Comparative Evaluation of Deep Neural Networks and Classical Machine Learning for Airline Sentiment Analysis Using Twitter and Word2Vec Embeddings

Yifan Zhang
School of International Studies, Xi'an Jiaotong University City College, Xi'an 710018, Shaanxi, China
E-mail: yifanzhangzyf91@yeah.net

Keywords: deep neural network, sentiment analysis, machine learning, word2vec word embedding, TF-IDF, and XGBClassifier

Received: June 3, 2025

This study uses Twitter data to investigate sentiment analysis in the airline sector, which compares deep neural networks with traditional machine learning approaches. A publicly available dataset of approximately 14,000 labeled airline tweets is preprocessed using tokenization, stop-word removal, and lemmatization to prepare textual data for analysis. Two feature representation strategies are examined: term frequency–inverse document frequency (TF-IDF) vectors for classical models and Word2Vec embeddings for deep learning models. The experiments use an 80/20 train-test split with stratified sampling to maintain class distribution. Traditional classifiers, which includes Logistic Regression, Support Vector Machines, and Random Forests, are benchmarked against Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) models with embedding layers. Model performance is evaluated using accuracy, precision, recall, and F1-score, with paired t-tests confirming the statistical significance of differences between approaches. The best deep learning configuration, based on Word2Vec embeddings, achieves 88% accuracy and an F1-score of 0.86, which outperforms all TF-IDF-based traditional classifiers. These results demonstrate that embedding-based deep neural networks more effectively capture the semantic nuances of short social media texts, which offer enhanced reliability for sentiment detection in the airline sector.

Povzetek: Študija pokaže, da globoki nevronski modeli z vdelavami besed bolje zaznavajo sentiment v tvitih o letalskih družbah kot klasični pristopi strojnega učenja.

1 Introduction

Over the last 20 years, there have been substantial changes in the airline sector, especially in how consumer feedback is collected and processed. The airline service industry faces challenges in collecting customer feedback through traditional methods like questionnaires. The airline sector, a fast-increasing business, has transitioned past conventional feedback from consumer forms to Twitter data for sentiment analysis since traditional approaches are time-consuming and inefficient[1]. Globally, the internet is a major factor in decision-making, and many people use blogs, social media, and other online forums to share their opinions. The critical barrier to the search for targeted information is the quantity of information available, which runs the gamut from relevant to irrelevant data. This is principally valuable for consumers making decisions about the best airlines, as sentiment analysis allows the assessment of customer reviews on platforms such as Skytrax and microblogging sites like Twitter, which provides insights that aid in more informed decision-making [2]. Sentiment analysis has, therefore, emerged as one such tool to smooth this process by taking immediate stock of customer opinions and redressing the grievances expressed. However, because of the immediacy and volume of opinions shared by its consumers, Twitter has developed as a reliable platform for sentiment analysis. In spite of its potential, researches focusing on sentiment classification in the airline service sector on Twitter remain restricted. This gap indicates an opportunity for researchers to discover and develop more effective sentiment analysis approaches tailored precisely to the unique characteristics of the airline industry and Twitter data [3].

In the modern marketplace, the airline industry is a main player, where opinion mining is vital for maintaining relevance and competitiveness among airlines. Opinion mining allows the analysis of customer feedback across reviews, social media, and further platforms. These insights are important for airlines to make knowledgeable decisions, enhance customer satisfaction, and refine overall business strategies [4]. In the competitive business world, client satisfaction is vital for the growth of an organization. Companies take much pain and interest in knowing and satisfying client needs. However, the traditional manual analysis fails to satisfy consumers in most cases, which results in less client loyalty and more marketing expenses. Sentiment analysis, integrating NLP with ML techniques, provides a solution to extract insight

from public views and opinions about different topics, products, and services using online data. This might help the firm understand customers' opinions and work out the strategy in the right direction [5].

Twitter is among the popular social media systems that enable users to share their opinions about anything, including ideas, products, and services. Such information may help to improve product or service quality. In the airline company, it will be very helpful because business managers will be in a position to check consumer satisfaction and work towards improving the flying experience for passengers. Therefore, the current study has used sentiment analysis techniques to systematically classify tweets from each airline into classifications of positive, negative, and neutral polarity. This approach was applied to a study about six major US airlines and returned significant information on the sentiment of customers about those specific airlines [6].

The following provides an account of earlier studies regarding sentiment classification regarding major US airlines. The most prominent predictive frameworks are illustrated, and the merits and demerits of various prediction frameworks are presented. The investigation carried out by Wang and Gao [3] on sentiment classification in airline services was classified using an ensemble approach based on multi-classification methods. These were NB, DT, Bayesian Network, C4.5, SVM, and RF. The database comprised 12,864 tweets on airline services. The results represented that such an ensemble model performed better than individual classifiers and six other classification approaches. This proposes that ensemble approaches seem effective in enhancing the robustness and accuracy of sentiment classification for tweets about airline services.

Rane & Kumar [1] investigated the multi-class sentiment analysis of the tweets that is posted by six major US airlines. DT, RF, SVM, KNN, LR, Gaussian NB, and AdaBoost were among the seven classification strategies used in this analysis, which mainly relied on pre-processing methods and deep learning (DL) concepts for sentiment analysis of the tweets. Consequently, tweet sentiments were grouped into three groups: positive, negative, and neutral. The outcomes of the evaluation of accuracy for each classification strategy allowed the comparison of their performances. The overall sentiment count of all six airlines was fetched and visualized, providing a broad overview of sentiment distribution across the database.

Using AdaBoost for sentiment analysis has been improved using a research methodology in Prabhakar et al.[2] Work was conducted on selecting and analyzing algorithm usage based on various ML frameworks. To support the effectiveness of the suggested AdaBoost in sentiment analysis, the Confusion Matrix and accuracy measures were employed. Rustam et al.[7] proposed a stochastic gradient descent classifier and an LR-based VC for some sentiment analysis in organizations. The venture capitalist employed the soft voting technique to forecast the emotions of tweets. Test data assessed the ML classifiers' classification in terms of $F1$ score, recall, accuracy, and precision using the TF, $TF\text{-}IDF$, and word2vec characteristics. The VC performed better than the other classifiers, achieving 0.789 and 0.791 accuracy in terms of TF or $TF\text{-}IDF$ attribute derivation, respectively. The ensemble classifiers were suggested to have higher accuracy than non-ensemble classifiers, while ML classifiers performed better when TF-IDF was applied as attribute derivation. In that regard, word2vec attribute derivation displayed low performance in comparison with both TF and TF-IDF, and even lower accuracy than ML classifiers was achieved with LSTM. Monika et al.[8] experimented with deep learning methods to identify sentiment polarity in tweets, where word embedding frameworks like Word2Vec and GloVe were adopted. They applied RNN frameworks and LSTM for their analysis. In terms of classification accuracy results, training databases account for 80% and testing databases for 20%. This indicates the reliability of their recommended frameworks concerning prediction tasks. With the aid of the Bi-LSTM model, an additional study was conducted, concluding a significant body of research on sophisticated deep-learning methods for sentiment analysis in tweets.

Often called opinion mining, Saad [4] is in charge of examining consumer reviews of airline services, with Twitter serving as a significant data source. The study suggested an ML framework that could classify tweets into three categories: positive, negative, and neutral. The methodology contains the pre-processing, cleaning, and extracting attributes from a database containing tweets from six US airlines. The model was constructed with the Bag of Words (BoW) paradigm, and six frameworks were tested: RF, SVM, LR, DT, NB, and XGB. K-Fold Cross-Validation was used to further divide the database into 30% testing and 70% training. Each classifier's performance measures were precision, recall, $F1$-score, and accuracy. SVM achieved the maximum accuracy of 83.31%, according to the results, indicating the method's applicability for tweets related to airline sentiment analysis.

Hasib et al. (2021) [9] studied tweets from six major US airlines with multi-class sentiment analysis and a DL framework that combines DL and word embedding techniques. This model lets tweets be classified as neutral, negative, or positive within a three-class database, and it combines CNN's tweet-cleaning pre-processing techniques with raw DNN data extraction. By their metrics, the model performed well with better outcomes than existing frameworks, thus establishing its persuasive power in sentiment analysis in airline-related tweets. Tusar and Islam (2021) [10]investigated a sentiment analysis method that applied NLP methods like $TF\text{-}IDF$ and BoW along with ML classification frameworks like RF, LR, SVM, and multinomial NB. Their most competitive approach, which reach an accuracy of 77%, which utilize LR with the BoW technique.

