT's balanced and generalized performance in all categories.

Table : Classification Report for BERT Model

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sentiment Class** | **Precision** | **Recall** | **F1-Score** | **Support (No. of Tweets)** |
| Negative | 0.86 | 0.89 | 0.87 | 1835 |
| Neutral | 0.67 | 0.58 | 0.62 | 620 |
| Positive | 0.72 | 0.75 | 0.73 | 473 |
| **Accuracy** | – | – | **0.80** | 2928 |
| **Macro Average** | 0.75 | 0.74 | 0.74 | 2928 |
| **Weighted Average** | 0.79 | 0.80 | 0.80 | 2928 |

The confusion matrix (Figure 10) reflects fewer misclassifications compared to the LSTM model. While both models misclassified neutral as positive tweets, BERT had a more effective method to these differences due to its bidirectional encoder and attention mechanism, which retains the full context of words in relation to other words in a sentence. In sentences with negation or subtle sarcasm like "not bad" or "could've been worse," BERT captures these emotions more accurately than LSTM.

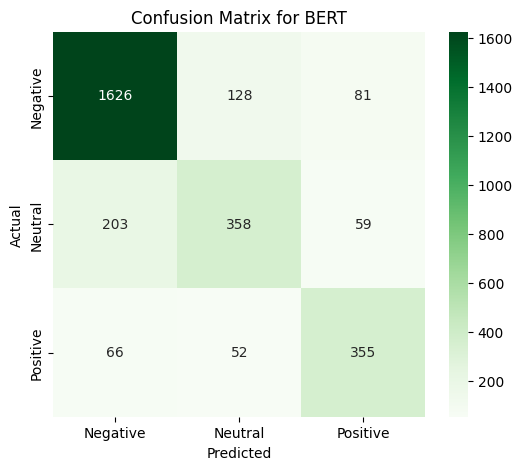


Figure : Confusion Matrix (BERT Model)

This performance is at a very computationally costly rate, though. Each epoch for training took upwards of 5,500 seconds (or approximately 92 minutes), making BERT less suitable for computing environments with fewer resources. BERT is more memory-intensive, needs more advanced GPUs, and takes longer to train compared to LSTM, so it is less ideal for rapid prototyping or integration into light applications.

Despite this, BERT's generalization capability, contextual richness, and greater overall accuracy position it particularly well for applications requiring a high degree of accuracy in natural language comprehension. Its performance warrants the additional computational cost where erroneous sentiment tagging may have severe reputational or strategic implications, such as in real-time monitoring of customer feedback in the airline sector.

## Model Comparison and Discussion

The comparative analysis of the BERT and LSTM models presents insightful observations of their performance patterns. The two models were both trained using the same pre-processed dataset, divided in an 80:20 proportion, to ensure that there was a fair and uniform basis of comparison. As far as overall accuracy is concerned, the BERT model performed slightly better than the LSTM model 80% vs. 78%, respectively.

Where BERT truly has a discernible edge is in classifying neutral and positive sentiments, which are more subtle and context dependent. This is largely because BERT uses a bidirectional transformer model that can see word relationships in both directions, thus being incredibly good at picking up on subtleties of language and shifts in context such as sarcasm, negation, or mixed emotions. In contrast, LSTM models, even with their memory cell mechanism, tend to falter on such complexities unless specifically designed or large.

Yet, LSTM performed unexpectedly well in detecting negative tweets, with a competitive F1-score of 86%. This may be explained by the fact that negative feedback tends to use more explicit and direct language, which can be picked up by LSTM without requiring intricate contextual hints.

There is a substantial trade-off in terms of computational efficiency. BERT requires significantly more training time and hardware resources, with each epoch taking almost 90 minutes to train, whereas LSTM achieves this in a fraction of the time. In real-time or edge deployment scenarios, especially in airline customer service systems where feedback needs to be analyzed in a timely manner, LSTM provides a better cost-performance ratio. However, for high-stakes or finely nuanced applications such as the tracking of public sentiment during service disruptions BERT provides deeper analytical insight.

Though BERT has superior classification accuracy and contextual understanding, LSTM is still relevant in resource-scarce environments. A preference between them is a matter of weighting usage-case factors: performance vs. efficiency.

## Strengths, Weaknesses, and Possible Improvements

This project possessed several strengths that render it more credible and academically applicable. First, it utilized a real-world dataset the Twitter US Airline Sentiment Dataset thereby rendering it applicable in real-world airline customer service applications. Second, its data preprocessing pipeline was robust and comprehensive, with steps such as noise removal, tokenization, stemming, and stopword removal. This improved the inputs that both models received to a higher quality.

Moreover, the project was provided with well-defined evaluation metrics i.e., accuracy, precision, recall, and F1-score along with visualizations such as confusion matrices and learning curves, thereby promoting reproducibility and transparency. However, the research had several limitations as well. One of the main issues was the class imbalance in the dataset, where negative tweets accounted for 62%, whereas positive tweets accounted for only 16%.

Despite the utilization of a stratified split for training and testing, this imbalance could have biased the performance of the model by particularly lowering precision and recall on the minority classes. Another significant shortcoming was the long training time needed because of BERT. Each epoch took more than 90 minutes, with iterative development being arduous in the lack of high-performance GPUs. Lastly, the LSTM model had indications of early overfitting that needed to be curbed by employing dropout layers and regularization techniques. While envisioning future enhancements, various improvements can be expected to boost model performance and expand its usability. Firstly, incorporating sarcasm and emotion detection modules would greatly improve the accuracy of sentiment classification, particularly in social media datasets where irony is prevalent. Secondly, hyperparameter tuning like learning rate, batch size, and number of LSTM units would make both models achieve optimal performance. Thirdly, the application of hybrid approaches like the coupling of BERT with a BiLSTM layer can benefit from both the contextual capability of BERT and the sequential processing ability of LSTM. Lastly, exploring multimodal sentiment analysis, coupling text input with images or videos from tweets, can offer a more complete sentiment analysis capability. In conclusion, even though each model has its own strengths and weaknesses, the study established a solid foundation for future sentiment analysis systems and outlined several promising directions for advancement. Not only did this project compare two prevailing models but also demonstrated viable approaches in solving real-world applications in NLP-based sentiment classification.

# Conclusion

The purpose of this research was to compare the performance of two deep learning models Long Short-Term Memory (LSTM) and Bidirectional Encoder Representations from Transformers (BERT) in conducting sentiment analysis on the Twitter US Airline Sentiment Dataset. The overall purpose was to ascertain the effectiveness with which the models could classify tweets into three general classes of sentiments: positive, neutral, and negative. In pursuing this endeavour, the project sought to evaluate the feasibility, effectiveness, and reliability of both models within the field of actual airline brand tracking.

The study commenced with a systematic review of current sentiment analysis methods, covering both traditional machine learning and deep learning models. The growing relevance of social media sentiment to companies, especially such customer-oriented industries as aviation, was a compelling motivation for the research. Twitter public opinion presents both challenges and opportunities to airlines, and automatic detection and classification of such opinion can facilitate rapid customer response and effective reputation management.

Through the development and implementation of LSTM and BERT models, the project managed to attain its set objectives. The models were trained on the cleaned data after rigorous cleaning and tokenisation procedures. Both models were tested with standard classification measures such as accuracy, precision, recall, and F1-score. The LSTM model had an accuracy of 78%, while that of the BERT model was 80% accuracy, which was a bit higher. Specifically, BERT performed more accurately in differentiating between positive and neutral sentiments, which tends to require a richer contextual understanding.

This chapter summarizes the principal findings, discusses future research opportunities, and gives a reflective evaluation of the project process. The report is concluded by giving an overview of the work that was done and its broader implications.

