Twitter Platform Sentiment Analysis

Nguyen Tung Lam
21lam.nt@vinuni.edu.vn
CS, VinUniversity
Vietnam

Nguyen Duc Trung
22trung.nd@vinuni.edu.vn
CS, VinUniversity
Vietnam

Nguyen Nhat Nam
23nam.nn@vinuni.edu.vn
CS, VinUniversity
Vietnam

1 Introduction

In the era of digital communication, social media platforms such as Twitter, Facebook, Instagram, or Threads have emerged as prominent channels for individuals to express opinions, share experiences, and engage in public discourse. With millions of tweets generated daily, Twitter represents a vast and dynamic source of data that reflects public sentiment on a wide range of topics, including politics, consumer products, social issues, and global events. However, the unstructured and high-volume nature of these data presents significant challenges for manual analysis and interpretation.

Our project seeks to address the problem of efficiently analyzing public sentiment on Twitter through the application of sentiment analysis, which aims to classify textual data based on emotional tone. The ability to systematically analyze the sentiment on social networks has significant implications in multiple domains. For businesses, it enables the identification of customer satisfaction trends, supports reputation management, and informs product or service improvements. In the political arena, sentiment analysis offers insight into public opinion, facilitating more responsive and strategically informed campaign messaging. In marketing, sentiment analysis helps uncover consumer preferences and enhances the development of targeted advertising strategies, etc.

Through this project, our goal is to build a robust analytical framework that contributes to data-driven decision-making in both academic research and industry practice.

2 Dataset

Our project employs an adapted version of the *Sentiment Analysis: Emotion in Text* dataset originally curated by Figure Eight (now Appen) and hosted on Kaggle under a Creative Commons Attribution 4.0 International License [4]. The dataset comprises English-language tweets annotated with sentiment labels (positive, negative, or neutral). Although the original data set includes granular annotations that identify specific text spans (`selected_text`) that justify the assigned sentiment, this work focuses exclusively on the `text` and `sentiment` columns to train and evaluate sentiment classification models (Table 1).

The dataset is originally provided in two separate files: `train.csv` and `test.csv`. Specifically curated for sentiment analysis tasks, this dataset comprises 27,481 unique tweets for training and 3,534 for testing, offering a reliable foundation for training and evaluating models on social media text, particularly Twitter data. To facilitate model development and performance evaluation, the `train.csv` file is further partitioned into training and validation subsets using a stratified split to preserve the label distribution. Specifically, 80% of the data is used for training and 20% for validation. The `test.csv` file is retained as the final test set for evaluating the model's generalization performance.

Column Name	Data Type	Description
`text`	String	The text content of the tweet, containing the user's message in natural language. This is the primary input for sentiment analysis.
`sentiment`	String	Categorical label indicating *positive*, *negative*, or *neutral* emotion.

Table 1: Description of columns in the Twitter Sentiment Dataset

This dataset was chosen because of: (1) its curation through a Kaggle competition with a $15,000 prize pool underscores rigorous quality standards and community validation, (2) the social media origin provides authentic, noisy text that challenges models to handle real-world linguistic variations, and (3) its public availability with open source licensing enables reproducible experimentation.

3 Methods

3.1 Problem Statement

The central problem addressed in this study is the lack of an efficient and accurate automated system for analyzing sentiment in Twitter data. This project aims to develop such a system, capable of accurately classifying tweets into sentiment categories (positive, negative, or neutral) while accounting for the platform's linguistic nuances and contextual variability.

Formally, let $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ be a dataset consisting of n tweets, where:

- $x_i \in \mathbb{X}$ represents the i-th tweet as a sequence of words or tokens.
- $y_i \in \{\text{Positive}, \text{Negative}, \text{Neutral}\}$ denotes the corresponding sentiment label.

The goal is to learn a classification function:

$$f : \mathbb{X} \to \{\text{Positive}, \text{Negative}, \text{Neutral}\}$$

such that the predicted sentiment $\hat{y}_i = f(x_i)$ closely matches the true sentiment y_i, minimizing the classification error over the dataset.