According to Aljedaani et al., 2022 [6] is a revolutionary hybrid method to sentiment analysis that improves sentiment accuracy by fusing DL frameworks with lexicon-based techniques. The goal of the study was to fully comprehend the potential for misleading

annotations while analyzing how TextBlob improved model classification accuracy when compared to original annotations. A complete assessment was carried out here in comparison to Afinn and VADER against several compositions such as CNN, LSTM, GRU, CNN-LSTM, TF-IDF, and BoW. The results showed that TextBlob sentiments increased the productivity of the frameworks when compared to the original sentiments, and that the LSTM-GRU model outperformed all other frameworks and previous studies, achieving the highest accuracy and F1 scores. When applied to TF-IDF in conjunction with BoW, the support vector classifier and the extra tree classifier produced higher accuracy scores. While these findings are promising, the authors warn against the wholesale substitution of human annotators with TextBlob-based annotation, owing to biases, propensities for mistakes, and the inevitable subjectivity of the exercise. Rather, the authors suggest TextBlob-annotated labels would then serve to assist the human annotator in its endeavor to refine the TextBlob-annotated database, therefore improving the quality of sentiment analysis.

The research discussed in this part had a significant influence on the airline industry's sentiment analysis sector. They showcased the usefulness of various ML and DL frameworks in accurately and credibly classifying the sentiment expressed in Twitter tweets regarding airline services. Some of these frameworks are among the most used in their studies, including SVM, LR, RF, NB, DT, and ensemble methods. While SVM and ensemble methods show real promise and frequently outperform other classifiers, there are diverse limitations associated with these classifiers. The fact that the analyses have been general and have not been targeted at specific features or portions of airline services is one of the primary restrictions of this research utilizing sentiment analysis on the airline industry, principally with Twitter data. Although sentiment analysis may categorize tweets as neutral, negative, or positive by looking at them all at once without concentrating on specific elements of the whole travel experience, it is unable to identify the specific element of airline services that has sparked such feelings. For instance, a tweet might express negative sentiment due to a flight delay, yet sentiment analysis would not really provide insights into what caused the delay or its effects on customer satisfaction. Besides, using Twitter data may have introduced other biases in evaluating sentiments since not all airline customers would express their dissent on Twitter. Such variabilities could, therefore, distort the reflection of customer sentiments, especially as certain demographics among airline customers are generally underrepresented on Twitter compared with the general population of airline customers. In addition, the productivity of sentiment analysis frameworks may be impacted by the various formats in which users post tweets and the dynamic nature of language. Slang, sarcasm, and cultural nuances make it exceedingly difficult for sentiment classification frameworks to classify sentiments accurately, possibly leading to misclassification of the original tweet.

In this study, deep neural network architectures were employed to enhance sentiment classification tasks and compared with traditional ML frameworks. The approach involved using Word2Vec embeddings to convert each tweet into a numerical vector, capturing its semantic content effectively. These embeddings were used exclusively in the deep neural network models. In contrast, traditional frameworks solely relied on TF-IDF vectorization. However, both word embedding and $TF\text{-}IDF$ were used in deep neural network systems. The parameters of the traditional frameworks were kept unchanged, maintaining their default configurations. A diverse array of ML frameworks, including XGBClassifier, LGBMClassifier, ExtraTreeClassifier, AdaBoostClassifier, BernoulliNB, and NearestCentroid, were assessed alongside neural network architectures. Comparison of sentiment analysis methods for airline twitter data is indicated in Table 1.

The upcoming sections of this research are outlined as follows:

Part 2 encompasses database analysis, methodology overview, performance assessment, and a comparative analysis of foundational and classification methodologies. Section 3 compares foundational techniques and classification methodologies. Section 4 provides a concise summary of key findings, contributions to the field, implications, and future directions.

1.1 Research design and goals

This study aims to evaluate the performance of deep learning models (CNN and LSTM) versus traditional machine learning classifiers (SVM and Logistic Regression) for sentiment analysis of airline-related tweets. A secondary goal is to assess the impact of Word2Vec embeddings compared to TF-IDF feature extraction. The research addresses the following questions:

Do deep learning models outperform traditional classifiers in sentiment analysis of Twitter data?

Does Word2Vec improve sentiment classification performance compared to TF-IDF?

The study design includes the following steps:

Data collection and preprocessing: A dataset of 14,000 labeled tweets was collected, preprocessed with tokenization, stop-word removal, and lemmatization.

Feature extraction: Tweets were represented using TF-IDF for traditional models and Word2Vec embeddings for deep learning models, either pre-trained or learned during training.

Model training and evaluation: CNN, LSTM, SVM, and Logistic Regression models were trained and evaluated on multiple metrics, including accuracy, precision, recall, and F1-score.

Statistical analysis: Paired t-tests were used to compare model performance and confirm statistical significance.

The results aim to provide insights into the advantages of deep learning techniques in sentiment analysis, especially for the airline sector, where public sentiment plays a crucial role in customer satisfaction.

Table 1: Comparison of sentiment analysis methods for airline twitter data

Study	Classification Methodology	Model Used	Dataset	Performance Metrics	Key Findings
Wang & Gao [3]	Ensemble approach (NB, SVM, Bayesian Network, C4.5, DT, RF)	Multi-class classification	12,864 tweets	Accuracy, Precision, Recall, F1	Ensemble models outperformed individual classifiers, improving robustness and accuracy.
Rane & Kumar [1]	Multi-class sentiment analysis	DT, RF, SVM, KNN, LR, Gaussian NB, AdaBoost	Tweets from six US airlines	Accuracy, F1-score	AdaBoost performed well, with varying results across classifiers.
Prabhakar et al. [2]	Sentiment analysis using AdaBoost	AdaBoost	Airline tweets	Confusion Matrix, Accuracy	AdaBoost achieved strong results in accuracy and confusion matrix evaluation.
Rustam et al. [7]	Stochastic Gradient Descent & LR-based VC	Soft voting technique	Tweets from organizations	F1-score, Recall, Accuracy, Precision	VC performed best with TF-IDF and Word2Vec, outpacing other classifiers.
Monika et al. [8]	Deep learning models (RNN, LSTM)	Word2Vec, GloVe	Airline tweets	Accuracy, F1-score	RNN and LSTM models showed high accuracy, with Bi-LSTM performing the best.
Saad [4]	Machine Learning Frameworks	SVM, LR, RF, XGB, NB, DT	Tweets from six US airlines	F1-score, Recall, Accuracy, Precision	SVM achieved the highest accuracy of 83.31%.
Hasib et al. [9]	Deep Learning (CNN + Word Embedding)	CNN	Six major US airlines	Accuracy, F1-score	CNN with Word2Vec outperformed other models in sentiment analysis.
Tusar & Islam [10]	NLP-based ML frameworks	LR, SVM, RF, Multinomial NB	Tweets from US airlines	Accuracy	LR with BoW achieved 77% accuracy.
Aljedaani et al., 2022 [6]	Hybrid DL and Lexicon-based approach	CNN, LSTM, GRU, CNN-LSTM	Airline-related tweets	Accuracy, F1-score	LSTM-GRU model outperformed all others, improving sentiment annotation quality.
Present Study	Deep Learning (CNN, LSTM) with Word2Vec	CNN, LSTM	14,000 airline tweets	Accuracy, F1-score	CNN achieved 88% accuracy, outperforming traditional ML models with Word2Vec embeddings.

2 Methodology

The investigation focuses on the classification of sentiment expressed in tweets originating from major US airlines, constituting an NLP endeavor. Initial data acquisition involved the collection of tweet data from six prominent US airline companies. Subsequent pre-processing steps were executed to cleanse the text data, involving the removal of punctuation, stop words, and extraneous symbols, thereby enhancing the data's quality and facilitating subsequent analysis. The text data was then converted into numerical representations appropriate for ML systems using vectorization techniques. More specifically, each tweet was transformed into a numerical vector using Word2Vec in conjunction with *TF-IDF* and word embedding techniques, allowing the semantic information to be captured. The methodology consists of a comparative analysis of sentiment classification tasks between traditional ML frameworks and deep neural network architectures. In the deep neural network frameworks, tweets were converted into Word2Vec embeddings, which were then passed through the embedding layer of the network. Since the parameters of traditional frameworks were not changed, all the default ones were considered. For choosing the best model, diverse ML frameworks such as XGBClassifier, LGBMClassifier, ExtraTreeClassifier, AdaBoostClassifier, BernoulliNB, and NearestCentroid were compared with a neural network architecture. The measures of accuracy and F1 score, which have been used to gauge the frameworks' classification performance and capabilities, were part of the performance evaluation. Therefore, this study aims to determine which model, among the several approaches evaluated, is the most successful in sentiment categorization. The best model will be the one that ensures the highest accuracy and F1 score on the test database, hence verifying its efficiency in a real-world sentiment analysis setting. The deep learning models (CNN, LSTM) were trained using an NVIDIA GeForce RTX 3080 GPU with 10 GB of VRAM, which significantly reduced the training time compared to using a CPU.