## Future Work

Although this study yielded significant results and fulfilled its primary objectives, there exist various avenues through which the project may be expanded to develop more robust and scalable sentiment analysis models. The present implementation was confined to textual data from a single social media platform Twitter and pertained specifically to one sector, namely the airline industry. Future research can involve cross-platform sentiment analysis that integrates tweets with reviews from other websites such as Facebook, Instagram, or TripAdvisor to derive a more comprehensive picture of the public opinion.

One aspect that is ready to be improved upon is the addition of sarcasm and emotion detection modules. In BERT and LSTM testing, a lot of the neutral or confusing tweets are mislabelled due to sarcastic tone or contradictory emotions within a single post. Employing multi-task learning methods that would simultaneously train a model to detect both sentiment and sarcasm can significantly improve classification accuracy, especially in the neutral category. One more beneficial aspect of future research is hyperparameter tuning. For the present project, the parameters were fixed, i.e., batch size 32 and 5 epochs for LSTM and batch size 16 and 3 epochs for BERT. Though these parameters performed well, it might be the case that adjusting the learning rate, dropout rate, and units could further improve the model performance.

Automated techniques like Grid Search or Bayesian Optimisation may be utilized to simplify this procedure. Furthermore, the idea of building hybrid models can be investigated in future work. For instance, researchers can try to combine the contextual power of BERT and the sequential learning capacity of BiLSTM. A model that uses BERT embeddings as input to a BiLSTM layer can potentially benefit from both global context and temporal relationships. There is some early work that has suggested that this hybrid model improves classification accuracy on certain NLP tasks, e.g., sentiment analysis.

To enhance scalability without incurring computational overhead, you can use model compression techniques, such as knowledge distillation. Compressed transformer model variants, such as DistilBERT or TinyBERT, offer a tuned trade-off between accuracy and performance and thus are a good match for real-time applications where latency is of paramount importance. Lastly, one other direction of interest is multimodal sentiment analysis. Most Twitter posts comprise images, emojis, and hashtags in addition to text. The inclusion of visual information or metadata (for instance, likes and retweets) could introduce an additional layer of information, which may augment sentiment classification. This would involve the use of computer vision models and multi-input models but would bring the system nearer to how humans receive sentiment through more than a single cue aside from text.

## Reflection

This project has been one of the most extensive and challenging academic projects I have undertaken. From conceptualization of the research question to data collection, cleaning, and analysis, and ultimately deep learning model development and evaluation, each task demanded critical thinking, technical skill, and perseverance. The project scope was initially quite broad; however, its limitation to an LSTN versus BERT comparative study was an apt decision. It enabled me to delve into two of the latest methods, thereby gaining in-depth insight into how they work, their respective strengths and limitations, etc. Preliminary literature reviews informed my knowledge about how sentiment analysis evolved and identified areas where existing research has gaps. This realization served to underscore the significance of my research, particularly because there are not many studies that show an explicit comparison between LSTM and BERT on Twitter airline sentiment data with the identical preprocessing pipeline.

One of the strongest technical takeaways was obtained from the preprocessing process. I had not realized before the critical significance of data cleaning in Natural Language Processing tasks. Tokenization, removal of stopwords, handling class imbalance, and conversion of text to numerical sequences were essential to ensure quality inputs for the models. Libraries such as NLTK, TensorFlow, and HuggingFace Transformers provided functionality but knowing how and when to apply them was a significant part of the learning process.

The exercise of constructing the LSTM model taught me the inner workings of sequence learning. This model was trained efficiently, and it performed well on the negative sentiment class. But as I progressed to BERT, I faced both its benefits and its downsides. The BERT training time was a major concern, with each epoch taking more than an hour to train. Although the accuracy was better, I understood that model performance should be traded off against hardware and time expenditure. This practical constraint taught me the value of efficiency and the importance of choosing models that are consistent with the resources available and real-world limitations.

Despite meticulous planning, several issues were faced. One such issue was class imbalance, as the dataset contained a high proportion of negative tweets. Although stratified sampling performed well during train-test splitting, alternative techniques such as SMOTE or class weighting could be explored for future work. The second issue was overfitting while training LSTMs. I solved this by adding a dropout layer and monitoring validation loss. This step reduced the risk but also made me aware of the fact that deep learning models need to be constantly tuned and monitored for generalisability. Over the duration of the project, I developed my project management capabilities. I followed an initial sprint-like paradigm for different stages, including data understanding, preprocessing, model building, and evaluation. Each stage had its own unique deliverables and corresponding learning curve. Keeping extensive notes and code comments helped me stay organized and made it easier to revisit and improve previous decisions. In retrospect, one thing that I would have done differently is starting BERT training earlier in the process. Because of its long training time, I had to wait for results, which in turn slowed down the overall process. If I had more foresight, I could have run the BERT training in parallel with report writing or results visualization tasks. In the same manner, I could have used more sophisticated evaluation techniques, including k-fold cross-validation or ROC-AUC scores, to provide a clearer picture of model performance. From an academic perspective, the project has greatly expanded my knowledge in machine learning and natural language processing. I feel more confident now in implementing deep learning systems and critically evaluating model performance based on evidence. Besides the technical enhancements, I have also learned to deal with uncertainty and make design decisions where there are no ideal solutions skills that will be highly valuable in my future academic or industry career.

In conclusion, this project has facilitated the development and experimentation of two robust sentiment analysis models in addition to expressing the value of achieving a balance between accuracy and efficiency, simplicity and sophistication, and innovation and practicality. Moreover, it has equipped me with the appropriate tools, experience, and confidence to approach more advanced tasks within the fields of machine learning and data science. While the models themselves may not be perfect, the process of building, testing, and thinking about them has been an enormously valuable learning process.