3.2 Methodology

Approach: This study proposes a deep learning-based approach for sentiment classification of tweets, employing the *Bidirectional Encoder Representations from Transformers (BERT)* architecture and its variants. By fine-tuning these pre-trained language models on the *Sentiment Analysis: Emotion in Text* dataset [4], we aim to develop a high-performing classification system capable of discerning the

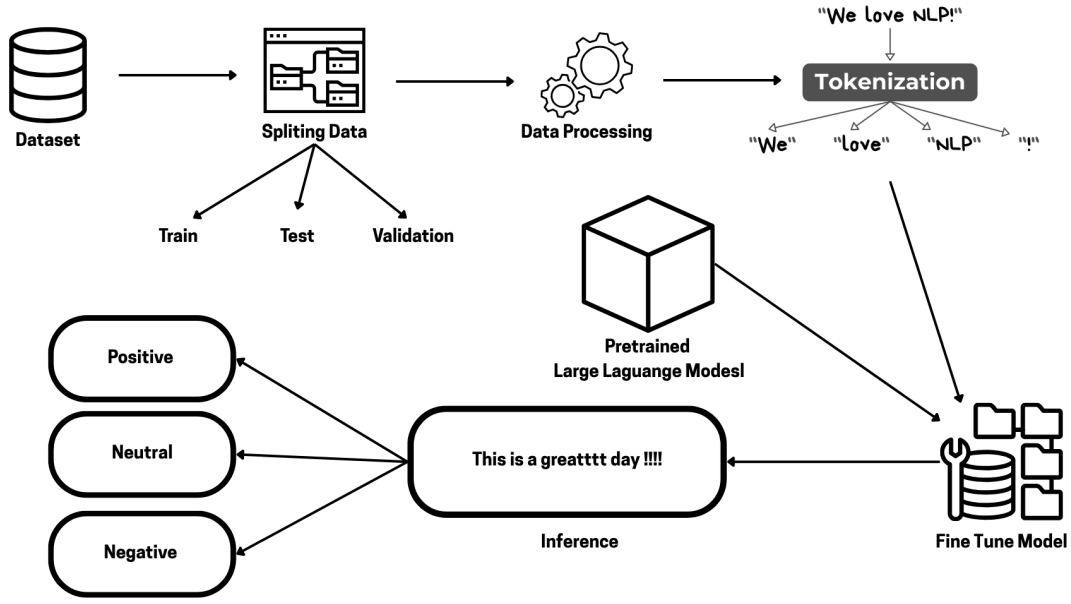

Figure 1: Overview of the Project Architecture

sentiment expressed in tweets. Leveraging the contextual understanding of BERT and its improved variants enables more nuanced and accurate sentiment detection in social media text.

Our method consists of the following stages (Figure 1):

(1) **Data Preprocessing**
Preprocessing steps are applied to clean and prepare the data for model training:
- **Remove rows with missing values**: Any row in the dataset with missing ('NaN') values is removed to ensure data quality.
- **Convert emoji to text**: Emojis are transformed into their corresponding text descriptions using the `emoji` library.
- **Remove URLs, emails, hashtags, and mentions**: Web addresses, email addresses, hashtags (keeping the keyword), and Twitter handles are removed to eliminate noise.
- **Expand contractions**: Common English contractions (e.g., "can't", "won't") are expanded to their full forms ("cannot", "will not") to improve clarity and consistency.
- **Normalize punctuation**: Repeated exclamation marks and question marks are replaced with placeholder tokens `<EXC>` and `<QUES>` respectively to retain their emphasis.
- **Reduce character repetitions and remove numbers**: Repeated letters in words (e.g., "soooo") are simplified, and numerical digits are removed as they often carry less semantic value in tweets.
- **Remove special characters and punctuation**: All special characters and punctuation (except predefined tokens like `<EXC>`, `<QUES>`) are removed.
- **Lowercase conversion**: All text is converted to lowercase to ensure case uniformity.
- **Normalize whitespace**: Extra spaces are reduced to a single space for consistency.