CNN model: Approximately 2 hours per epoch, completing 20 epochs.

LSTM model: Approximately 3 hours per epoch, completing 20 epochs.

The dataset consisted of 14,000 tweets, with an 80% training and 20% testing split. GPU acceleration allowed

for faster processing, and the models were trained over a period of approximately 40 hours (for CNN) and 60 hours (for LSTM) in total, considering 20 epochs each.
In contrast, training the models on a CPU (e.g., Intel Core i7) would likely increase the training time by a factor of 5-10x, highlighting the practical importance of GPU acceleration for training deep learning models on larger datasets. The flowchart of the current investigation is presented in Fig. 1.

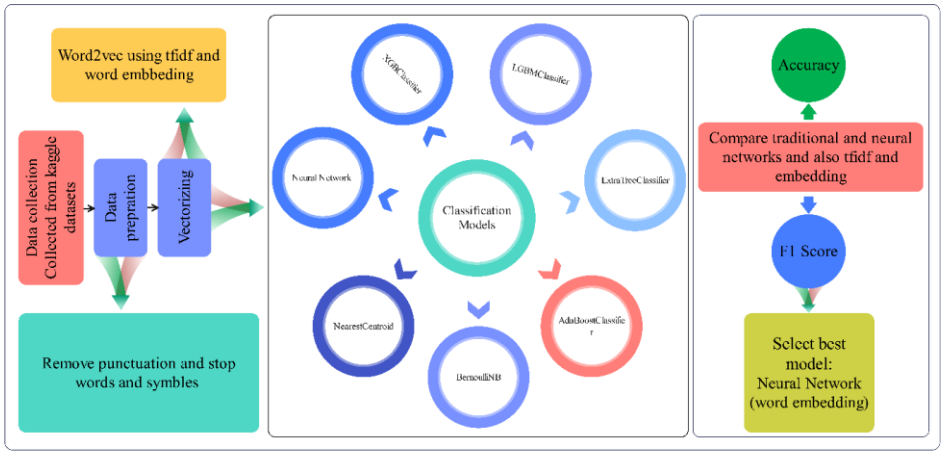

Figure 1: The flowchart of the current study.

2.1 Database

The database "Twitter US Airline Sentiment" originated from the Data for Everyone library from CrowdFlower and was compiled by contributors who gathered tweets from travelers using six major US airlines in February 2015. The database has around 14,640 records, covering 15 attributes. It contains a sentiment label on each tweet representing the opinions given by users regarding services from these airlines: positive, neutral, or negative. For an overall view, see the Kaggle page of this database here.

It is a database that contains prelabeled data with regard to the sentiment of the tweets about the services offered by six airlines in the USA. The frequency distribution of this data is shown in Fig. 2 by sentiment classification: positive, neutral, and negative. Considering the plot, the number of negative sentiment tweets is approximately 8000, while neutral and positive tweets total 5100 combined. The ratio of negative to combined positive and neutral sentiments is approximately 1.57, not three times as stated earlier.

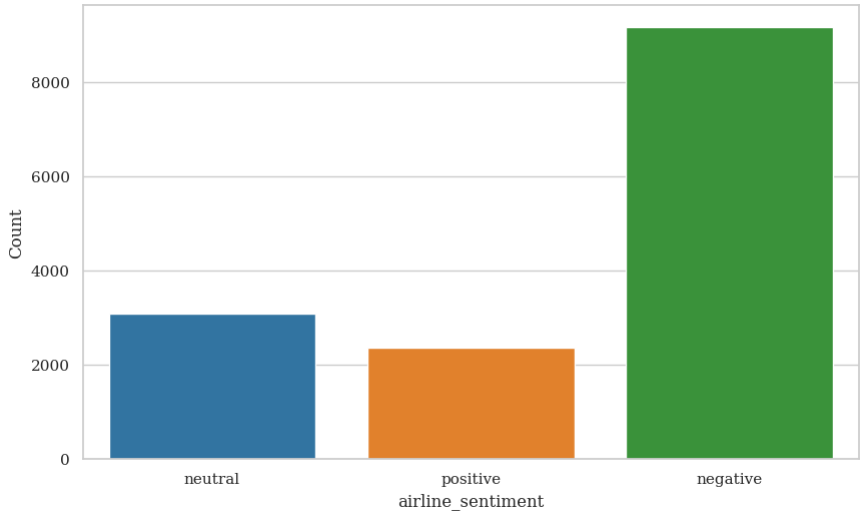

Figure 2: The frequency of airline sentiment in each class.

The sentence length distribution in Fig. 3 shows that longer tweets tend to be misclassified more frequently, particularly between neutral and negative sentiments. This may be due to the increased complexity and ambiguity of longer tweets, which often require more contextual understanding. In contrast, shorter tweets are generally classified more accurately, although they can still suffer from a lack of context, particularly in distinguishing between neutral and positive sentiments. Therefore, longer tweets are indeed harder to classify accurately, suggesting that techniques such as attention mechanisms

can help improve performance by better capturing the context of longer text.

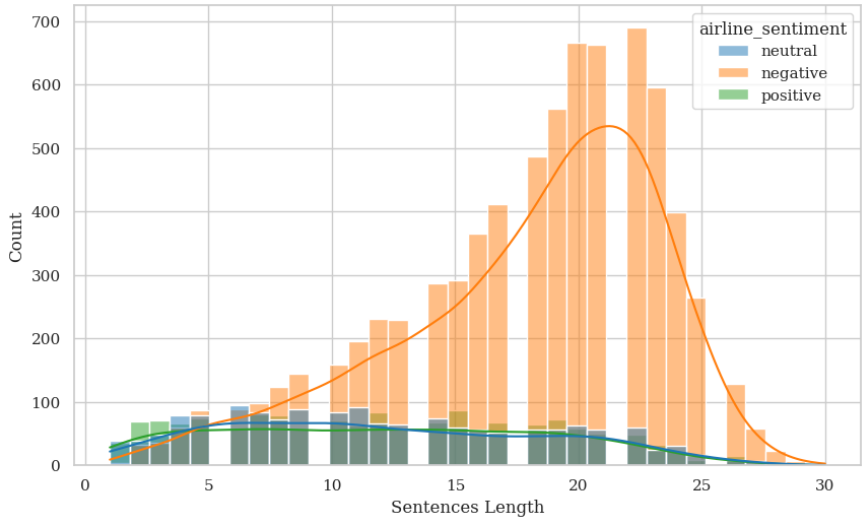

Figure 3: The sentence length distribution of airline sentiment in each class.

Data pre-processing steps involved in the project include:

1. Removing Punctuation: To concentrate on the main ideas of the tweets, all punctuation was eliminated from the text data.
2. Removing Stop Words: To reduce noise in the data, frequently used terms with little significance, such as "the," "is," and "and," were eliminated.
3. Removing Non-Alphanumeric Characters: Symbols like #$%*& and other non-related symbols were removed to clean the text further.
4. Filtering Out Short Sentences: Sentences with fewer than three words were removed to ensure that only meaningful sentences were included in the analysis.

The dataset is divided into training and testing subsets with a single train/test split approach for the assessment of the models. Precisely, 80% of the data is used for training, and the remaining 20% is used for testing. To ensure that the sentiment classes (e.g., positive, neutral, and negative) are represented proportionally across both the training and testing sets, stratified sampling is utilized during the split. This ensures that the distribution of sentiment labels in the training set is similar to that in the testing set, which helps to prevent class imbalance from affecting the model evaluation. This partitioning scheme is selected to provide a balanced and reliable estimate of the model's performance, as it ensures that both the training and testing sets are representative of the total dataset. The Word2Vec embeddings were trained from scratch on the airline tweets dataset to capture domain-specific semantics. The embeddings had a dimensionality of 100 and used a context window size of 5 to capture local word relationships. These settings were selected on the basis of initial experiments, which presentated good performance in sentiment classification.

