Team-26: Pandemic Tweet Sentiment Prediction Report

Kazi Mahathir Rahman
Department of Computer Science, BRAC University
kazimahathir73.github.io
kazi.mahathir.rahman@g.bracu.ac.bd

1 Introduction

In recent years, the rise of social media platforms such as Twitter has provided an abundance of user-generated content that can be analyzed to extract meaningful insights. During the COVID-19 pandemic, Twitter has been a major source of public opinion, where users have expressed their sentiments regarding the pandemic. The ability to classify the sentiment behind these tweets can help in understanding public perception and guiding policy decisions.

This project addresses the task of sentiment classification of COVID-related tweets into one of five categories: *extremely positive*, *positive*, *neutral*, *negative*, or *extremely negative*. The dataset used for this task consists of over 41,157 labeled tweets, which provide the training foundation for developing models capable of predicting sentiments in unseen data.

In this task, one machine learning model, Random Forest Azhari et al. (2019), and three advanced deep learning models—Recurrent Neural Network (RNN) Feng et al. (2017), Long Short-Term Memory (LSTM) Hochreiter and Schmidhuber (1997), and Gated Recurrent Units (GRU) Dey and Salem (2017)—have been employed. These models are well-suited for handling the sequential nature of text. The results from Random Forest, RNN, LSTM, and GRU models are compared to identify the most effective model for sentiment classification in this specific context. The code for this implementation can be found in the GitHub repository. [1]

2 Data

2.1 Data Exploration and Visualization

The dataset provided for this project contains tweets related to the COVID-19 pandemic, along with their corresponding sentiment labels. The dataset has 41,157 instances in the training set, with the following columns: *Username*, *ScreenName*, *Location*, *Tweet At (Date)*, *Original Tweet*, and *Sentiment*. Without *Original Tweet* and *Sentiment*, all the columns were dropped. *Original Tweet* is my feature column and *Sentiment* is my label column.

The total number of instances for each sentiment class was plotted to understand the class imbalance in the dataset. As shown in Figure 1, the majority of the tweets are labeled as *neutral*, followed by *positive* and *negative* sentiments.

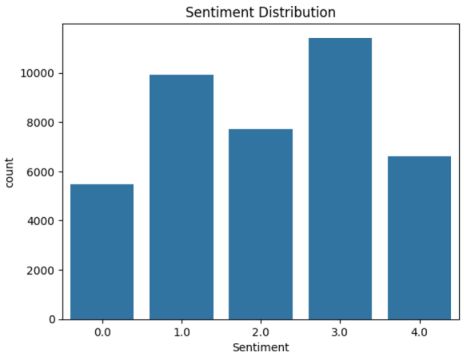

Figure 1: Distribution of sentiment classes in the dataset.

The top 10 most frequent words were identified for each sentiment class, providing insight into the words that commonly appear in tweets expressing various sentiments. In addition, WordCloud visualizations were created for each class to highlight the prominence of different words.

[1] https://github.com/kazimahathir73/CSE440-Natural-Language-Processing-II-Project

Figure 2: WordCloud representation of words in neutral classes.

The distribution of tweet lengths (in terms of the number of tokens) was plotted to analyze the variation in tweet lengths across the dataset. Most tweets tend to have a moderate number of words, with a small number of instances being significantly longer or shorter.

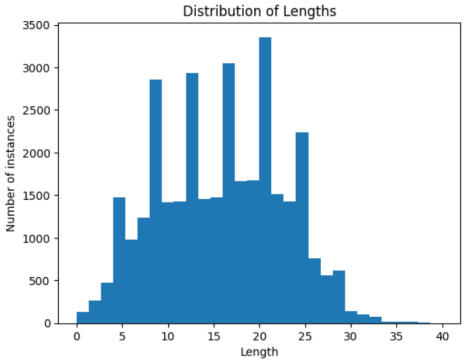

Figure 3: Distribution of tweet lengths (number of tokens) across the dataset.

2.2 Data Preprocessing

To prepare the data for model training, several preprocessing steps were undertaken:

1. **Handling Missing Values:** Tweets with missing values were identified and removed from the dataset.

2. **Data Cleaning:** Regular expressions (`re`) were used to clean the tweets by removing special characters, URLs, hashtag signs, and unnecessary punctuation and spaces.