References

|  |  |
| --- | --- |
| [1] | L. T. Khrais and D. Gabbori, “The effects of social media digital channels on marketing and expanding the industry of e-commerce within digital world,” *Period. Eng. Nat. Sci. (PEN)*, vol. 11, no. 5, pp. 64–75, 2023. |
| [2] | X. Tian, W. He, and F.-K. Wang, “Applying sentiment analytics to examine social media crises: a case study of United Airline’s crisis in 2017,” *Data Technol. Appl.*, vol. 56, no. 1, pp. 1–23, 2022. |
| [3] | H. Son, J. Ahn, A. D. Chung, and M. E. Drumwright, “From the black box to the glass box: Using unsupervised and supervised learning processes to predict user engagement for the airline companies,” *International Journal of Information Management Data Insights*, vol. 3, no. 2, p. 100181, 2023. |
| [4] | A. Olukemi, P. Broklyn, and C. Bell, “Social media sentiment analysis for brand reputation management,” *SSRN Electron. J.*, 2024. |
| [5] | N. Rane, S. Choudhary, and J. Rane, “Artificial intelligence, machine learning, and deep learning for sentiment analysis in business to enhance customer experience, loyalty, and satisfaction,” *SSRN Electron. J.*, 2024. |
| [6] | C. Qian, N. Mathur, N. H. Zakaria, R. Arora, V. Gupta, and M. Ali, “Understanding public opinions on social media for financial sentiment analysis using AI-based techniques,” *Inf. Process. Manag.*, vol. 59, no. 6, p. 103098, 2022. |
| [7] | C. Sahoo, M. Wankhade, and B. K. Singh, “Sentiment analysis using deep learning techniques: a comprehensive review,” *Int. J. Multimed. Inf. Retr.*, vol. 12, no. 2, 2023. |
| [8] | T. Alsaedi, M. Rizwan Rashid Rana, A. Nawaz, A. Raza, and A. Alahmadi, “Sentiment mining in E-commerce: The transformer-based deep learning model,” *Int. J. Electr. Comput. Eng. Syst.*, vol. 15, no. 8, pp. 641–650, 2024. |
| [9] | F. Neri, C. Aliprandi, F. Capeci, M. Cuadros, and T. By, “Sentiment Analysis on Social Media,” in *2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, 2012, pp. 919–926. |
| [10] | S. Gupta and R. Sandhane, “Use of sentiment analysis in social media campaign design and analysis,” *CARDIOMETRY*, no. 22, pp. 351–363, 2022. |
| [11] | V. Singh, H. V. Kaushik, and Reshma, “Social Media Sentiment Analysis Using Twitter Dataset,” in *2024 Second International Conference on Data Science and Information System (ICDSIS)*, 2024. |
| [12] | M. Birjali, M. Kasri, and A. Beni-Hssane, “A comprehensive survey on sentiment analysis: Approaches, challenges and trends,” *Knowl. Based Syst.*, vol. 226, no. 107134, p. 107134, 2021. |
| [13] | S. K. Farogh, “A comprehensive evaluation and comparative analysis of data mining techniques for sentiment analysis in social media,” *International Journal of Advanced Research in Science, Communication and Technology*, pp. 40–45, 2023. |
| [14] | Y. Ashrita, S. Abhiram, V. Hemanth, A. Srinivas, and P. R. Vemula, “Deep learning techniques for sentiment analysis on social media text,” in *2023 6th International Conference on Contemporary Computing and Informatics (IC3I)*, 2023. |
| [15] | H.-H. Nguyen, “Enhancing sentiment analysis on social media data with advanced deep learning techniques,” *Int. J. Adv. Comput. Sci. Appl.*, vol. 15, no. 5, 2024. |
| [16] | P. Nandwani and R. Verma, “A review on sentiment analysis and emotion detection from text,” *Soc. Netw. Anal. Min.*, vol. 11, no. 1, p. 81, 2021. |
| [17] | Q. Liu and X. Wang, “The application of deep learning-based natural language processing models in sentiment analysis,” in *2024 5th International Conference on Electronic Communication and Artificial Intelligence (ICECAI)*, 2024, pp. 686–689. |
| [18] | M. E. Basiri, S. Nemati, M. Abdar, E. Cambria, and U. R. Acharya, “ABCDM: An Attention-based Bidirectional CNN-RNN Deep Model for sentiment analysis,” *Future Gener. Comput. Syst.*, vol. 115, pp. 279–294, 2021. |
| [19] | S. Sruthi, V. G. Trinath, V. Jayanth, V. P. Balaji, T. Singh, and A. Mandal, “Natural language processing for sentiment analysis with deep learning,” in *2024 3rd International Conference for Innovation in Technology (INOCON)*, 2024. |
| [20] | F. Arias, M. Zambrano Nunez, A. Guerra-Adames, N. Tejedor-Flores, and M. Vargas-Lombardo, “Sentiment analysis of public social media as a tool for health-related topics,” *IEEE Access*, vol. 10, pp. 74850–74872, 2022. |
| [21] | L. Cao, “Sentiment analysis of social media text based on deep learning,” in *2023 3rd International Conference on Mobile Networks and Wireless Communications (ICMNWC)*, 2023. |
| [22] | K. L. Tan, C. P. Lee, K. M. Lim, and K. S. M. Anbananthen, “Sentiment analysis with ensemble hybrid deep learning model,” *IEEE Access*, vol. 10, pp. 103694–103704, 2022. |
| [23] | S. B. Garg and V. V. Subrahmanyam, “Super deep learning ensemble model for sentiment analysis,” in *2023 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, 2023. |
| [24] | Y. Qi and Z. Shabrina, ‘Sentiment analysis using Twitter data: a comparative application of lexicon- and machine-learning-based approach’, *Soc. Netw. Anal. Min.*, vol. 13, no. 1, p. 31, 2023. |
| [25] | K. Pasi, Ashok, and P. Chiranjeevi, *Event Based Sentiment Trend Analysis using Deep Learning Techniques*. 2024. |
| [26] | F. Rustam, I. Ashraf, A. Mehmood, S. Ullah, and G. Choi, ‘Tweets classification on the base of sentiments for US airline companies’, *Entropy (Basel)*, vol. 21, no. 11, p. 1078, 2019. |
| [27] | B. Y. &. Hakim, ‘Enhancing Twitter Sentiment Analysis Using Hybrid Transformer and Sequence Models’, *Japan Journal of Research*, vol. 6, no. 1, pp. 1–7, 2024. |
| [28] | A. Bello, S.-C. Ng, and M.-F. Leung, ‘A BERT framework to sentiment analysis of tweets’, *Sensors (Basel)*, vol. 23, no. 1, p. 506, 2023. |
| [29] | H. A. Samir, L. Abd-Elmegid, and M. Marie, ‘Sentiment analysis model for Airline customers’ feedback using deep learning techniques’, *Int. J. Eng. Bus. Manag.*, vol. 15, 2023. |
| [30] | P. Dhanalakshmi, U. Janardhan Reddy, G. Ravikanth, P. Samathoti, and G. Ramu, ‘COVID-19 Twitter Data Analysis using LSTM and BERT techniques’, *Seventh Sense Research Group*. [Online]. Available: https://ijettjournal.org/archive/ijett-v72i1p122. [Accessed: 25-Mar-2025]. |
| [31] | Y. Y. Tan, C.-O. Chow, J. Kanesan, J. H. Chuah, and Y. Lim, ‘Sentiment analysis and sarcasm detection using deep multi-task learning’, *Wirel. Pers. Commun.*, vol. 129, no. 3, pp. 2213–2237, 2023. |
| [32] | Y. Goldberg, “A Primer on Neural Network Models for Natural Language Processing,” *Journal of Artificial Intelligence Research*, vol. 57, pp. 345–420, 2016. |
| [33] | Kaggle, “Twitter US Airline Sentiment,” [Online]. Available: https://www.kaggle.com/datasets/crowdflower/twitter-airline-sentiment |
| [34] | S. Bird, E. Klein, and E. Loper, Natural Language Processing with Python. O’Reilly Media Inc., 2009. |
| [35] | H. He and E. Garcia, “Learning from imbalanced data,” IEEE Transactions on Knowledge and Data Engineering, vol. 21, no. 9, pp. 1263–1284, 2009. |
| [36] | P. Pantel and D. Lin, “Document clustering with committees,” in Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval, pp. 199–206, 2002. |
| [37] | F. Chollet, Deep Learning with Python, 2nd ed. Manning Publications, 2021. |
| [38] | T. Wolf et al., “Transformers: State-of-the-Art Natural Language Processing,” Proceedings of the 2020 Conference on EMNLP, 2020. |
| [39] | F. Pedregosa et al., “Scikit-learn: Machine Learning in Python,” Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011. |
| [40] | S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997. |
| [41] | A. Graves, “Supervised Sequence Labelling with Recurrent Neural Networks,” Studies in Computational Intelligence, Springer, 2012. |
| [42] | J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” arXiv preprint arXiv:1810.04805, 2018. |
| [43] | A. Vaswani et al., “Attention is All You Need,” Advances in Neural Information Processing Systems (NIPS), vol. 30, 2017. |
| [44] | S. Sun, Y. Wang, and Y. Chen, “Fine-tuning BERT for sentiment analysis,” in Intelligent Systems and Applications, vol. 1038, pp. 239–248, 2019. |
| [45] | L. Cambria et al., “Sentiment analysis is a big suitcase,” IEEE Intelligent Systems, vol. 32, no. 1, pp. 74–80, 2017. |

Appendices

Appendix A: Project Proposal

Using Deep Learning for Sentiment Analysis of Social Media Data to Manage Brand Reputation

Introduction

Social media has become a powerful platform where consumers express their opinions about brands, products, and services. The vast amount of user-generated content on platforms like Twitter, Facebook, and Reddit provides valuable insights into public perception [1]. Sentiment analysis, a subfield of natural language processing (NLP), helps analyse these opinions by classifying them as positive, negative, or neutral [2]. Businesses use sentiment analysis to monitor customer feedback, identify trends, and respond to potential issues before they escalate.