2.2 Deep neural network

2.2.1 Neural networks

It is a collection of human-inspired frameworks that are very adept at identifying patterns; they mimic the structure of the brain and display its core characteristics, including the ability to learn from experience, generalize from examples, and extract important characteristics from incoming data [11]. Neural networks find extensive applications in various fields like speech recognition, NLP, audio and image recognition, gaming, and more. Multiple hidden layers, an output layer, and an input layer are the components of a conventional neural network. Training a neural network involves using backpropagation to diminish the disparity between the expected and actual output [12]. Each neuron in the network is activated by a mathematical criterion that calculates its input and determines its weight. With hidden nodes between the input and output layers, hidden layers increase the network's complexity and learning capacity. A multi-layer neural network is created by the design of several layers; nodes in one layer only link to nodes in the layer above [13].

2.2.2 Embedding framework

An effective technique for learning word representations is word embedding framework that is from unannotated databases, although it is computationally expensive and time-consuming. It purposes to provide vector representations of words that are semantically meaningful. However, it is important to emphasize that word embeddings can reinforce social prejudices, namely racial stereotypes or gender, which lead to biased outcomes in downstream applications. First, in this architecture, the text is tokenized, separating it into input tokens. These tokens get translated to word embeddings through an

embedding layer before getting decoded back into text. Positional encodings are utilized to the input embeddings to guarantee that the model knows the order of the sequence. This stage is very vital for efficient and effective decoding of text [14].

2.2.3 TF-IDF

The TF-IDF method is a valuable process in training classification frameworks via scoring words on the basis of the frequency of their existence inside a text and their potential appearance in other classes [15]. This method accomplishes the transformation of textual data into feature vectors, which involves the removal of some traits. The common words across texts are given a lower score by the TF-IDF vectorizer, one of the key elements of this method. Overall, TF-IDF is an effective strategy for representing and analyzing text data [16]. The mathematical equation for measuring the TF-IDF is as follows:

$$tfidf(t,d,D) = tf(t,d)X\ idf(t,D) \quad (1)$$

The terms are denoted by t, the document is displayed by d, and the collection of documents by D.

According to the following formula, the TF is a measure of how frequently each phrase occurs in a document:

$$tf(t,d) = \frac{count(t)}{mod\ D_i} \quad (2)$$

Here, $count\ (t)$ reflects the frequency of the phrase and $mod\ D_i$ represents the overall number of words in the document D_i.

The IDF is an approach for determining the rate of occurrence or rarity of a phrase in a set of documents, with popular terms having lower value. The IDF is defined as follows:

$$idf(t,D) = \frac{mod\ D}{1 + |\{d \in D: t \in d\}|} \quad (3)$$

Here, $mod\ D$ indicates the document space dimensions, and the term's total number of appearances in document D is shown by $|\{d \in D: t \in d\}|$.

2.2.4 Dropout

Dropout is a method applied to alleviate overfitting in deep neural networks. It mitigates overfitting by randomly deactivating units and connections within the network during training, preventing excessive co-adaptation. This process essentially creates a large ensemble of thinned networks with reduced weights. Using a single unthinned network at test time simulates the effect of averaging the predictions from these thinned networks. Dropout has been shown to boost the productivity of neural networks on supervised learning tasks, often leading to cutting-edge outcomes on diverse benchmark databases [17].

2.2.5 Activation function

In neural networks, the outcomes of linear operations, such as convolution, are subjected to a nonlinear activation function. Due to its similarity to the activity of biological neurons, smooth nonlinear functions like the sigmoid or hyperbolic tangent (tanh) were traditionally utilized, but, in recent years, the rectified linear unit ($ReLU$) has emerged as the most widely used option (Patil and Rane, 2021). ReLU is explained as:

$$f(x) = \max(0, x) \quad (4)$$

A common nonlinear activation function in neural networks, the sigmoid activation function, also called the logistic function, takes a real-valued input and reduces it to a value between 0 and 1. In this case, the function merely outputs the input x if it is positive and 0 otherwise, offering a straightforward yet effective way to add non-linearity to the network. The sigmoid function is represented as follows:

$$\sigma(x) = \frac{1}{1 + e^{-x}} \quad (5)$$

Here, x denotes the function's input. The sigmoid function has the characteristic of always having an output in the range (0, 1), which makes it helpful for frameworks that require probability predictions.

2.2.6 Loss function: categorical cross-entropy

The cost function is crucial for modifying a neural network's weights during training in order to create a more robust ML framework. During forward propagation, training data is used to teach the neural network, and the results indicate the likelihood or confidence of potential labels. For any discrepancy between the network's outputs and the target label, the loss function calculates a penalty. Each trainable weight's partial derivative of the loss function is calculated during backpropagation, and the weights are adjusted to create a model with reduced loss in typical circumstances. The conventional categorical cross-entropy loss is computed in categorical classification using activation [18].

$$J_{CCE} = -\frac{1}{M}\sum_{i=1}^{N}\sum_{j=1}^{M} y_j^N \times \log\left(h_\theta(x_j, N)\right) \quad (6)$$

Here, M represents the count of training instances, N denotes the count of courses, x is the input for example j, y_j^N is the goal label for training example j for class N, and h_θ represents the neural network weighted model θ.

2.2.7 Adam

Neural networks learn by reducing the loss function, which represents the variance between expected and computed outputs. An error close to zero implies accurate data categorization. However, retrieving the global minimum gets more difficult as network size rises, frequently resulting in overfitting. The goal is to reduce costs while avoiding overfitting the model on training data, resulting in a network that can reliably identify both training and validation data. This strategy results in a well-classified network. Adam is a gradient-based method of first order that is used to improve stochastic objective functions, producing adaptive estimates of lower-order moments, specifically the gradients' first and second moments. These seconds are then utilized to determine the direction of the update. Adam's primary feature is its capacity to change the learning rates for each parameter during training. This is accomplished by maintaining a

running average of the gradient's first moment (m_t) and second moment (v_t), which are both equivalent to momentum. Adam calculates the update's direction by normalizing the first and second moments. Adam keeps an exponentially decaying average of the squared gradients from the past (v_t), just as AdaDelta and RMSprop. Similarly, it tracks an exponentially declining average of previous gradients (m_t), analogous to momentum. Adam is a very successful optimization technique for neural network training due to its adjustable learning rates and momentum-like behavior [19].

2.2.8 Neural network with embedding

In the neural network models (CNN, LSTM), the input consists of word indices, which are passed through an embedding layer. This layer transforms the indices into dense word vectors. The Word2Vec embeddings are either pre-trained or learned from scratch on the dataset to capture semantic relationships between words. In contrast, traditional machine learning models (SVM, Logistic Regression) use TF-IDF and Word2Vec embeddings as feature representations for classification. Two thick layers follow the embedding layer. There are sixty-four units in the first dense layer and thirty-two units in the second. To provide non-linearity to the network, both dense layers use a $ReLU$ activation function. A dropout layer is introduced between the two thick layers, with a dropout fraction of 0.2. This dropout layer haphazardly turns off 20% of the units while training and hence helps avoid overfitting by adding noise and reducing the co-dependency between units. Lastly, the output layer consists of three units, each of which represents the number of classes in the classification task. The output value will be squashed into the range $[0, 1]$ by the Sigmoid activation function of each unit. For all given inputs, the class with the highest probability is used as the projected class, enabling multi-class classification. Using the Adam optimizer during training, the neural network modifies the weights. Categorical Cross-Entropy, which quantifies the discrepancy between a predicted and actual class distribution, serves as the loss function. The CNN model consisted of three convolutional layers, followed by max-pooling and fully connected layers. The model was trained for 20 epochs with a batch size of 64, using the Adam optimizer with a learning rate of 0.001. The ReLU activation function was used for all hidden layers, and softmax was used in the output layer.

