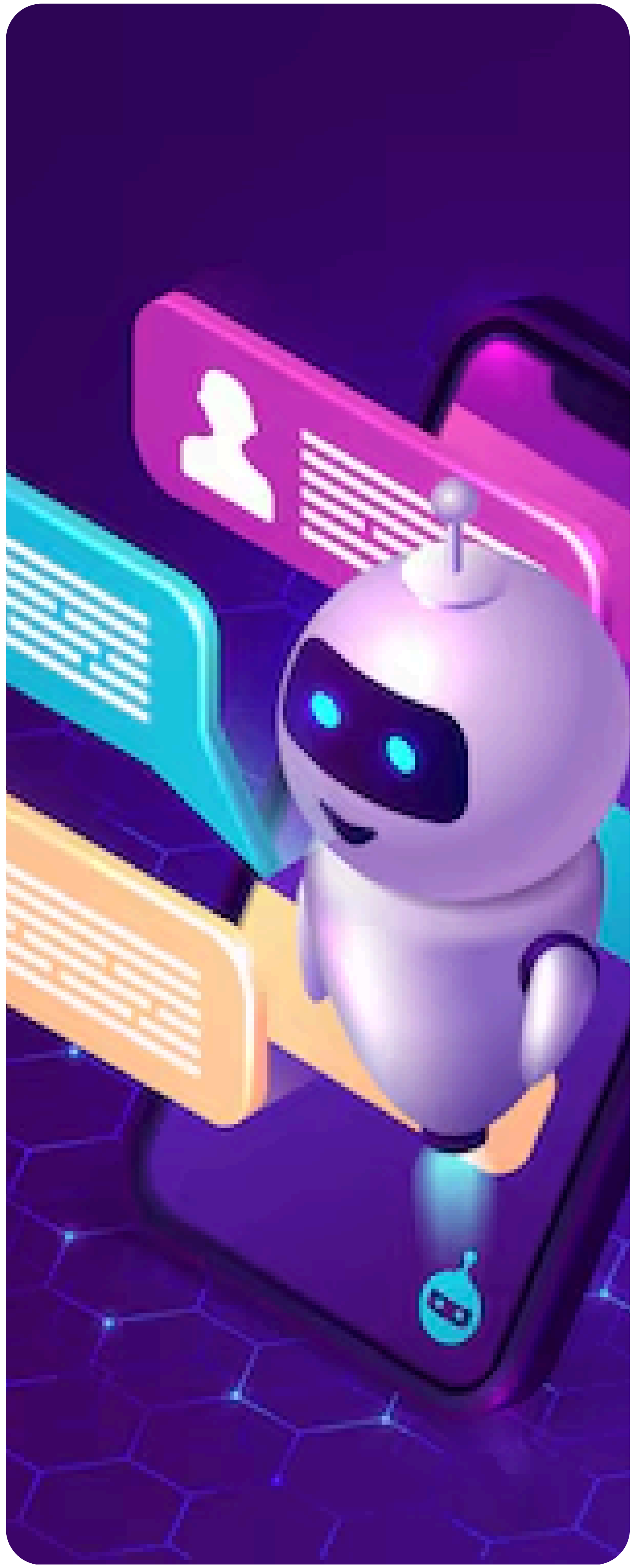
FINAL PROJECT

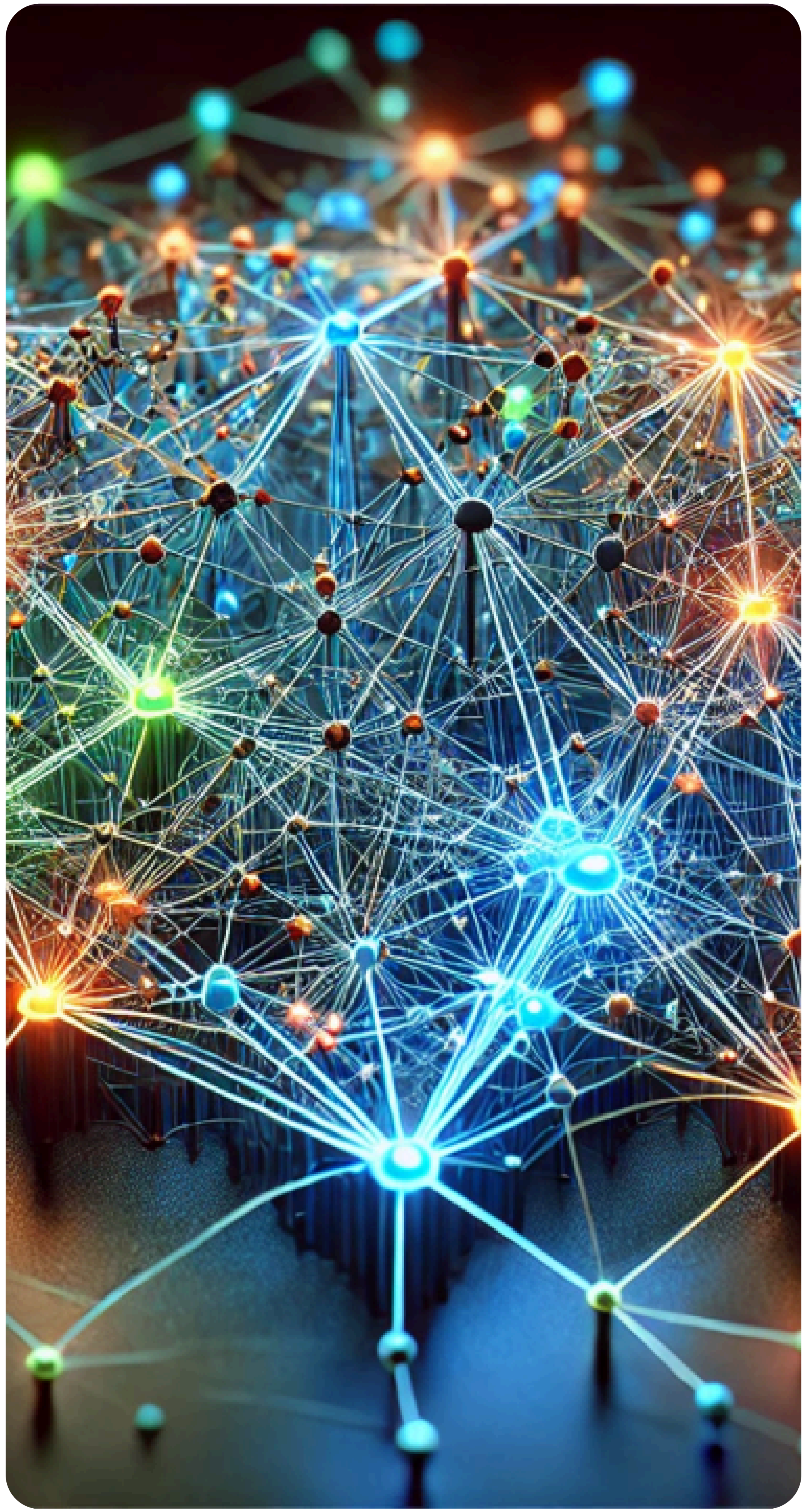
Twitter Platform Sentiment Analysis
Natural Language Processing - COMP4020

June 20, 2025

Team Nguyen Duc Trung - Nguyen Tung Lam - Nguyen Nhat Nam

Advisor Prof. Wray Buntine - CECS - VinUniversity
Vo Diep Nhu - CECS - VinUniversity