3. **Stopword Removal:** Common stopwords (e.g., *the*, *and*, *is*) were removed to reduce noise in the text data.

4. **Tokenization:** The tweets were tokenized into individual words for further processing.

5. **Word2Vec Vectorization:** Pre-trained Model named googlenews-vectors-negative300 Word2Vec were used to convert the tokens into vector representations. Each tweet was represented by a vector of word embeddings. Each token is converted into 300 dim of the vector.

6. **Padding:** To ensure uniform input size for the LSTM and GRU models, padding was applied to all tweet vectors to match the maximum tweet length.

7. **Data Splitting:** The dataset was split into training, validation, and test sets for model evaluation. (80:10:10)

3 Model

In this section, we describe the machine learning model and the neural network architecture used for the classification task.

3.1 Random Forest Model

Random Forest is an ensemble learning method that constructs multiple decision trees during training and outputs the mode of the classes for classification tasks. This model is particularly effective for handling high-dimensional data and can capture complex interactions between features. The Random Forest algorithm operates by creating a subset of the training data for each tree, making it robust to noise and overfitting.

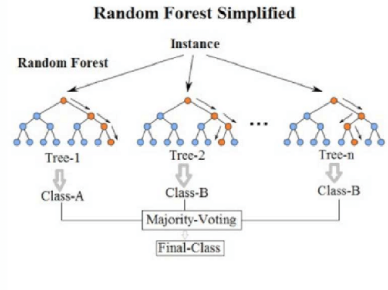

Figure 4: Architecture of the Random Forest model. (Azhari et al., 2019)

3.2 RNN Model

Recurrent Neural Networks (RNNs) are a class of neural networks designed to recognize patterns in sequences of data, such as time series or natural language. Unlike traditional feedforward neural networks, RNNs have connections that loop back on themselves, allowing them to maintain a memory of previous inputs in the sequence. This unique architecture makes RNNs particularly suitable for tasks that involve sequential dependencies.

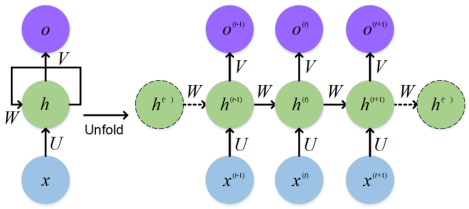

Figure 5: Architecture of the RNN model. (Feng et al., 2017)

3.3 LSTM Model

Long Short-Term Memory (LSTM) networks are a specialized form of recurrent neural networks (RNNs) that are particularly effective for sequence prediction problems. LSTMs are designed to overcome the limitations of traditional RNNs, particularly the vanishing gradient problem, by utilizing a memory cell that maintains information over long periods. This makes them suitable for tasks involving sequential data, such as natural language processing. The architecture of the LSTM model consists of a series of memory cells, each containing input, output, and forget gates that regulate the flow of information.

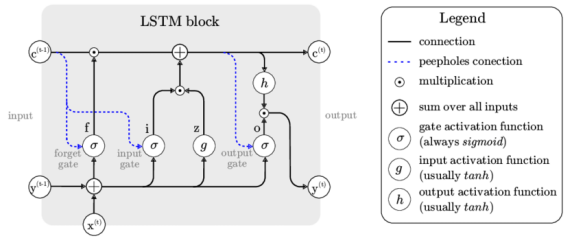

Figure 6: Architecture of the LSTM model. (Van Houdt et al., 2020)

3.4 GRU Model

Gated Recurrent Units (GRUs) are another variant of recurrent neural networks (RNNs) that were designed to address some of the shortcomings of traditional RNNs and even Long Short-Term Memory (LSTM) networks. GRUs simplify the architecture by combining the forget and input gates into a single update gate, which allows the network to efficiently manage the flow of information. This streamlined approach often leads to faster training times while maintaining comparable performance for many sequence prediction tasks. The GRU architecture consists of a series of gated units that control the information flow within the network.