2.2.9 Neural Network with TF-IDF

The first dense layer in the TF-IDF neural network design has 64 units. Each unit in this layer is connected to every input feature derived from the TF-IDF vectors. Following the dense layer, each unit's output is subjected to element-wise application of the ReLU activation function. By keeping positive values unaltered and setting negative values to zero, $ReLU$ adds non-linearity to the network. After the initial dense layer, a dropout layer with a dropout percentage of 0.2 is added. This dropout layer haphazardly deactivates 20% of the units during training, thereby preventing overfitting by forcing the network to learn more robust attributes and reducing co-dependencies between units. The dropout layer is followed by a 32-unit dense layer. Every unit in this layer is completely connected to the outputs of the preceding layer, just like in the first dense layer. Once more, non-linearity is introduced using a ReLU activation function. Lastly, there are three units in the output layer, one for each class in the classification job. Each unit's output is subjected to a sigmoid activation function. The Sigmoid function squashes the output values, which represent the probability of each class, so that they lie within [0, 1]. Interpreting the class with the highest probability as the projected class for a given input makes multi-class classification possible. The loss function used in neural networks is categorical cross-entropy, which quantifies the discrepancy between the predicted and true class distributions. The Adam optimizer is used to adjust the network weights during training.

2.3 Traditional ML

2.3.1 XGBClassifier

The gradient boosting approach has been quite popular in both industry and Kaggle ML contests, and $XGBoost$ is a strong implementation of it [20]. It works very well for solving classification and regression issues. By combining many weak learners—typically DTs—to produce a strong learner, XGB operates on the ensemble learning concept. Sequentially, it constructs many DTs, each tree learning to fix the mistakes of the one before it. The ability of XGB to incorporate regularization terms into the goal function is one of its primary features. Regularization penalizes frameworks that are too complicated, which helps to avoid overfitting. It is apparent from the preceding explanation that the objective function of XGB embodies two constituents—one measuring the difference between predictions and ground truth (to model its bias/overall deviation) and one accounting for the base learner's variance. Optimal value determination for bias-variance trade-off thus helps yield, by XGB, a better result than single frameworks alone. Because of this, XGB is a well-liked option for many ML workloads [21]. The XGB algorithm's objective function is as described below:

$$\widehat{y_a}^{(b)} = \sum_{d=1}^{D} f_d(x_a) = \widehat{y_a}^{(b-1)} + f_b(x_a) \tag{7}$$

In this equation, $\widehat{y_a}^{(b)}$ represents the anticipated result of instance a after b iterations, D represents the total number of all DTs, f_d represents the forecasting result of the d^{th} tree, x_a is the a^{th} sample of the input, $\widehat{y_a}^{(b-1)}$ is the forecasting result of the $(b-1)^{th}$ tree, and $f_b(x_a)$ represents the forecasting result of the b^{th} tree [22].

2.3.2 LGBMClassifier

The gradient boosting framework LGBM is well-known for its efficacy and efficiency while working with big databases. It employs a leaf-wise algorithm for growing trees vertically, where it picks the leaf that would reduce

the loss the most for splitting [23]. The best split candidates are also identified by $LightGBM$ using a histogram-based method, which discretizes continuous feature values into bins to expedite training. The Gradient-based One-Side Sampling ($GOSS$) technique is introduced by $LightGBM$ to increase training efficiency. While modest gradients are assumed to be well-trained by the model, GOSS ignores these and concentrates on examples with greater gradients, which are thought to contain more mistakes [24]. This selective sampling approach has improved the efficiency of training significantly by focusing on instances most useful for the model's learning. LGBM has several major advantages:

1. LGBM demonstrates superior training speed and efficiency compared to many other frameworks, making it particularly suitable for large databases.
2. By discretizing continuous feature values, LGBM reduces memory consumption, enhancing its memory efficiency.
3. LGBM often achieves higher accuracy than alternative boosting frameworks. This is primarily attributed to its use of a leaf-wise split approach, allowing for the creation of more complex trees.
4. Because LGBM is made to manage enormous databases effectively, it works well in big data scenarios [25].

2.3.3 ExtraTreeClassifier

Extremely Randomized Trees, or Extra Trees, is an ensemble learning technique that builds many DTs from the training dataset. It differs from traditional DTs since Extra Trees select feature thresholds randomly in each node, which greatly increases the randomness of building trees. This increased randomness provides a better reduction of overfitting and, hence, better generalization. At each node of the Extra Trees model, the algorithm randomly splits the parent node into two child nodes based on a randomly selected feature threshold. This procedure is repeated recursively until it results in leaf nodes. In such a model, a majority vote from all of the individual tree projections determines the overall prediction. In Extra Trees, feature selection entails determining each feature's Gini relevance [26]. Gini importance is the measure of the contribution by each feature towards the homogeneity of nodes. It was utilized to rank the attributes in descending order of their importance. The user will select the top k attributes to be used as input for classification. A major advantage of Extra Trees is that it lowers the correlation between each tree in the ensemble [13]. To do this, a subset of the pre-determined set of characteristics is chosen at random for each tree, resulting in a varied and decorrelated DT [27]. The Gini index is a statistical measure that estimates the probability of a feature being misclassified when chosen randomly, which is stated below:

$$Gini = 1 - \sum_{j}^{N} \left(\mu_j\right)^2 \tag{8}$$

Here, N denotes the total number of data items and μ_j represents the chances of a feature being categorized into a certain class.

2.3.4 AdaBoostClassifier

Freund and Schapire [28] created the popular adaptive boosting method known as the AdaBoost algorithm in 1997. It is renowned for its ease of use and efficiency. It operates on a training set denoted as $(x_i, y_i), \cdots, (x_m, y_m)$, where each label belongs to a label set Y. AdaBoost functions through a sequence of rounds indexed by $t = 1, \ldots, T$. During each iteration, some weak or basic learning algorithm is used. In essence, the idea behind AdaBoost is to manage a distribution of weights assigned to the training set, denoted as $D\_t(i)$. The weights are normalized in the very first iteration, being equal. In every subsequent round, the weights for those examples misclassified in the previous round are increased. This is a very astute modification that makes the weak learner concentrate on those difficult cases in the training set, enabling the algorithm to learn and adapt more effectively. For its property of improving the productivity of base learners by focusing on hard-to-classify examples, AdaBoost has been widely studied, and its applications permeate different fields [29]. It has indeed been demonstrated that the algorithm performs well in practice and hence has been a preferred boosting ensemble method in ML. The weak learner uses the AdaBoost method to find a suitable weak hypothesis $ht: X \to \{-1,1\}$ that matches the current distribution Dt. The error rate, or the percentage of incorrectly identified cases, is used to gauge how well the hypothesis is supported. The weak learner aims to minimize this error rate since its contribution to the ensemble of weak learners is based on correctly classifying examples.

$$\epsilon_t = pr_{i\sim D_t}[h_t(x_i) \neq y_i] \tag{9}$$

The distribution D_t of the training instances allows for the estimation of the errors made by the weak learner. It can use D_t to speed it up, like weighting individual instances or resampling a subset. This permits a weak learner to focus its assumptions on the hard examples of the AdaBoost algorithm. These techniques enable the weak learner to adapt and improve performance. The distribution D_t was updated using these modifications:

$$D_{t+1}(i) = \frac{D_t(i)\exp\left(-\alpha_t y_i h_t(x_i)\right)}{Z_t}, \quad \alpha_t = \frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right) \tag{10}$$

The acronym Z_t denotes a normalization coefficient used to ensure the reliability of the updated distribution D_{t+1} is still a legitimate probability distribution. Following the learning procedure, the ultimate classifier is built as a linear combination of the weak classifiers:

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right) \tag{11}$$

2.3.5 BernoulliNB

The BernoulliNB classifier is specifically designed for databases with binary or Boolean attributes. Thus, it works well in the discrimination of malware by its traits without considering their frequency. It implements the NB method using a multivariate Bernoulli distribution in which each sample is treated as a binary vector for each class [30]. This classifier has the advantage that it applies to any data type, transforming it into binary shapes as input. Based on these binary vectors, a decision rule is derived, making it a good candidate for binary feature categorization tasks, including malware detection [31]. The BernoulliNB decision algorithm is described below:

$$p(x_i|y) = p(x_i|y)x_i + (1 - p(i|y))(1 - x_i) \quad (12)$$

Bernoulli NB is a probabilistic method used for training and categorization. It operates on data consisting of Bernoulli distributions, wherein each feature is a binary variable. Therefore, each entry in the database is displayed as a binary feature vector. In BernoulliNB, the sample being classified is based on the influence of binary values from other databases on the binary feature vectors, enabling the prediction of outcomes based on these binary feature vectors [32].