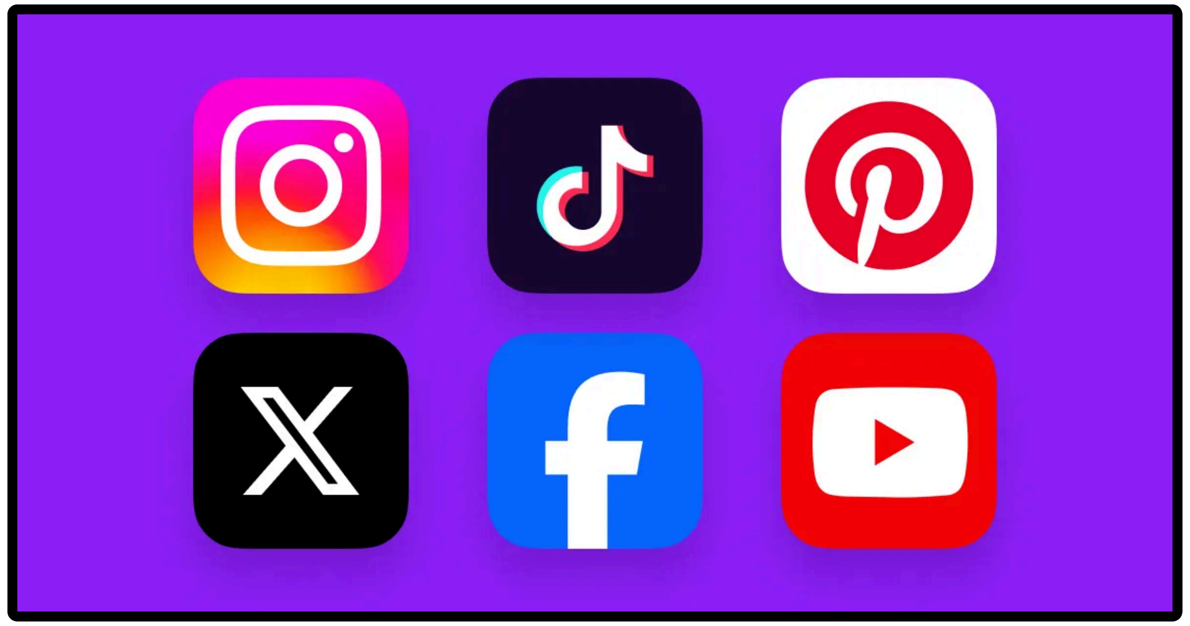
OUTLINE

INTRODUCTION

Motivation

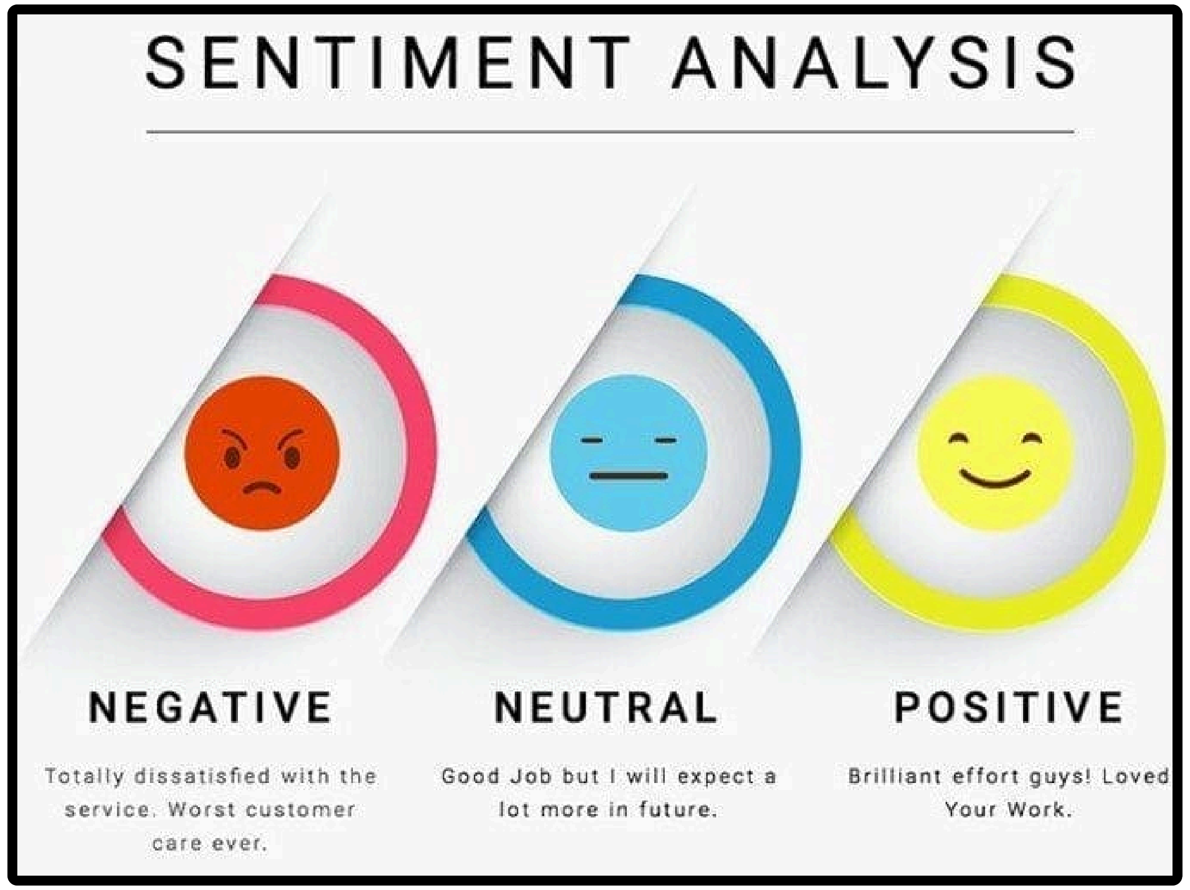
Social Media Platforms

- In the era of digital communication, social media platforms have emerged as prominent channels of communication.
- Millions of tweets daily, X represents a vast and dynamic source of data, reflects public sentiment on a wide range of topics.
- 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.



Problem Statement

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.
- $y_i \in$ {Positive, Negative, Neutral} denotes the corresponding sentiment label.