(2) **Model Training**
In this phase, we fine-tune pre-trained transformer-based language models for the task of sentiment classification. The following steps are involved:
- **Tokenization and Input Representation**: Tokenization is performed using the `AutoTokenizer` interface from the HuggingFace Transformers library, converting each tweet into token IDs and corresponding attention masks compatible with BERT-like models.
- **Model Selection**: Some possible transformer-based architectures can be evaluated, including:
  - `bertweet-base-sentiment-analysis`
  - `twitter-roberta-base-sentiment`
  - `distilbert-base-uncased`

  Each model will be fine-tuned using the `TFBertForSequenceClassification` (or equivalent for non-BERT models) with a softmax classification head.
- **Optimization and Loss Function**: The models are trained using the Adam optimizer with weight decay, and the categorical cross-entropy loss is employed to handle the multi-class sentiment classification task.
- **Hyperparameter Tuning**: Optimal values for training parameters such as learning rate, batch size, and

number of epochs are determined through empirical experimentation and grid search. Finally, we train the model with learning rate in the range $[1e{-}5, 5e{-}5]$, batch sizes of 16, and training for 4 - 7 epochs.

- **Finetuning Strategies**: Some finetuning strategies are applied to enhance performance and speed of training procedure.
  - **Gradient Accumulation**: To effectively increase the batch size without exceeding memory limits, gradient accumulation was set to 8 - 12 steps. This allows the model to simulate a bigger batch size, stabilizing updates and improving convergence.
  - **Warmup Scheduler**: A warmup scheduler was used to gradually increase the learning rate at the start of training, helping to avoid unstable updates in the early phase.
  - **Cosine Learning Rate Scheduler**: The learning rate was scheduled using a cosine decay, which helps fine-tune the model more smoothly toward the end of training.
  - **Mixed Precision Training**: The use of FP16 (half-precision) training reduced memory usage and accelerated training without compromising model accuracy.

(3) **Model Evaluation**: After training, the model is evaluated on the hold-out test set. Performance is quantified using standard classification metrics, including **Accuracy**, **Precision**, **Recall**, and **F1-score**.

4 Experiments

4.1 Evaluation Metrics

We utilize a set of evaluation metrics well-suited for multi-label classification tasks and imbalanced datasets: Accuracy, precision, recall, and F1 score.

4.2 Baseline

We adopt a Long Short-Term Memory (LSTM) network [2] as our baseline model, a widely used standard in sequential text processing tasks. Long short-term memory (LSTM) is a type of recurrent neural network (RNN) aimed at mitigating the vanishing gradient problem commonly encountered by traditional RNNs.

- **Embedding Layer**: Learned embeddings initialized randomly, mapping input tokens to 300-dimensional vectors
- **Regularization**: Immediate dropout layer with 50% rate to prevent feature overfitting
- **LSTM Layer**: Single directional LSTM with 100 hidden units, employing both input dropout (20%) and recurrent dropout (20%) for sequence modeling
- **Output Layer**: Dense layer with 3 neurons using softmax activation for multi-class classification.

On the implementation side, EarlyStopping and ModelCheckpoint were employed to stop the model from overfitting once the validation accuracy no longer improves.