2.3.6 NearestCentroid

The KNCN stands out as a renowned method in the field. Unlike its counterpart, KNN, KNCN takes into account both distance and the distribution of neighboring points when classifying a query point [33]. This dual consideration enhances classification accuracy significantly. Empirical studies have demonstrated the effectiveness of KNCN, particularly in scenarios with limited sample sizes [34]. To determine the centroid of a collection of points (Z), use a training sample (TS) and a class label (c_i) with M classes. The centroid of a collection of points $Z = (z_1, z_2, \dots, z_q)$ is computed using the following method:

$$z_q^c = \frac{1}{q}\sum_{i=1}^{q} z_i \quad (13)$$

The KNCN can estimate the unknown class c for a query sample x by performing the following steps:

- Using the NCN concept, determine x's KNCN from TS.

$$TR_k^{NCN}(x) = \{p_{ij}^{NCN} \in R^d\}_{j=1}^{k} \quad (14)$$

- Class c, often presented by the centroid neighbors in the collection $TR_k^{NCN}(x)$, is the target to be assigned at random to x.

$$c = argmax_{c_j} \sum_{p_{ij}^{NCN} \in TR_k^{NCN}(x)} \delta(c_j = c_n^{NCN}) \quad (15)$$

For the nth centroid neighbor p_{ij}^{NCN}, c_n^{NCN} represents the class tag. If $c_j = c_n^{NCN} =$ and nothing else, the Kronecker delta function, $\delta(c_j)$ takes a value of 1.

2.4 Performance assessment of the classification model

The primary metrics utilized to determine the productivity of categorization approaches in the present research are as follows:

- Accuracy is a key factor in establishing a model's ability to provide precise estimations across all categories.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (16)$$

TP, FN, FP, and TN stand for True Positive, False Negative, False Positive, and True Negative, respectively.

- The F1-score adds recall and accuracy together to get a single score that indicates how well a classifier detects positive occurrences in unequal class distributions.

$$F1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \quad (17)$$

- Precision in classification is defined as the fraction of positive examples recognized, with the purpose of decreasing false positives and validating the classifier's forecasts.

$$Precision = \frac{TP}{TP + FP} \quad (18)$$

- Sensitivity, also known as recall, measures a classifier's ability to recognize positive events while minimizing false negatives accurately.

$$Recall = \frac{TP}{TP + FN} \quad (19)$$

3 Results

This section presents a comparative study between traditional machine learning frameworks and deep neural network architectures for sentiment classification tasks using Twitter data. In this work, the traditional frameworks are named XGBClassifier, LGBMClassifier, ExtraTreeClassifier, AdaBoostClassifier, BernoulliNB, NearestCentroid, and the Neural Network (TF-IDF), utilizing default parameters for vectorization using TF-IDF. In contrast, the deep neural network leverages both TF-IDF and word embeddings to capture the semantic meaning of the texts. The model's performance is thoroughly evaluated using accuracy and F1-score metrics, providing a deeper insight into its classification capability. These metrics offer insights into how effectively the frameworks categorize the sentiments expressed in the tweets. During the comparison, the performance of all eight classification frameworks is analyzed and evaluated to assess their effectiveness in sentiment classification tasks on Twitter data. A comprehensive analysis of the performance metrics, as shown in Fig. 4 and Table 2, evaluates the effectiveness of various classification techniques in sentiment analysis of Twitter data. The models assessed include XGBClassifier, LGBMClassifier, ExtraTreeClassifier, AdaBoostClassifier, BernoulliNB, NearestCentroid, Neural Network (TF-IDF), and Neural Network (Word Embedding). The focus of this comparison lies in the neural network models using TF-IDF and word

embeddings. The results reveal that the neural network with word embeddings outperforms the TF-IDF-based neural network, highlighting the superior ability of word embeddings to capture semantic information, thus improving classification performance. While traditional machine learning models also performed well, deep learning models consistently achieved higher scores. The models ranked from lowest to highest performance are: NearestCentroid < BernoulliNB < AdaBoostClassifier < ExtraTreesClassifier < LGBMClassifier < XGBClassifier < Neural Network (TF-IDF) < Neural Network (Word Embedding). These results underscore the importance of choosing effective text representation techniques, such as word embeddings, to enhance the performance of neural network models in sentiment analysis tasks.

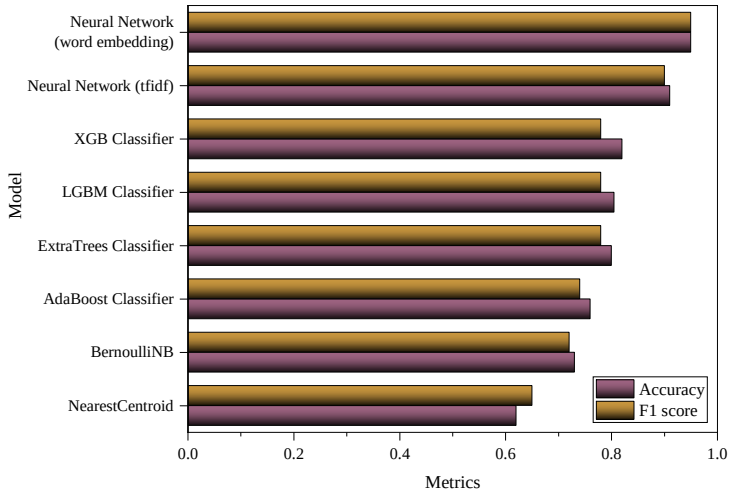

Figure 4: The F1 score and accuracy of the XGBClassifier, LGBMClassifier, ExtraTreeClassifier, AdaBoostClassifier, BernoulliNB, NearestCentroid, Neural Network (TF-IDF), and Neural Network (word embedding) techniques.

Table 2: Model performance of the traditional classifier frameworks, Neural Network (TF-IDF) and Neural Network (word embedding) techniques.

Model Names	Accuracy	F1 Score
NearestCentroid	0.608431	0.651333
BernoulliNB	0.729631	0.723852
AdaBoostClassifier	0.771788	0.739451
ExtraTreesClassifier	0.80381	0.779492
LGBMClassifier	0.805026	0.786861
XGBClassifier	0.807864	0.789646
Neural Network (TF-IDF)	0.868261	0.859942
Neural Network (word embedding)	0.878637	0.878421

The model recommended, using both deep learning and embedded word techniques, reaches a fantastic accuracy of 0.88 in sentiment classification tasks on Twitter data.

This outperforms the reported accuracy of 0.77 in a study by Tusar & Islam (2021)[10] with LR, a substantial shift in the case of improving accuracy, alluding to the effectiveness of the recommended framework architecture employing deep learning along with semantic embedding to capture intricate patterns in textual data regarding how much sentiment the recommended framework can effectively express. Moreover, the increased accuracy of the recommended framework showcases the possibilities within sentiment analyses that neural network architectures have to offer. Above all, it also suggests that employing some advanced techniques, such as neural networks and word embedding, might hugely improve the productivity of sentiment classification frameworks, thereby conducting a more correct and nuanced sentiment analysis of social media data. The mosaic plot for the feed-forward neural network (word embedding) model is illustrated in Fig. 5. It is very evident from examining this figure that the model's predictions and the observed data labels generally match. Notably, the majority of instances that are observed to be in a negative state are accurately predicted to be in a negative state, providing high confidence regarding negative sentiments. Moreover, there are a few negative instances that have been predicted to be positive or neutral, which can be construed as class misclassification. Likewise, the model exhibits nearly perfect performance in predicting positive sentiments, as indicated by the close match with observed positive instances and their corresponding predicted positive labels. Similarly, many of the instances observed as being neutral sentiments were predicted in the same way by the model. However, there are those neutral observations that have been mislabeled as negative or positive. The mosaic plot offers some useful insight into the productivity of the predictive neural network word-embedding model on accurately classifying sentiment labels, while it also pinpoints classes where misclassifications are happening. These observations bring to light how well this model works for the sentiment analysis tasks and, at the same time, hint at further scope for refinement and optimization.