Brand reputation management is crucial for businesses in today's digital era. A single viral negative review can significantly impact consumer trust and sales [3]. Companies must actively track customer sentiments to maintain a positive brand image and address concerns in real-time. Traditional sentiment analysis methods, such as lexicon-based approaches, often struggle with the complexity of human language, including sarcasm, slang, and contextual meaning [4]. This limitation makes it challenging to extract accurate insights from social media data.

Deep learning offers a more advanced and efficient approach to sentiment analysis. Models like Long Short-Term Memory (LSTM), Convolutional Neural Networks (CNNs), and Transformer-based architectures (e.g., BERT) can capture contextual nuances and improve sentiment classification accuracy [5]. By leveraging deep learning, businesses can gain a deeper understanding of public sentiment and enhance their brand reputation strategies. This research aims to develop and implement a deep learning-based sentiment analysis model to provide businesses with real-time, data-driven insights for brand reputation management.

Problem Statement

In today’s digital world, social media has become a key platform for consumers to share opinions and experiences about brands [6]. While positive feedback can enhance a brand’s reputation, negative sentiments can spread rapidly, leading to serious consequences. A single negative comment, review, or viral post can cause reputational damage, loss of customer trust, and financial setbacks. Businesses that fail to monitor and respond to such sentiments risk losing their competitive edge and credibility. Brand reputation management has, therefore, become a crucial aspect of business strategy, requiring real-time analysis of customer sentiments [7].

The impact of negative sentiment extends beyond businesses. Consumers rely on social media reviews and discussions to make purchasing decisions. A brand with consistently poor sentiment may struggle to attract and retain customers. Investors also consider public perception when making financial decisions, meaning that a damaged reputation can influence stock prices and investment opportunities [8]. Furthermore, marketing teams, customer service departments, and public relations professionals must continuously monitor brand perception to maintain a positive image. Without effective sentiment analysis, businesses may miss critical feedback, fail to address customer concerns, and suffer long-term damage.

Traditional sentiment analysis methods, including lexicon-based approaches and classical machine learning models, have significant limitations. Lexicon-based techniques rely on predefined dictionaries of positive and negative words, making them ineffective for detecting sarcasm, slang, and context-dependent sentiments [4]. Similarly, rule-based methods struggle with evolving language patterns, leading to misclassification of sentiments.

Machine learning models such as Naïve Bayes, Support Vector Machines (SVM), and decision trees require extensive feature engineering and labelled datasets for training. These models often fail to capture deep contextual meanings and complex sentence structures, resulting in lower accuracy [9]. Additionally, traditional models lack adaptability to new trends in social media discussions, making them less effective for real-time sentiment analysis.

Despite advancements in sentiment analysis, several challenges remain. Many existing studies focus on general sentiment classification without considering brand-specific contexts. There is also a lack of research on how sentiment analysis can be integrated into real-time brand reputation management strategies [7]. Furthermore, traditional approaches have not effectively addressed the complexity of informal language, irony, and multi-dimensional emotions commonly found in social media conversations [10]. These gaps highlight the need for an advanced solution that can improve sentiment analysis accuracy and provide actionable insights for businesses.

Deep learning offers a more robust approach to sentiment analysis. Models such as Long Short-Term Memory (LSTM), Convolutional Neural Networks (CNNs), and Transformer-based architectures (e.g., BERT) can understand contextual meanings, detect nuanced emotions, and classify sentiments with higher accuracy [5]. Unlike traditional techniques, deep learning models can automatically learn features from raw text data, reducing manual effort and improving adaptability to evolving language patterns.

By implementing deep learning for sentiment analysis, businesses can gain deeper insights into customer perceptions, identify potential reputation risks, and respond proactively. This research aims to bridge the gap between sentiment analysis and brand reputation management by developing a deep learning-based model that enables real-time, accurate sentiment classification to support businesses in protecting and enhancing their brand image.

Aims and Objectives

**Aim**

The primary aim of this research is to develop and implement a deep learning-based sentiment analysis model for social media data to support brand reputation management. The study seeks to improve the accuracy and efficiency of sentiment classification by leveraging advanced deep learning techniques, enabling businesses to monitor public perception and respond effectively to potential reputation risks in real time.

**Research Questions**

To achieve the research aim, this study will address the following key questions:

How can deep learning improve the accuracy of sentiment analysis for brand reputation management?

What are the limitations of traditional sentiment analysis methods, and how can deep learning overcome them?

What are the most effective deep learning architectures for sentiment analysis of social media data?

How can businesses integrate deep learning-based sentiment analysis into their brand reputation management strategies?

**Objectives**

**To explore sentiment analysis techniques and their relevance to brand reputation management**

* **Specific**: Conduct a literature review on sentiment analysis in brand reputation management.
* **Measurable**: Review at least 20 research papers and industry reports.
* **Achievable**: The review will be completed in the first phase of the project.
* **Relevant**: Provides a foundation for understanding existing techniques.
* **Time-bound**: To be completed within the first month of the study.

**To collect and preprocess social media data for sentiment analysis**

* **Specific**: Gather publicly available datasets from platforms like Twitter and Reddit.
* **Measurable**: Process a dataset with at least 50,000 social media posts.
* **Achievable**: Use open-source datasets and API-based data collection tools.
* **Relevant**: Essential for training and testing the deep learning model.
* **Time-bound**: Data collection and preprocessing to be completed within two months.

**To develop and implement a deep learning model for sentiment classification**

* **Specific**: Design a deep learning model using LSTM, CNN, or BERT for sentiment analysis.
* **Measurable**: Train the model on labelled social media datasets and achieve a minimum accuracy threshold of 80%.
* **Achievable**: Use Python-based libraries like TensorFlow and PyTorch.
* **Relevant**: Directly addresses the limitations of traditional sentiment analysis.
* **Time-bound**: Model development and training to be completed within three months.

**To evaluate the effectiveness of the developed sentiment analysis model**

* **Specific**: Assess model performance using precision, recall, F1-score, and accuracy metrics.
* **Measurable**: Compare the model with traditional machine learning methods.
* **Achievable**: Use standard evaluation datasets for benchmarking.
* **Relevant**: Ensures that the model is reliable and useful for businesses.
* **Time-bound**: Evaluation phase to be completed in one month.

**To analyse the impact of sentiment analysis on brand reputation management**

* **Specific**: Conduct case studies on businesses using sentiment analysis for reputation management.
* **Measurable**: Analyse at least three case studies and document the findings.
* **Achievable**: Use secondary data sources and interviews with industry professionals.
* **Relevant**: Demonstrates the practical application of the research findings.
* **Time-bound**: Case study analysis to be completed in the final month of the study.

**Research Approach and Methods**

To achieve these objectives, the research will adopt the following approaches:

**Data Collection**: Social media datasets will be gathered from platforms like Twitter, Reddit, and Facebook using open-source datasets and APIs.

**Data Preprocessing**: Techniques such as tokenisation, stopword removal, stemming, and lemmatisation will be used to clean and prepare text data.

**Model Development**: A deep learning model will be implemented using Python-based frameworks such as TensorFlow and PyTorch.

**Evaluation**: The model’s performance will be measured against traditional machine learning techniques using standard evaluation metrics.

**Case Study Analysis**: Real-world business applications of sentiment analysis will be examined to assess the impact on brand reputation management.

**Legal, Social, Ethical and Professional Considerations**

Since this research relies on secondary data from publicly available social media sources, the ethical considerations are minimal. However, certain legal, social, and ethical aspects must still be addressed to ensure responsible research practices.

One major concern is data privacy and user consent. While social media data is publicly accessible, it is essential to ensure compliance with data protection laws such as the General Data Protection Regulation (GDPR) and India’s Information Technology Act. No personally identifiable information (PII) will be collected or processed, and data will be anonymised to prevent privacy violations.