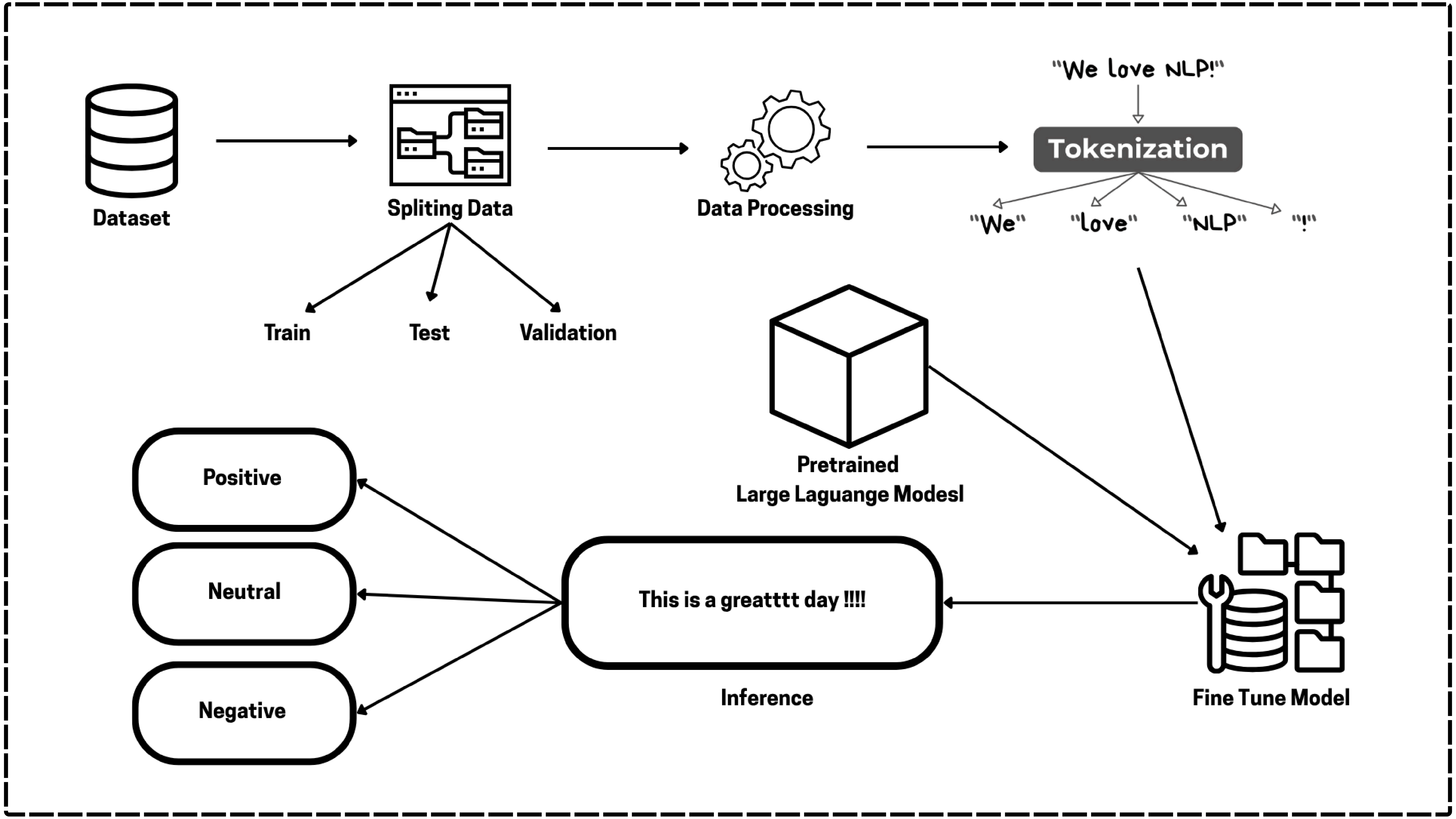
The goal is to learn a classification function:

f : X → {Positive, Negative, 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.

METHODOLOGY & IMPLEMENTATION

Overview of Project Architecture



Dataset

Adapted version of the Sentiment Analysis: Emotion in Text dataset originally curated by Figure Eight (now Appen) and hosted on Kaggle.

- Contain 2 files: *train.csv* and *test.csv* (***27,481*** unique tweets in *train* & ***3,534*** unique tweets in *test)*
- The *train.csv* file is split into ***80%*** training and ***20%*** validation sets using stratified sampling to preserve label distribution, while *test.csv* is reserved for final model evaluation.

Source: https://www.kaggle.com/competitions/tweet-sentiment-extraction/data

textID	text	selected_text	sentiment
27481 unique values	27481 unique values	22464 unique values	neutral 40% positive 31% Other (7781) 28%
cb774db0d1	I`d have responded, if I were going	I`d have responded, if I were going	neutral
549e992a42	Sooo SAD I will miss you here in San Diego!!!	Sooo SAD	negative
088c60f138	my boss is bullying me...	bullying me	negative
9642c003ef	what interview! leave me alone	leave me alone	negative

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.

Data Preprocessing

Preprocessing steps are applied to clean and prepare the data for model training:

- *Remove rows with missing values*
- *Convert emoji to text*
- *Remove URLs, emails, hashtags, and mentions*
- *Expand contractions*
- *Normalize punctuation*
- *Reduce character repetitions and remove numbers*
- *Remove special characters and punctuation*
- *Lowercase conversion*
- *Normalize whitespace*

Heyyyy!!! 😃 Can't wait to see you @party tonight!!! 🎉🎉 Check this out: https://fun.com #excited

hey cannot wait to see you at party tonight smiley face check this out excited

Transformer-based Models Training

Model Selection

- These transformer-based model are evaluated:
  - bertweet-base-sentiment-analysis
  - twitter-roberta-base-sentiment
  - distilbert-base-uncased

Hyperparameter Tuning

- Optimal learning rate, batch size, and epochs found via empirical testing and grid search.