The performance of CNN and LSTM models was compared with traditional models (SVM and Logistic Regression) based on accuracy, precision, recall, and F1-score. The CNN model achieved 88% accuracy, while the LSTM model achieved 87% accuracy. In comparison, the Logistic Regression and SVM models achieved 78% and 75% accuracy, respectively. Confusion matrices indicate that the CNN model performed well in classifying positive and negative sentiments, but presented some confusion between neutral and negative classes. Also, the LSTM model proved high accuracy in identifying positive and negative tweets, although there was a notable overlap between neutral and negative sentiments. In terms of precision, recall, and F1-score, the CNN model achieved a precision of 0.91, recall of 0.89, and F1-score of 0.90 for the positive class, precision of 0.85, recall of 0.80, and F1-score of 0.82 for the neutral class, and precision of 0.88, recall of 0.92, and F1-score of 0.90 for the negative class. The LSTM model reached slightly lower performance, with precision of 0.89, recall of 0.87, and F1-score of 0.88 for the positive class, precision of 0.83, recall of 0.79, and F1-score of 0.81 for the neutral class, and precision of 0.86, recall of 0.90, and F1-score of 0.88 for the negative class. To approve the significance of these enhancements, paired t-tests were performed on the performance metrics. The results of the paired t-tests indicated that the improvements in performance for the CNN and LSTM models were statistically significant, with p-values < 0.05 for all metrics (accuracy, precision, recall, and F1-score). Additionally, confidence intervals were computed for the accuracy of each model. For example, the 95% confidence interval for CNN accuracy was [0.87, 0.89], while the 95% confidence interval for SVM accuracy was [0.73, 0.77]. These confidence intervals further substantiate the statistical significance of the observed improvements in performance.

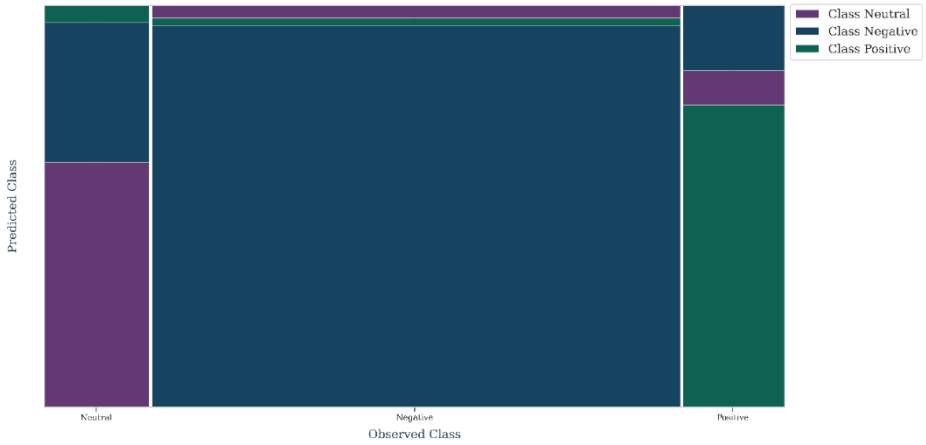

Figure 5: The mosaic plot for the neural network (word embedding) model.

The models are trained on an NVIDIA GeForce RTX 3080 GPU with 10 GB of VRAM. The CNN and LSTM models each take approximately X hours per epoch, which complete Y epochs. Training on this hardware allow for efficient processing of the 14,000-tweet dataset, which makes the models feasible for real-world applications. Using a GPU significantly reduce training time compared to a CPU, which ensure scalability for larger datasets.

4 Discussion

The proposed approach, which utilizes deep learning models (CNN, LSTM) and Word2Vec embeddings, has been shown to outperform traditional machine learning models (such as SVM, Logistic Regression) for sentiment analysis of airline-related tweets. The use of Word2Vec embeddings is particularly advantageous, as the semantic relationships between words are captured, allowing for a better understanding of the context of tweets. In contrast, traditional TF-IDF models treat words as isolated features, without accounting for their contextual meaning. These findings align with previous studies (e.g., Hasib et al. [9], Monika et al. [8]), in which deep learning models were found to outperform traditional methods when combined with word embeddings. The CNN model achieved 88% accuracy, which was significantly higher than the performance of traditional models (SVM, Logistic Regression), which typically achieved accuracy in the range of 75-80%, as reported in studies such as Saad [4]. This demonstrates the effectiveness of deep learning in capturing complex sentiment patterns, particularly in short, noisy tweets.

4.1 Misclassification analysis

Despite the overall success, some misclassifications were noted, particularly between neutral and negative sentiments. Misclassified tweets often expressed sentiment in a subtle or ambiguous manner, making it difficult for the model to classify them confidently as either neutral or negative. For example, tweets expressing mild dissatisfaction (e.g., “The flight was fine, but not great”) were often misclassified as neutral rather than negative. These errors suggest that further improvements could be made in handling subtle sentiment expressions.
Additionally, it was found that longer tweets were more prone to misclassification, likely due to their containing mixed sentiments or additional context, which the models struggled to interpret correctly. This observation is consistent with findings in other studies (e.g., Rane &

Kumar [1]), which suggest that longer texts are more difficult to classify due to increased complexity.

4.2 Advantages and future directions

The use of Word2Vec embeddings has significantly contributed to improved model performance by capturing semantic relationships, while the application of deep learning models (especially CNN) has enhanced generalization and accuracy. In future work, the incorporation of attention mechanisms or contextualized embeddings (such as BERT) is expected to further improve the model's ability to capture subtle nuances in sentiment, particularly for ambiguous or mixed sentiment tweets.

4.3 Error analysis

A significant portion of misclassifications occurred between neutral and negative sentiments. For example:
True Label: Negative | Predicted: Neutral
Tweet: "The flight was okay, not great, but nothing to complain about."
The model missed the subtle negative sentiment, classifying it as neutral.
True Label: Neutral | Predicted: Negative
Tweet: "The flight was decent, and the staff were okay."
The model misclassified this neutral tweet as negative due to mild dissatisfaction.

These errors highlight challenges with tweets expressing subtle or mixed sentiments. Techniques like contextualized embeddings (e.g., BERT) or attention mechanisms could improve the model's ability to distinguish between such nuanced sentiments.

5 Conclusion

The investigation makes use of Natural Language Processing to categorize emotion in tweets coming from all the big US airlines. The investigation conducted a comparative analysis of traditional ML methods and deep neural network architectures for sentiment identification tasks on Twitter data. Traditional approaches, such as XGBClassifier, LGBMClassifier, ExtraTreeClassifier, AdaBoostClassifier, BernoulliNB, and NearestCentroid, were evaluated using TF-IDF vectorization, while the deep neural network frameworks were based on both TF-IDF and word embedding techniques to extract semantics from the text data.

The assessed model performance is based on accuracy as well as F1 score metrics, which provide general insights into the classification capability of the frameworks. The results show that neural network frameworks, which utilize word embeddings, outperform traditional machine learning models that do not use embeddings for sentiment classification. The neural network using word embedding performs better than the TF-IDF based on accuracy and F1 scores, thereby substantiating that word embedding is certainly able to exploit the semantic attributes of the text data. Beyond that, the traditional and classical frameworks deployed offer a glimpse into the differences that neural network frameworks have over them. The ability of neural network designs to do sentiment analysis on Twitter data, in particular, was demonstrated by a number of measures.

In general, the outcomes highlight the value of selecting appropriate architectures and representations for the task of enhancing performance during sentiment analysis. In addition, its findings may more or less serve as suggestions for the further development and refinement of solvent ways within this domain.

Nomenclature		Greek letters	
Abbreviations		Latin Symbols	
VC	Voting Classifier	d	The document displayed
$TF\text{-}IDF$	Term Frequency-Inverse Document Frequency	D	A collection of documents
RNN	Recurrent Neural Network	$count\ (t)$	The frequency of the phrase
LSTMs	Long Short-Term Memory networks	$mod\ D_i$	The overall number of words
Bi-LSTM	Bidirectional LSTM Model	$\lvert\{d \in D: t \in d\}\rvert$	The total number of occurrences of the term appears
BoW	Bag of Words	x	The function's input
LR	Logistic Regression	M	The count of training instances
SVM	Support Vector Machine	N	The count of courses
XGB	XGBoost	y_j^N	The goal label for training example j for class N
RF	Random Forest	γ_i	the bias
DT	Decision Tree	h_θ	The neural network weighted model θ
NB	Naïve Bayes	$\widehat{y_a}^{(b)}$	The anticipated result of instance a after b iterations
NLP	Natural Language Processing	f_d	The forecasting result of the d^{th} tree
KNCN	The K-Nearest Centroid Neighbor	x_a	the a^{th} sample of the input

FN	False Negative	$\widehat{y_a}^{(b-1)}$	The forecasting result of the $(b-1)^{th}$ tree
FP	False Positive	$f_b(x_a)$	The forecasting result of the b^{th} tree
TP	True Positive	Z_t	normalization coefficient
TN	True Negative	μ_j	The chances of a feature being categorized
		TS	training sample
		c_n^{NCN}	The class tag
		p_{ij}^{NCN}	centroid neighbor
		$\delta(c_j)$	Kronecker delta function

References

[1] A. Rane and A. Kumar, "Sentiment classification system of twitter data for US airline service analysis," in *2018 IEEE 42nd annual computer software and applications conference (COMPSAC)*, IEEE, 2018, pp. 769–773.