Another key issue is bias in sentiment analysis models. Deep learning models can inherit biases present in training data, leading to unfair or inaccurate sentiment classification. If a model misclassifies neutral or constructive criticism as negative, it may lead to misguided business decisions. To mitigate this, the research will use diverse and balanced datasets, apply bias detection techniques, and evaluate the model’s fairness.

From a professional standpoint, the findings of this study should be used responsibly by businesses. Automated sentiment analysis should complement, not replace, human judgment in brand reputation management. By following ethical AI principles, the research ensures fairness, transparency, and responsible use of sentiment analysis for business applications.

**Background**

**Introduction to Sentiment Analysis**

Sentiment analysis, also known as opinion mining, is a computational approach that leverages natural language processing (NLP) and machine learning to extract and classify sentiments from textual data. It enables businesses, policymakers, and researchers to gauge public perception, providing crucial insights into consumer behaviour and trends. In the realm of social media, sentiment analysis helps organizations monitor user sentiments toward products, brands, and global events, making it an invaluable tool for marketing and strategic planning [11].

Social media platforms generate massive amounts of user-generated content, offering a rich source of data for sentiment analysis. By applying computational models, businesses can assess consumer feedback, detect emerging trends, and respond proactively to customer concerns [12]. Furthermore, sentiment analysis plays a pivotal role in crisis management, allowing companies to identify and mitigate negative sentiment before it escalates. This ability to derive actionable insights from user sentiments is crucial for brand reputation management and customer engagement [13].

**Evolution of Sentiment Analysis Techniques**

Initial sentiment analysis techniques relied on lexicon-based and rule-based approaches, which involved predefined word lists and syntactic rules to classify sentiment. These methods were effective for simple tasks but struggled with context-dependent sentiment expressions and sarcasm [14]. While lexicon-based models performed well in structured domains, they were often limited by their inability to capture dynamic and evolving language patterns in social media content.

The advent of machine learning significantly improved sentiment classification by enabling models to learn patterns from labelled datasets rather than relying solely on pre-defined word lists. Traditional machine learning classifiers, such as Support Vector Machines (SVM) and Naïve Bayes, provided better generalization and adaptability to different domains. These methods marked a shift from manual rule-setting to data-driven learning, reducing the dependency on domain-specific lexicons [15].

With the rise of deep learning, sentiment analysis experienced a paradigm shift, with models such as Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks outperforming traditional approaches. These deep learning models improved sentiment classification by capturing complex dependencies within text data and leveraging word embeddings for contextual understanding [16]. Moreover, the integration of transformer-based architectures, such as BERT, has further enhanced the accuracy of sentiment classification by considering the nuances of language in social media texts [17].

**Machine Learning-Based Sentiment Analysis**

Traditional machine learning models such as Naïve Bayes, SVM, and Decision Trees have been widely used in sentiment analysis. Naïve Bayes, a probabilistic classifier, has been particularly effective due to its ability to handle high-dimensional text data efficiently. However, it assumes independence between features, which is often unrealistic in natural language processing [18].

SVM, on the other hand, is known for its robustness in text classification tasks, as it effectively separates classes using high-dimensional feature spaces. Despite its high accuracy, SVM is computationally expensive and requires significant tuning to perform optimally [19]. Decision Trees, while interpretable and computationally efficient, tend to overfit, especially in large and noisy datasets commonly found in social media analysis.

The effectiveness of machine learning-based sentiment analysis depends on the chosen model and its ability to generalize across different datasets. While traditional models like Naïve Bayes and SVM perform well on structured text, they struggle with the dynamic and informal nature of social media content, which often includes slang, sarcasm, and mixed sentiments [20]. This limitation has led to the adoption of deep learning models, which can extract high-level features automatically and improve sentiment classification accuracy.

Despite their advantages, deep learning models require substantial computational resources and large labeled datasets for effective training. Additionally, interpretability remains a challenge, as deep learning models function as "black boxes," making it difficult to understand their decision-making processes [21].

Feature engineering plays a crucial role in improving the performance of sentiment analysis models. Traditional approaches rely on feature extraction techniques such as Term Frequency-Inverse Document Frequency (TF-IDF) and Word2Vec, which convert textual data into numerical representations. Recent research has highlighted the effectiveness of transformer-based embeddings, such as BERT, in capturing contextual semantics, significantly enhancing sentiment classification performance [22].

As social media platforms continue to evolve, the complexity of sentiment expression increases, necessitating further advancements in sentiment analysis techniques. Hybrid approaches that combine lexicon-based, machine learning, and deep learning methods have emerged as a promising direction for improving sentiment classification accuracy [23].

**Deep Learning-Based Sentiment Analysis**

Deep learning models have revolutionized sentiment analysis by capturing contextual nuances in text and improving classification accuracy. Unlike traditional machine learning approaches, deep learning models do not require extensive feature engineering, allowing them to extract hierarchical representations from raw text data [24].

Recurrent Neural Networks (RNNs) have been widely used in sentiment analysis due to their ability to process sequential text data. However, RNNs suffer from vanishing gradient problems, limiting their ability to capture long-range dependencies. To address this, Long Short-Term Memory (LSTM) networks were introduced, which utilize gating mechanisms to retain information over extended sequences, making them effective for sentiment classification [25]. Convolutional Neural Networks (CNNs), traditionally used for image processing, have also been adapted for text classification by leveraging local feature extraction techniques. CNNs have shown competitive performance in sentiment analysis, particularly in capturing key phrases associated with sentiment polarity. However, they lack the ability to understand sequential dependencies compared to RNN-based models [26].

The introduction of transformer-based models, such as BERT and GPT, has significantly improved sentiment analysis by leveraging self-attention mechanisms to capture global dependencies within text. These models outperform traditional deep learning approaches by considering word context and enabling transfer learning, making them highly effective in real-world sentiment classification tasks [27].

Comparing deep learning models with traditional machine learning techniques reveals that while deep learning achieves superior accuracy, it requires extensive computational resources and large annotated datasets. Machine learning models, such as Support Vector Machines (SVM) and Naïve Bayes, remain relevant for smaller-scale sentiment analysis tasks due to their lower computational overhead [28].

**Applications of Sentiment Analysis in Brand Reputation Management**

Businesses leverage sentiment analysis to monitor brand reputation, assess customer feedback, and refine marketing strategies. Social media platforms serve as rich sources of user opinions, allowing companies to analyse customer sentiments in real time [29]. Several companies have integrated sentiment analysis into their reputation management strategies. For instance, major airlines use sentiment analysis to track passenger satisfaction and address complaints proactively. By analysing social media mentions and reviews, airlines can identify service-related issues and implement improvements [30]. Similarly, e-commerce platforms utilize sentiment analysis to monitor product reviews and optimize inventory decisions based on consumer feedback.

Real-time sentiment tracking allows businesses to respond promptly to emerging crises. For example, negative sentiment spikes on social media can indicate potential PR crises, prompting companies to issue timely responses and mitigate reputational damage. Advanced sentiment analysis models enhance crisis management by identifying sentiment shifts and predicting potential backlash [31].

**Challenges in Automating Sentiment Analysis for Businesses**

Despite its advantages, automating sentiment analysis presents several challenges. One of the primary difficulties is handling sarcasm, slang, and evolving language patterns. Social media users often express sentiments in a nuanced manner, making it challenging for sentiment analysis models to accurately interpret intent. Deep learning approaches have improved sarcasm detection, but further research is needed to enhance contextual understanding [32]. Ethical concerns and data privacy are critical considerations in sentiment analysis. Businesses must ensure that customer data is collected and analysed in compliance with privacy regulations. Additionally, bias in sentiment classification models can lead to skewed interpretations, affecting decision-making processes. Researchers have highlighted the importance of fairness-aware sentiment analysis models to mitigate bias and improve model transparency [33]. Processing large-scale social media data poses computational challenges. Deep learning-based sentiment analysis models require substantial processing power, making it difficult for small businesses to deploy real-time sentiment analysis solutions. Cloud-based and distributed computing frameworks have been proposed to address scalability issues and improve real-time sentiment classification [34].