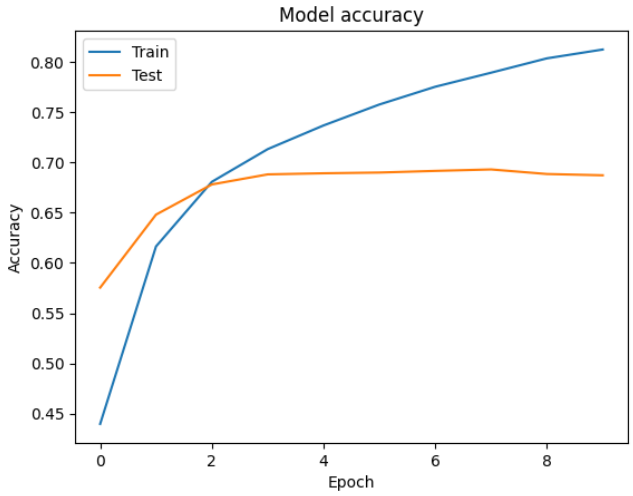

Figure 2: Train accuracy of the baseline LSTM model

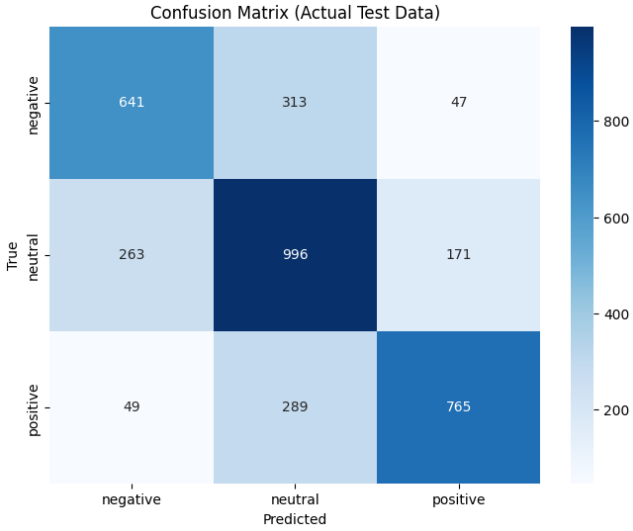

Figure 3: LSTM Prediction Confusion matrix

4.3 Results & Discussion

All models were evaluated on the same dataset, using the provided `test.csv` file from the *Sentiment Analysis: Emotion in Text* dataset [4]. The dataset consists of labeled text samples for sentiment classification. For each models, we use different sets of hyper-parameters for highest accuracy, refer to Table 2.

As shown in Table 3, BERTweet (a BERT-based model pre-trained specifically on Twitter data) achieved the highest scores across all metrics, with an Accuracy and F1 Score of 0.81. This superior performance can likely be attributed to its domain-specific pretraining, which provides the model with a nuanced understanding of social media text—characterized by informal language, abbreviations, hashtags, and emoticons—that is prevalent in the test dataset. This

Model	# Epochs	Batch Size	Learning Rate	Weight Decay	Grad. Accum. Steps	Warmup Steps
DistilBERT	4	16	2e-5	0.01	8	687
BERT	4	16	3e-5	0.01	8	500
RoBERTa	7	16	5e-6	0.1	12	1000

Table 2: Training Hyperparameters for Different Models

Model	Model Path	Accuracy	Precision	Recall	F1 Score
DistilBert	distilbert-base-uncased	0.78	0.78	0.78	0.78
Bert	finiteautomata/bertweet-base-sentiment-analysis	0.81	0.81	0.81	0.81
RoBERTa	cardiffnlp/twitter-roberta-base-sentiment	0.80	0.80	0.80	0.80
LSTM		0.69	0.69	0.69	0.69

Table 3: Performance Comparison of Different Models

domain alignment enables BERTweet to better capture sentiment-related linguistic patterns that may be overlooked by more general-purpose models.

RoBERTa, another Transformer-based architecture, also demonstrated strong performance, achieving an Accuracy and F1 Score of 0.80. While slightly trailing BERTweet, RoBERTa's results reaffirm the effectiveness of Transformer architectures in sentiment classification tasks. Notably, RoBERTa was not pre-trained on Twitter data, yet it still delivered robust results, underscoring the power of large-scale pretraining and contextual embeddings.

DistilBERT, a lighter and more computationally efficient version of BERT, maintained competitive performance with an F1 Score of 0.78. While slightly less accurate than BERTweet and RoBERTa, DistilBERT offers an appealing trade-off between computational efficiency and classification performance, making it suitable for deployment in resource-constrained environments.

In contrast, the Long Short-Term Memory (LSTM) model substantially underperformed across all metrics, achieving an Accuracy and F1 Score of only 0.69. Despite its historical success in sequence modeling, the LSTM model's relatively poor performance suggests that it struggles to capture the complex contextual dependencies necessary for effective sentiment analysis, particularly in noisy and informal text domains such as social media. Moreover, the LSTM model lacked the benefit of transfer learning, which has become a significant advantage for Transformer-based models through large-scale pretraining on diverse corpora.

4.4 Implementation

Our implementation can be found at this Github Repository.