Fine tuning Strategies

- Some finetuning strategies are applied to enhance performance and speed of training procedure.
  - Gradient Accumulation
  - Warmup Scheduler
  - Cosine Learning Rate Scheduler
  - Mixed Precision Training

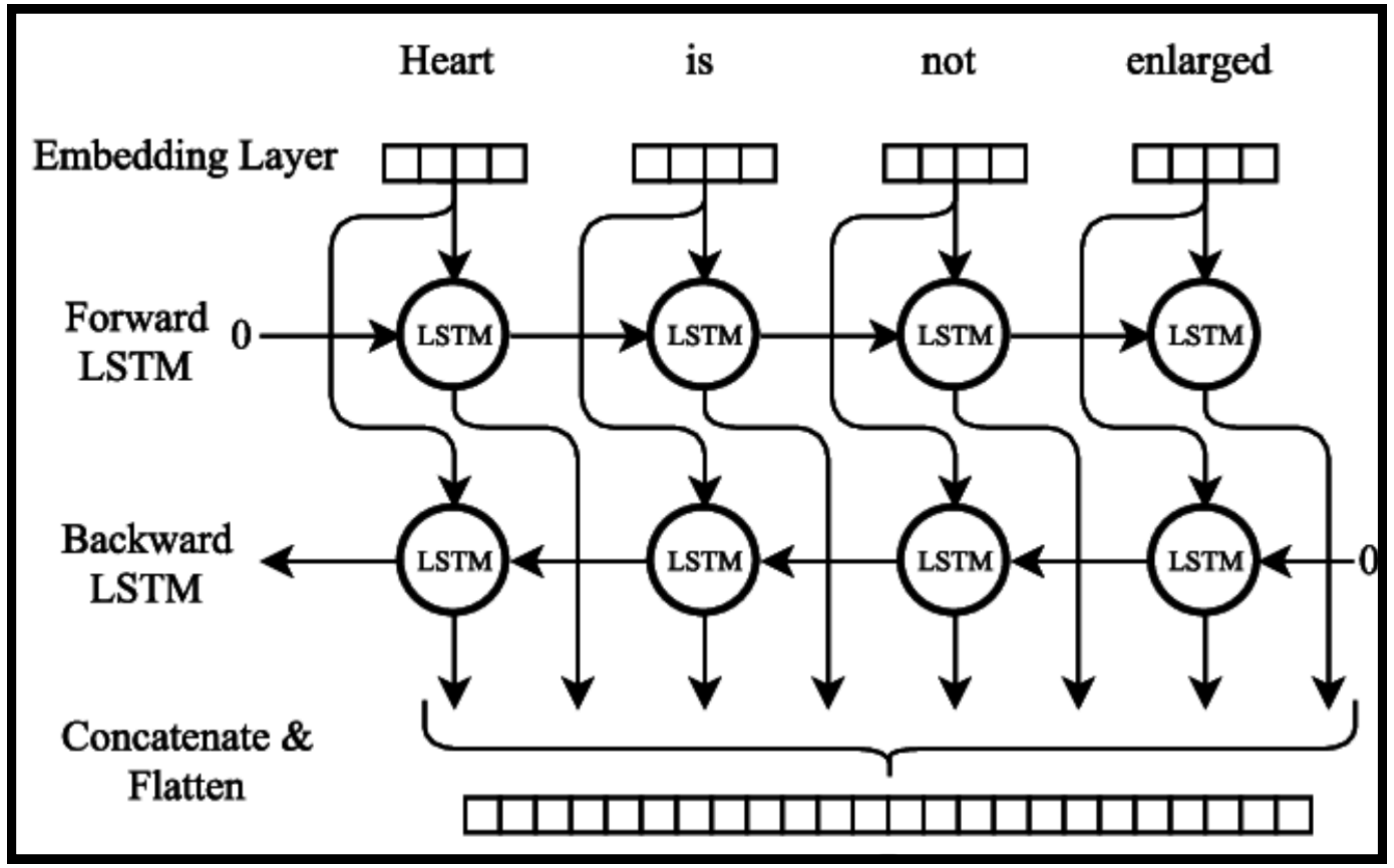
```
training_args = TrainingArguments(
    output_dir=output_dir,
    num_train_epochs=4,
    load_best_model_at_end=True,
    greater_is_better=True,
    eval_strategy="epoch",
    save_strategy="epoch",
    logging_strategy="steps",
    logging_steps=50,
    per_device_train_batch_size=16,
    per_device_eval_batch_size=16,
    gradient_accumulation_steps=8,
    learning_rate=3e-5,
    weight_decay=0.01,
    warmup_steps=500,
    lr_scheduler_type="cosine",
    fp16=True,
    save_total_limit=3,
    logging_dir='./logs',
    push_to_hub=False  # Set True only if needed
)
```