**Summary and Research Gaps**

This review highlights the evolution of sentiment analysis techniques, emphasizing the superiority of deep learning models in capturing contextual nuances. Businesses increasingly use sentiment analysis for brand monitoring, but challenges such as sarcasm detection, bias mitigation, and scalability remain unresolved. Future research should focus on developing interpretable and resource-efficient sentiment analysis models to enhance applicability across diverse business domains. Addressing these gaps will improve the robustness and fairness of automated sentiment analysis systems.

References

|  |  |
| --- | --- |
| [1] | Y. Zhan, R. Liu, Q. Li, S. J. Leischow, and D. D. Zeng, “Identifying topics for E-cigarette user-generated contents: A case study from multiple social media platforms,” *J. Med. Internet Res.*, vol. 19, no. 1, p. e24, 2017. |
| [2] | F. Hemmatian and M. K. Sohrabi, “A survey on classification techniques for opinion mining and sentiment analysis,” *Artif. Intell. Rev.*, vol. 52, no. 3, pp. 1495–1545, 2019. |
| [3] | J.-B. E. M. Steenkamp, “Global brand building and management in the digital age,” *J. Int. Mark.*, vol. 28, no. 1, pp. 13–27, 2020. |
| [4] | N. A. Sharma, A. B. M. S. Ali, and M. A. Kabir, “A review of sentiment analysis: tasks, applications, and deep learning techniques,” *Int. J. Data Sci. Anal.*, 2024. |
| [5] | C. Sahoo, M. Wankhade, and B. K. Singh, “Sentiment analysis using deep learning techniques: a comprehensive review,” *Int. J. Multimed. Inf. Retr.*, vol. 12, no. 2, 2023. |
| [6] | Y. K. Dwivedi *et al.*, “Setting the future of digital and social media marketing research: Perspectives and research propositions,” *Int. J. Inf. Manage.*, vol. 59, no. 102168, p. 102168, 2021. |
| [7] | R. T. Rust, W. Rand, M.-H. Huang, A. T. Stephen, G. Brooks, and T. Chabuk, “Real-time brand reputation tracking using social media,” *J. Mark.*, vol. 85, no. 4, pp. 21–43, 2021. |
| [8] | A. Shaji George and T. Baskar, “Leveraging big data and sentiment analysis for actionable insights: A review of data mining approaches for social media.” PU Publications, 2024. |
| [9] | S. F. Ahmed *et al.*, “Deep learning modelling techniques: current progress, applications, advantages, and challenges,” *Artif. Intell. Rev.*, vol. 56, no. 11, pp. 13521–13617, 2023. |
| [10] | D. Godoy, “Are online harm spreaders birds of the same feather? A multi-dimensional study on the characteristics of social media harm spreaders,” *Soc. Netw. Anal. Min.*, vol. 14, no. 1, 2024. |
| [11] | F. Neri, C. Aliprandi, F. Capeci, M. Cuadros, and T. By, “Sentiment Analysis on Social Media,” *2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pp. 919–926, 2012. |
| [12] | S. Gupta and R. Sandhane, “Use of sentiment analysis in social media campaign design and analysis,” *Cardiometry*, 2022. |
| [13] | V. Singh, H. V. Kaushik, and Reshma, “Social Media Sentiment Analysis Using Twitter Dataset,” in *2024 Second International Conference on Data Science and Information System (ICDSIS)*, 2024, pp. 1–5. |
| [14] | M. Birjali, M. Kasri, and A. Beni-Hssane, “A comprehensive survey on sentiment analysis: Approaches, challenges and trends,” *Knowl. Based Syst.*, vol. 226, no. 107134, p. 107134, 2021. |
| [15] | S. K. Farogh, “A comprehensive evaluation and comparative analysis of data mining techniques for sentiment analysis in social media,” *International Journal of Advanced Research in Science, Communication and Technology*, pp. 40–45, 2023. |
| [16] | Y. Ashrita, S. Abhiram, V. Hemanth, A. Srinivas, and P. R. Vemula, “Deep learning techniques for sentiment analysis on social media text,” in *2023 6th International Conference on Contemporary Computing and Informatics (IC3I)*, 2023. |
| [17] | H.-H. Nguyen, “Enhancing sentiment analysis on social media data with advanced deep learning techniques,” *Int. J. Adv. Comput. Sci. Appl.*, vol. 15, no. 5, 2024. |
| [18] | M. Kavitha, B. B. Naib, B. Mallikarjuna, R. Kavitha, and R. Srinivasan, “Sentiment analysis using NLP and machine learning techniques on social media data,” in *2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, 2022, pp. 112–115. |
| [19] | S. M. Pardeshi, D. Chandra Jain, and R. B. Wagh, “A qualified study of machine learning algorithms for sentiment analysis in social media,” in *2023 10th IEEE Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON)*, 2023. |
| [20] | G. S. Talwar, B. K. Saini, N. Kaur, B. Kaur, and P. K. Sarao, “A machine learning approach to Analyse sentiment on social media,” in *2023 2nd International Conference on Futuristic Technologies (INCOFT)*, 2023. |
| [21] | S. S. Suganya, B. Narmadha, Mukeshkumar, and Mugeesha, “Sentiment Analysis using Deep Learning based Emotion Extraction from Social Media Texts,” in *2023 International Conference on Sustainable Communication Networks and Application (ICSCNA)*, 2023, pp. 1581–1586. |
| [22] | X. Li *et al.*, “Enhancing BERT representation with context-aware embedding for aspect-based sentiment analysis,” *IEEE Access*, vol. 8, pp. 46868–46876, 2020. |
| [23] | S. L. V. Durga, A. R, N. Singh, A. Rana, R. Kalra, and S. A. Al Khidhir Abdullah, “Mining social media data for sentiment analysis and trend prediction,” in *2023 10th IEEE Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON)*, 2023. |
| [24] | A. Kumar, N. Gupta, and A. Shrivastava, “A critical review on sentiment analysis based on deep learning techniques,” *International Journal For Multidisciplinary Research*, vol. 6, no. 5, 2024. |
| [25] | Q. Liu and X. Wang, “The application of deep learning-based natural language processing models in sentiment analysis,” in *2024 5th International Conference on Electronic Communication and Artificial Intelligence (ICECAI)*, 2024, pp. 686–689. |
| [26] | M. E. Basiri, S. Nemati, M. Abdar, E. Cambria, and U. R. Acharya, “ABCDM: An Attention-based Bidirectional CNN-RNN Deep Model for sentiment analysis,” *Future Gener. Comput. Syst.*, vol. 115, pp. 279–294, 2021. |
| [27] | S. Sruthi, V. G. Trinath, V. Jayanth, V. P. Balaji, T. Singh, and A. Mandal, “Natural language processing for sentiment analysis with deep learning,” in *2024 3rd International Conference for Innovation in Technology (INOCON)*, 2024. |
| [28] | F. Arias, M. Zambrano Nunez, A. Guerra-Adames, N. Tejedor-Flores, and M. Vargas-Lombardo, “Sentiment analysis of public social media as a tool for health-related topics,” *IEEE Access*, vol. 10, pp. 74850–74872, 2022. |
| [29] | L. Cao, “Sentiment analysis of social media text based on deep learning,” in *2023 3rd International Conference on Mobile Networks and Wireless Communications (ICMNWC)*, 2023. |
| [30] | K. L. Tan, C. P. Lee, K. M. Lim, and K. S. M. Anbananthen, “Sentiment analysis with ensemble hybrid deep learning model,” *IEEE Access*, vol. 10, pp. 103694–103704, 2022. |
| [31] | S. B. Garg and V. V. Subrahmanyam, “Super deep learning ensemble model for sentiment analysis,” in *2023 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, 2023. |
| [32] | S. Anis, M. M. Morsey, and M. Aref, “Fintech sentiment analysis using deep learning approaches: A survey,” in *2024 5th International Conference on Artificial Intelligence, Robotics and Control (AIRC)*, 2024, pp. 118–122. |
| [33] | Q. Zhao, H. Zhang, and J. Shang, “Interpretable sentiment analysis based on sentiment words’ syntax information,” in *2022 International Conference on Industrial Automation, Robotics and Control Engineering (IARCE)*, 2022. |
| [34] | Y. Luo, Y. Wang, Y. Xiong, and C. Yu, “Deep learning techniques for multimodal sentiment analysis,” in *2022 IEEE 2nd International Conference on Electronic Technology, Communication and Information (ICETCI)*, 2022. |