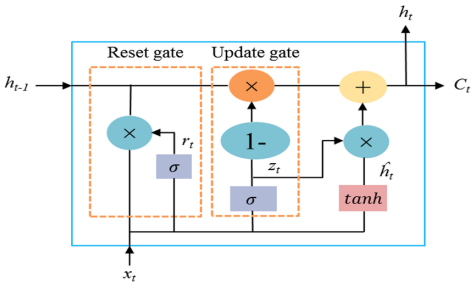

Figure 7: Architecture of the GRU model. (Mohsen, 2023)

	RF	RNN	LSTM	GRU
Input Size	300	300	300	300
Hidden Layers	N/A	1	1	1
Hidden Size	N/A	64	64	64
Optimizer	N/A	Adam	Adam	Adam
Learning Rate	N/A	0.0001	0.0001	0.0001
Epochs	N/A	40	30	25
Dropout	N/A	0.1	0.15	0.2

Table 1: Model Parameters for Random Forest, RNN, LSTM, and GRU

4 Results

In this section, we analyze and compare the performance of the models, including Random Forest, RNN, LSTM, and GRU. Each model was evaluated based on their training and validation losses, as well as their training, validation, and test accuracies.

Figure 8 compares the train, validation, and test accuracies of all models. Each model exhibited higher training accuracy than validation accuracy, a clear indication of overfitting.

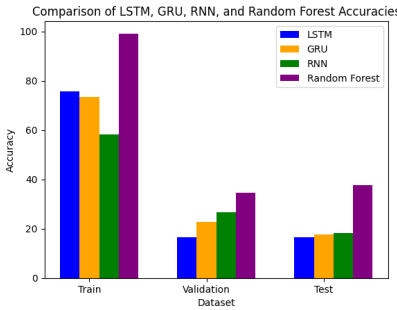

Figure 8: Accuracy Comparison of all models.

We can visualize, the Random Forest achieved a training accuracy of **100%** while the validation accuracy was only **34.66%**. Similarly, the RNN, LSTM, and GRU models demonstrated comparable patterns with peak training accuracies of **58.15%**,

75.54%, and **73.30%** respectively, contrasted with lower validation accuracies.

	RF	**RNN**	**LSTM**	**GRU**
Train accuracy	100	58.15	75.54	73.3
Validation accuracy	34.66	18.16	16.66	17.63
Test accuracy	37.61	18.16	16.66	17.63

Table 2: Accuracy of Random Forest, RNN, LSTM, and GRU

All of the models showed signs of overfitting, even though we used regularization techniques like L1, dropout, and hyperparameters tuning. Despite these efforts, we couldn't reduce the overfitting. This suggests that the problem is not with the models or methods we used but with the dataset itself. To improve the models' ability to work well on new data, the quality of the dataset needs to be improved by getting more diverse and representative data.

5 Conclusion

In this project, we explored the performance of multiple machine learning and deep learning models, including Random Forest, RNN, LSTM, and GRU, for sentiment analysis on tweet data. While each model showed strong results on the training set, all of them struggled with overfitting. Since even the simplest ML model was prone to overfitting, more complex models would likely overfit even more easily. This indicates that the issue is not with the models themselves but with the dataset.

To address this, future work should focus on improving the dataset quality by removing noise and possibly acquiring more diverse and representative data. By refining the dataset, the models could potentially generalize better and deliver more reliable predictions on unseen data. Despite the challenges faced in reducing overfitting, the project provides valuable insights into the complexities of sentiment analysis and highlights the importance of quality data in building robust models.

References

Mourad Azhari, Altaf Alaoui, Zakia Acharoui, Badia Ettaki, and Jamal Zerouaoui. 2019. Adaptation of the random forest method: solving the problem of pulsar search. pages 1–6.

Rahul Dey and Fathi M. Salem. 2017. Gate-variants of gated recurrent unit (gru) neural networks. In *2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)*, pages 1597–1600.

Weijiang Feng, Naiyang Guan, Yuan Li, Xiang Zhang, and Zhigang Luo. 2017. Audio visual speech recognition with multimodal recurrent neural networks. pages 681–688.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9:1735–80.

Saeed Mohsen. 2023. Recognition of human activity using gru deep learning algorithm. *Multimedia Tools and Applications*, 82.

Greg Van Houdt, Carlos Mosquera, and Gonzalo Nápoles. 2020. A review on the long short-term memory model. *Artificial Intelligence Review*, 53.