5 Limitations

Despite the considerable theoretical appeal of **transformer-based architectures**—whose deep self-attention mechanisms and contextualized embedding spaces have demonstrably advanced numerous natural language processing tasks—the empirical efficacy of our fine-tuned model was materially constrained by the modest scale and homogeneity of the training corpus, which comprised a mere **25,000 labeled instances**. In practice, transformer models derive their representational potency from exposure to extensive and diverse data distributions; absent sufficiently large fine-tuning sets, these architectures frequently exhibit overfitting to idiosyncratic patterns and fail to capture the long-tail phenomena inherent in real-world language use. Accordingly, our transformer implementation attained only **78–81% classification accuracy**, a performance plateau that underscores the architecture's sensitivity to sample size and topical breadth during parameter adjustment. By contrast, classical machine learning paradigms—exemplified by **Support Vector Machines** and **Multinomial Naive Bayes classifiers**—leveraged manually engineered or statistically derived feature representations and benefited from inductive biases that naturally regularize learning in low-data regimes, thereby achieving substantially higher accuracy (**88–90%**) on the identical task [1] [3]. Moreover, practical constraints on computational resources curtailed exhaustive hyperparameter optimization, including exploration of alternative learning rate schedules, gradient accumulation strategies, and regularization coefficients, any of which might have marginally ameliorated the transformer's adaptability. However, we believe that the performance can still be improved, and we will discuss potential strategies for enhancement in more detail in Section 6.

6 Future Improvements

Future work on the project can focus on several specific areas to enhance its performance and analytical depth. First, the current implementation relies on a relatively small and general-purpose labeled dataset, which may not fully represent the linguistic diversity and informal nature of Twitter content. A key improvement would be to source or construct a larger, high-quality labeled dataset specifically curated from Twitter, incorporating domain-specific slang, emojis, abbreviations, and hashtags.

Additionally, the sentiment classification scheme in this project is limited to three basic categories: positive, negative, and neutral. This simplification fails to capture the emotional richness of tweets. Future versions could incorporate multi-class classification with finer emotional labels—such as anger, joy, sadness, disgust, fear, and surprise. Furthermore, the current model utilizes a pretrained transformer (e.g., BERT or RoBERTa) with minimal fine-tuning due to computational constraints. Given access to GPUs or TPUs,

Name	Contributions
Nguyen Tung Lam	- Comparison of different datasets and collecting a dataset, perform data preprocessing. - Implementation of BERT model
Nguyen Duc Trung	- Proposal of the team's workflow, problem statement definition, evaluation, discussion. - Implementation of BERT model.
Nguyen Nhat Nam	- Proposal and implementation of baseline LSTM model. - Refine dataset preprocessing.

Table 4: Contributions of team members to the project.

extensive fine-tuning using Twitter-specific corpora could significantly improve contextual understanding and sentiment accuracy, especially for tweets with ambiguous or sarcastic tones.

Another improvement involves implementing real-time tweet streaming using Twitter's API, allowing the system to track sentiment shifts live during events like political debates or public health announcements. A live dashboard built with platforms like Dash could then visualize trends and anomalies interactively. Furthermore, incorporating multilingual support would make the tool more globally relevant, especially by integrating models like XLM-R or mBERT for sentiment analysis in multiple languages or code-switched tweets.

Finally, sentiment analysis could benefit from temporal modeling—such as using rolling averages or recurrent neural networks to track sentiment evolution over time—especially for analyzing events or crisis response. These targeted improvements would make the sentiment analysis system significantly more robust, context-aware, and actionable for both academic research and practical applications.

7 Contributions

Our contributions are presented in Table 4.

8 Acknowledgements

We would like to express our deep and sincere gratitude to our research supervisor, **Prof. Wray Buntine**, for giving us the opportunity to conduct research and for providing invaluable guidance throughout this work. His dynamism, vision, sincerity, and motivation have deeply inspired us. He has taught us the methodology to carry out the research and to present our work as clearly as possible. It was a great privilege and honor to work and study under his guidance.

We are also greatly indebted to our honorable teaching assistant, **Ms. Vo Diep Nhu**, from the College of Engineering and Computer Science, VinUniversity. Without any doubt, her teaching and guidance have had a profound impact on our development.

We would like to say thanks to our friends and relatives for their kind support and care.

Finally, we would like to thank all the people who have supported us, directly or indirectly, in completing this project.

References

[1] Tyagi Ayush. 2020. Tweet Sentiment Predictions. https://www.kaggle.com/code/ayushtyagi10/tweet-sentiment-predictions#OPTION-1

[2] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9, 8 (1997), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

[3] what&how kvdprasad. 2020. Tweet Sentiment Predictions. https://www.kaggle.com/code/kvdprasad/tweet-sentiment-extraction

[4] Maggie, Phil Culliton, and Wei Chen. 2020. Tweet Sentiment Extraction. https://kaggle.com/competitions/tweet-sentiment-extraction. Kaggle.