Appendix B: Project Management

Gantt Chart

**A graph with blue rectangles

AI-generated content may be incorrect.**

Appendix C: Artefact/Dataset

**Data Source Link:** <https://www.kaggle.com/datasets/crowdflower/twitter-airline-sentiment>

**Github:** [**https://github.com/Aswin132639/Airline\_Twitter\_Data\_sentiment\_analysis**](https://github.com/Aswin132639/Airline_Twitter_Data_sentiment_analysis)

**Platform Used Google Colab:** [**https://colab.research.google.com/drive/1quE59NZMGI-BDMxaIe3Dw\_S3sdgaGcSc?usp=sharing**](https://colab.research.google.com/drive/1quE59NZMGI-BDMxaIe3Dw_S3sdgaGcSc?usp=sharing)

**Link:**

**Code:**

import random

import numpy as np

import tensorflow as tf

# Set seeds for reproducibility

random.seed(42)

np.random.seed(42)

tf.random.set\_seed(42)

!pip install nltk

!pip install transformers

!pip install tensorflow

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

import nltk

from nltk.corpus import stopwords

from nltk.stem import PorterStemmer

from nltk.tokenize import word\_tokenize

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import classification\_report, confusion\_matrix, accuracy\_score

import tensorflow as tf

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Embedding, LSTM, Dense, Dropout, Bidirectional

from tensorflow.keras.preprocessing.text import Tokenizer

from tensorflow.keras.preprocessing.sequence import pad\_sequences

from transformers import BertTokenizer, TFBertForSequenceClassification

nltk.download('punkt')

nltk.download('stopwords')

data = pd.read\_csv("/content/Tweets.csv")

data.head()

# Shape of the dataset (how many rows and columns)

print("Dataset shape:", data.shape)

# General information about columns, non-null counts, datatypes

data.info()

# See first 5 rows to understand data structure

data.head()

# See last 5 rows

data.tail()

# Total missing values per column

print("Missing values per column:")

print(data.isnull().sum())

# Percentage of missing values

missing\_percent = (data.isnull().sum() / len(data)) \* 100

print("\nMissing values percentage:")

print(missing\_percent)

# Check duplicate rows

duplicate\_rows = data.duplicated().sum()

print("Number of duplicate rows:", duplicate\_rows)

# If needed, remove duplicates

# data = data.drop\_duplicates()

# Count of each sentiment

print(data['airline\_sentiment'].value\_counts())

# Plot the distribution

sns.countplot(x='airline\_sentiment', data=data, palette='pastel')

plt.title('Distribution of Sentiments')

plt.xlabel('Sentiment')

plt.ylabel('Count')

plt.show()

sentiment\_percent = data['airline\_sentiment'].value\_counts(normalize=True) \* 100

print(sentiment\_percent)

# Cross table

sentiment\_airline = pd.crosstab(data['airline'], data['airline\_sentiment'])

# Plot it

sentiment\_airline.plot(kind='bar', stacked=True, figsize=(12,7), colormap='viridis')

plt.title('Sentiment count per Airline')

plt.ylabel('Number of Tweets')

plt.xlabel('Airline')

plt.xticks(rotation=45)

plt.show()

# Only keep negative tweets

negative\_tweets = data[data['airline\_sentiment'] == 'negative']

# Check reasons

print(negative\_tweets['negativereason'].value\_counts())

# Plot

sns.countplot(y='negativereason', data=negative\_tweets, order=negative\_tweets['negativereason'].value\_counts().index)

plt.title('Reasons for Negative Sentiments')

plt.xlabel('Number of Tweets')

plt.ylabel('Negative Reason')

plt.show()

# Create a new column for tweet length

data['text\_length'] = data['text'].apply(len)

# Plot distribution

sns.histplot(data=data, x='text\_length', hue='airline\_sentiment', bins=30, kde=True)

plt.title('Tweet Length Distribution by Sentiment')

plt.xlabel('Tweet Length')

plt.ylabel('Number of Tweets')

plt.show()

# Check average length by sentiment

print(data.groupby('airline\_sentiment')['text\_length'].mean())

!pip install wordcloud

from wordcloud import WordCloud

# Positive Tweets Wordcloud

positive\_words = " ".join(data[data['airline\_sentiment']=='positive']['text'])

wordcloud\_positive = WordCloud(width=800, height=500, random\_state=21, max\_font\_size=110).generate(positive\_words)

plt.figure(figsize=(10,7))

plt.imshow(wordcloud\_positive, interpolation="bilinear")

plt.axis('off')

plt.title('Word Cloud for Positive Tweets')

plt.show()

# Negative Tweets Wordcloud

negative\_words = " ".join(data[data['airline\_sentiment']=='negative']['text'])

wordcloud\_negative = WordCloud(width=800, height=500, random\_state=21, max\_font\_size=110).generate(negative\_words)

plt.figure(figsize=(10,7))

plt.imshow(wordcloud\_negative, interpolation="bilinear")

plt.axis('off')

plt.title('Word Cloud for Negative Tweets')

plt.show()

# Convert tweet\_created column to datetime

data['tweet\_created'] = pd.to\_datetime(data['tweet\_created'])

# Set time as index (optional)

# data.set\_index('tweet\_created', inplace=True)

# Plot number of tweets by time

data['tweet\_created'].dt.hour.value\_counts().sort\_index().plot(kind='bar', figsize=(10,6))

plt.title('Number of Tweets by Hour')

plt.xlabel('Hour of Day')

plt.ylabel('Number of Tweets')

plt.show()

# Keep only 'text' and 'airline\_sentiment'

data = data[['text', 'airline\_sentiment']]

import re

def clean\_text(text):

text = text.lower() # Lowercase all text

text = re.sub(r'http\S+', '', text) # Remove URLs

text = re.sub(r'www.\S+', '', text) # Remove websites

text = re.sub(r'@\w+', '', text) # Remove mentions (@airline)

text = re.sub(r'#\w+', '', text) # Remove hashtags

text = re.sub(r'\d+', '', text) # Remove numbers

text = re.sub(r'[^a-z\s]', '', text) # Remove punctuations and special characters

text = re.sub(r'\s+', ' ', text).strip() # Remove extra spaces

return text

# Apply cleaning function

data['clean\_text'] = data['text'].apply(clean\_text)

import nltk

from nltk.corpus import stopwords

from nltk.stem import PorterStemmer

from nltk.tokenize import word\_tokenize

# Download required resources properly

nltk.download('punkt')

nltk.download('punkt\_tab')

nltk.download('stopwords')

nltk.download('wordnet')

nltk.download('omw-1.4')

nltk.download('averaged\_perceptron\_tagger')

# Preprocessing Functions

stop\_words = set(stopwords.words('english'))

stemmer = PorterStemmer()

def preprocess(text):

words = word\_tokenize(text) # Break into words

words = [stemmer.stem(word) for word in words if word not in stop\_words] # Remove stopwords, apply stemming

return " ".join(words)