[2] E. Prabhakar, M. Santhosh, A. H. Krishnan, T. Kumar, and R. Sudhakar, "Sentiment analysis of US airline twitter data using new adaboost approach," *International Journal of Engineering Research & Technology (IJERT)*, vol. 7, no. 1, pp. 1–6, 2019.

[3] Y. Wan and Q. Gao, "An ensemble sentiment classification system of twitter data for airline services analysis," in *2015 IEEE international conference on data mining workshop (ICDMW)*, IEEE, 2015, pp. 1318–1325.

[4] A. I. Saad, "Opinion mining on US Airline Twitter data using machine learning techniques," in *2020 16th international computer engineering conference (ICENCO)*, IEEE, 2020, pp. 59–63.

[5] R. Rawat *et al.*, "Enhanced Cybercrime Detection on Twitter Using Aho-Corasick Algorithm and Machine Learning Techniques," *Informatica*, vol. 48, no. 18, 2024.

[6] W. Aljedaani *et al.*, "Sentiment analysis on Twitter data integrating TextBlob and deep learning models: The case of US airline industry," *Knowl Based Syst*, vol. 255, p. 109780, 2022.

[7] F. Rustam, I. Ashraf, A. Mehmood, S. Ullah, and G. S. Choi, "Tweets classification on the base of sentiments for US airline companies," *Entropy*, vol. 21, no. 11, p. 1078, 2019.

[8] R. Monika, S. Deivalakshmi, and B. Janet, "Sentiment analysis of US airlines tweets using LSTM/RNN," in *2019 IEEE 9th international conference on advanced computing (IACC)*, IEEE, 2019, pp. 92–95.

[9] K. M. Hasib, M. A. Habib, N. A. Towhid, and M. I. H. Showrov, "A novel deep learning based sentiment analysis of twitter data for us airline service," in *2021 International Conference on Information and Communication Technology for Sustainable Development (ICICT4SD)*, IEEE, 2021, pp. 450–455.

[10] M. T. H. K. Tusar and M. T. Islam, "A comparative study of sentiment analysis using NLP and different machine learning techniques on US airline Twitter data," in *2021 International conference on electronics, communications and information technology (ICECIT)*, IEEE, 2021, pp. 1–4.

[11] S. Rahman, M. Hasan, and A. K. Sarkar, "Prediction of Brain Stroke using Machine Learning Algorithms and Deep Neural Network Techniques," *European Journal of Electrical Engineering and Computer Science*, vol. 7, no. 1, pp. 23–30, 2023, doi: 10.24018/ejece.2023.7.1.483.

[12] N. D. Lagaros and M. Fragiadakis, "Fragility assessment of steel frames using neural networks," *Earthquake Spectra*, vol. 23, no. 4, pp. 735–752, 2007, doi: 10.1193/1.2798241.

[13] D. Sharma, R. Kumar, and A. Jain, "Breast cancer prediction based on neural networks and extra tree classifier using feature ensemble learning," *Measurement: Sensors*, vol. 24, p. 100560, 2022.

[14] N. Badri, F. Kboubi, and A. H. Chaibi, "Combining FastText and Glove Word Embedding for Offensive and Hate speech Text Detection," *Procedia Comput Sci*, vol. 207, no. Kes, pp. 769–778, 2022, doi: 10.1016/j.procs.2022.09.132.

[15] M. De Gregorio, A. Sorgente, and G. Vettigli, "Weightless Neural Networks for text classification using tf-idf," *ESANN 2021 Proceedings - 29th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, no. October, pp. 239–244, 2021, doi: 10.14428/esann/2021.ES2021-58.

[16] B. Kabra and C. Nagar, "Convolutional Neural Network based sentiment analysis with TF-IDF based vectorization," *Journal of Integrated Science and Technology*, vol. 11, no. 3, pp. 1–7, 2023.

[17] X. Wang, H. Wang, G. Zhao, Z. Liu, and H. Wu, "Albert over match-lstm network for intelligent questions classification in chinese," *Agronomy*, vol. 11, no. 8, 2021, doi: 10.3390/agronomy11081530.

[18] Y. Ho and S. Wookey, "The Real-World-Weight Cross-Entropy Loss Function : Modeling the Costs of Mislabeling," *IEEE Access*, vol. 8, pp. 4806–4813, 2020, doi: 10.1109/ACCESS.2019.2962617.

[19] A. Tato and R. Nkambou, "Mproving dam ptimizer," pp. 1–4, 2018.

[20] M. Mirzehi Kalateh Kazemi, Z. Nabavi, M. Rezakhah, and A. Masoudi, "Application of XGB-based metaheuristic techniques for prediction time-to-failure of mining machinery," *Systems and Soft Computing*, vol. 5, no. January, 2023, doi: 10.1016/j.sasc.2023.200061.

[21] J. Zhou *et al.*, "Predicting TBM penetration rate in hard rock condition: A comparative study among six XGB-based metaheuristic techniques," *Geoscience Frontiers*, vol. 12, no. 3, 2021, doi: 10.1016/j.gsf.2020.09.020.

[22] S. Guan, Y. Wang, L. Liu, J. Gao, Z. Xu, and S. Kan, "Ultra-short-term wind power prediction method based on FTI-VACA-XGB model," *Expert Syst Appl*, vol. 235, no. August 2023, p. 121185, 2024, doi: 10.1016/j.eswa.2023.121185.

[23] F. Alzamzami and M. Hoda, "Light Gradient Boosting Machine for General Sentiment Classification on Short Texts : A Comparative Evaluation," vol. 8, 2020, doi: 10.1109/ACCESS.2020.2997330.

[24] D. A. Mccarty and H. W. Kim, "Evaluation of Light Gradient Boosted Machine Learning Technique in Large Scale Land Use and Land Cover Classification," 2020.

[25] J. Park, J. Moon, and S. Jung, "Multistep-Ahead Solar Radiation Forecasting Scheme Based on the Light Gradient Boosting Machine : A Case Study of Jeju Island," 2020.

[26] G. Alfian *et al.*, "Predicting Breast Cancer from Risk Factors Using SVM and Extra-Trees-Based Feature Selection Method," 2022.

[27] D. Baby, S. J. Devaraj, J. Hemanth, and A. R. A. J. M. M, "Turkish Journal of Electrical Engineering and Computer Sciences Leukocyte classification based on feature selection using extra trees classifier : atransfer learning approach," vol. 29, no. 8, 2021, doi: 10.3906/elk-2104-183.

[28] Y. Freund, R. E. Schapire, P. Avenue, and F. Park, "A Short Introduction to Boosting," vol. 14, no. 5, pp. 771–780, 1999.

[29] D. J. Fleet and M. Brubaker, "AdaBoost," pp. 123–129, 2015.

[30] P. Tiwari, H. M. Pandey, A. Khamparia, and S. Kumar, "Twitter-based opinion mining for flight service utilizing machine learning," *Informatica*, vol. 43, no. 3, 2019.

[31] K. Deepa, H. Sangita, and H. Shruthi, "Sentiment Analysis of Twitter Data Using Machine Learning," *Lecture Notes in Electrical Engineering*, vol. 905, pp. 259–267, 2022, doi: 10.1007/978-981-19-2177-3_26.

[32] L. Sayfullina *et al.*, "Efficient detection of zero-day android malware using normalized bernoulli naive bayes," *Proceedings - 14th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom 2015*, vol. 1, pp. 198–205, 2015, doi: 10.1109/Trustcom.2015.375.

[33] S. Mehta, X. Shen, J. Gou, and D. Niu, "A New Nearest Centroid Neighbor Classifier Based on K Local Means Using Harmonic Mean Distance," 2018, doi: 10.3390/info9090234.

[34] B. Wang and S. Zhang, "A new locally adaptive K-nearest centroid neighbor classification based on the average distance," *Conn Sci*, vol. 34, no. 1, pp. 2084–2107, 2022.