# Apply preprocessing

data['processed\_text'] = data['clean\_text'].apply(preprocess)

def remove\_short\_words(text):

words = text.split()

words = [word for word in words if len(word) > 2]

return " ".join(words)

data['processed\_text'] = data['processed\_text'].apply(remove\_short\_words)

# Map sentiments to numbers

sentiment\_mapping = {'negative': 0, 'neutral': 1, 'positive': 2}

data['label'] = data['airline\_sentiment'].map(sentiment\_mapping)

data[['text', 'clean\_text', 'processed\_text', 'airline\_sentiment', 'label']].head()

# Mapping sentiments to numbers

sentiment\_mapping = {'negative': 0, 'neutral': 1, 'positive': 2}

data['label'] = data['airline\_sentiment'].map(sentiment\_mapping)

# Check few rows

data[['airline\_sentiment', 'label']].head()

# Features and labels

X = data['processed\_text'] # Input tweets

y = data['label'] # Sentiment labels (0, 1, 2)

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(

X, y, test\_size=0.2, random\_state=42, stratify=y

)

from tensorflow.keras.preprocessing.text import Tokenizer

# Create the Tokenizer

tokenizer = Tokenizer(num\_words=5000, oov\_token="<OOV>")

# num\_words = how many most common words to keep

# oov\_token = token for unknown words

# Fit the tokenizer on training text

tokenizer.fit\_on\_texts(X\_train)

# Convert text to sequences

X\_train\_seq = tokenizer.texts\_to\_sequences(X\_train)

X\_test\_seq = tokenizer.texts\_to\_sequences(X\_test)

# Check example

print(X\_train.iloc[0])

print(X\_train\_seq[0])

from tensorflow.keras.preprocessing.sequence import pad\_sequences

# Padding

X\_train\_pad = pad\_sequences(X\_train\_seq, maxlen=100, padding='post')

X\_test\_pad = pad\_sequences(X\_test\_seq, maxlen=100, padding='post')

# Check shapes

print(X\_train\_pad.shape)

print(X\_test\_pad.shape)

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Embedding, LSTM, Dense, Dropout, Bidirectional

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Embedding, LSTM, Dense, Dropout, Bidirectional

# Create the model

model = Sequential()

# 1. Embedding Layer (NO input\_length needed)

model.add(Embedding(input\_dim=5000, output\_dim=64))

# 2. LSTM Layer

model.add(Bidirectional(LSTM(64)))

# 3. Dropout Layer

model.add(Dropout(0.5))

# 4. Output Layer

model.add(Dense(3, activation='softmax'))

# Compile the model

model.compile(loss='sparse\_categorical\_crossentropy', optimizer='adam', metrics=['accuracy'])

# OPTIONAL: Build manually to see good summary (not necessary)

model.build(input\_shape=(None, 100))

# Show model summary

model.summary()

history = model.fit(

X\_train\_pad,

y\_train,

epochs=5, # Number of times model sees full data

batch\_size=32, # Number of samples seen before updating weights

validation\_data=(X\_test\_pad, y\_test)

)

import matplotlib.pyplot as plt

# Accuracy plot

plt.plot(history.history['accuracy'], label='Training Accuracy')

plt.plot(history.history['val\_accuracy'], label='Validation Accuracy')

plt.title('Training and Validation Accuracy')

plt.xlabel('Epoch')

plt.ylabel('Accuracy')

plt.legend()

plt.show()

# Loss plot

plt.plot(history.history['loss'], label='Training Loss')

plt.plot(history.history['val\_loss'], label='Validation Loss')

plt.title('Training and Validation Loss')

plt.xlabel('Epoch')

plt.ylabel('Loss')

plt.legend()

plt.show()

# Save model (NEW recommended format)

model.save('/content/airline\_sentiment\_lstm.keras')

# Load model

from tensorflow.keras.models import load\_model

model = load\_model('/content/airline\_sentiment\_lstm.keras')

# Evaluate or Predict

loss, accuracy = model.evaluate(X\_test\_pad, y\_test)

print(f"Test Accuracy: {accuracy:.2f}")

# Make predictions

y\_pred = model.predict(X\_test\_pad)

# Convert predicted probabilities into class labels

y\_pred\_classes = np.argmax(y\_pred, axis=1)

from sklearn.metrics import accuracy\_score

# Calculate accuracy

accuracy = accuracy\_score(y\_test, y\_pred\_classes)

print(f"Test Accuracy: {accuracy:.2f}")

from sklearn.metrics import classification\_report

# Generate classification report

print(classification\_report(y\_test, y\_pred\_classes, target\_names=['Negative', 'Neutral', 'Positive']))

from sklearn.metrics import confusion\_matrix

import seaborn as sns

import matplotlib.pyplot as plt

# Create confusion matrix

cm = confusion\_matrix(y\_test, y\_pred\_classes)

# Plot heatmap

plt.figure(figsize=(6,5))

sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=['Negative', 'Neutral', 'Positive'], yticklabels=['Negative', 'Neutral', 'Positive'])

plt.xlabel('Predicted Labels')

plt.ylabel('True Labels')

plt.title('Confusion Matrix')

plt.show()

# Show confusion matrix with labels

import pandas as pd

labels = ['Negative', 'Neutral', 'Positive']

cm\_df = pd.DataFrame(cm, index=labels, columns=labels)

print(cm\_df)

# Install transformers if not done

!pip install transformers

# Imports

from transformers import BertTokenizer, TFBertForSequenceClassification, create\_optimizer

import tensorflow as tf

# Load tokenizer and model

bert\_tokenizer = BertTokenizer.from\_pretrained('bert-base-uncased')

bert\_model = TFBertForSequenceClassification.from\_pretrained('bert-base-uncased', num\_labels=3)

# Define function to tokenize data properly for BERT

def tokenize\_data(texts, tokenizer, max\_len=128):

return tokenizer(

list(texts),

max\_length=max\_len,

truncation=True,

padding=True,

return\_tensors='tf'

)

# Tokenize both training and testing sets

train\_encodings = tokenize\_data(X\_train, bert\_tokenizer)

test\_encodings = tokenize\_data(X\_test, bert\_tokenizer)

# Create optimizer

num\_train\_steps = int(len(X\_train) / 16) \* 3 # batch size 16, epochs 3

optimizer, lr\_schedule = create\_optimizer(

init\_lr=2e-5,

num\_warmup\_steps=0,

num\_train\_steps=num\_train\_steps

)

# Compile model

bert\_model.compile(

optimizer=optimizer,

loss=tf.keras.losses.SparseCategoricalCrossentropy(from\_logits=True),

metrics=['accuracy']

)

# Train BERT

bert\_model.fit(

x=train\_encodings['input\_ids'],

y=y\_train,

validation\_data=(test\_encodings['input\_ids'], y\_test),

epochs=3,

batch\_size=16

)

# Predict on test set

y\_bert\_pred = bert\_model.predict(test\_encodings['input\_ids'])

# Convert logits to predicted class

y\_bert\_pred\_classes = tf.argmax(y\_bert\_pred.logits, axis=1)

# Import libraries for evaluation

from sklearn.metrics import classification\_report, confusion\_matrix

import numpy as np

# Generate classification report

print(classification\_report(y\_test, y\_bert\_pred\_classes, target\_names=['Negative', 'Neutral', 'Positive']))

import seaborn as sns

import matplotlib.pyplot as plt

# Confusion matrix

cm = confusion\_matrix(y\_test, y\_bert\_pred\_classes)

# Plot it

plt.figure(figsize=(6,5))

sns.heatmap(cm, annot=True, fmt='d', cmap='Greens', xticklabels=['Negative', 'Neutral', 'Positive'], yticklabels=['Negative', 'Neutral', 'Positive'])

plt.xlabel('Predicted')

plt.ylabel('Actual')

plt.title('Confusion Matrix for BERT')

plt.show()

Appendix D: Screencast

Youtube link: <https://youtu.be/VptScq_xCRo?feature=shared>