Baseline Model - LSTM

Architecture

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.



RESULTS & DISCUSSION

Evaluation Metrics

- **Accuracy**: The proportion of total predictions the model got correct ($\frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{Total}}$).
- **Precision**: Of all items the model labeled positive, the fraction that were actually positive ($\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}$).
- **Recall**: Of all actual positives in the data, the fraction the model correctly identified ($\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}$).
- **F1 Score**: The harmonic mean of precision and recall, balancing the two ($2 \times \frac{\mathrm{Precision} \times \mathrm{Recall}}{\mathrm{Precision}+\mathrm{Recall}}$).

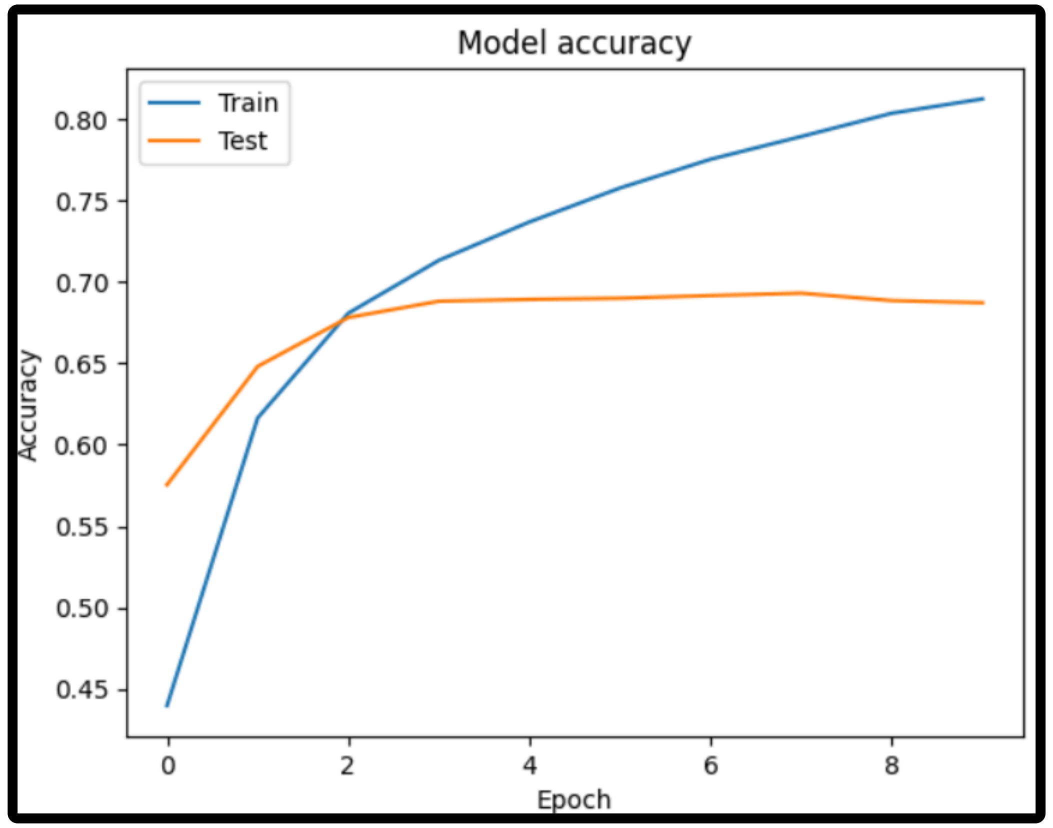
Baseline Model - LSTM

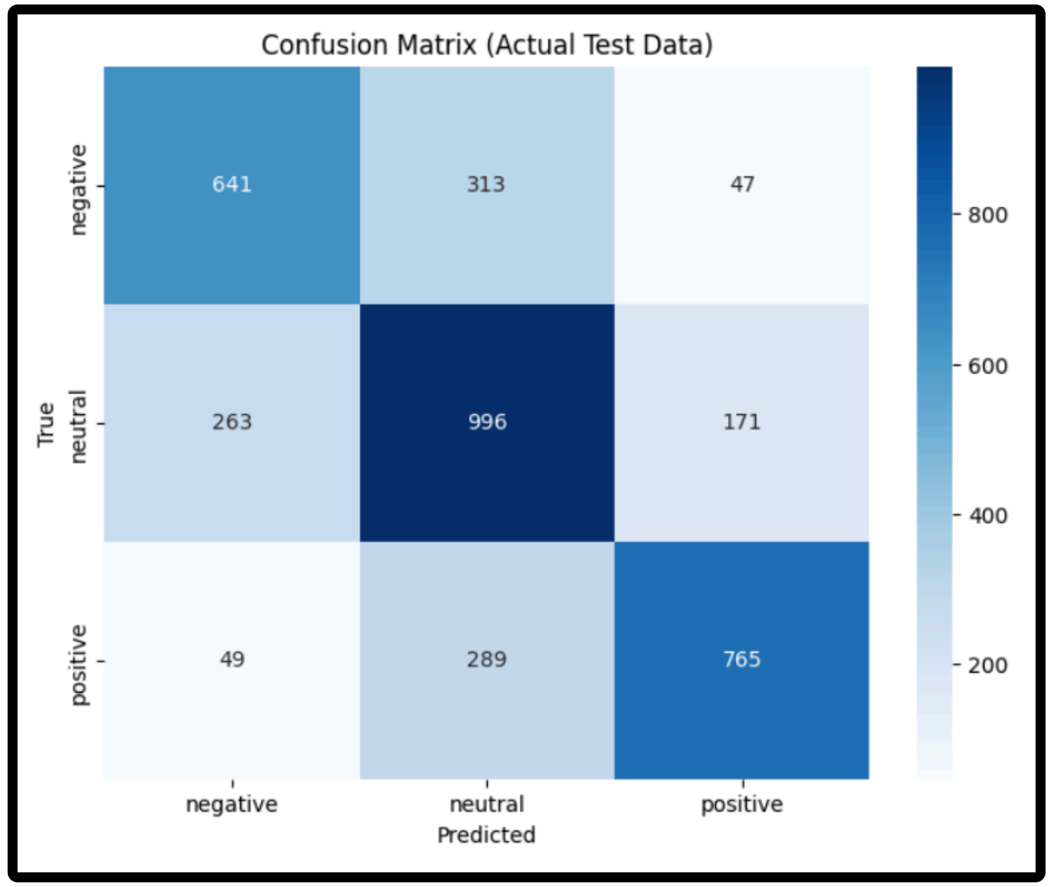
Result

The Long Short-Term Memory (LSTM) model achieved an Accuracy and F1 Score of only ***0.69***

The LSTM model shows a steady increase in training accuracy across epochs, reaching above 80%. Test accuracy plateaus early (~69%), indicating possible overfitting.

- Model performs best on the neutral class (996 correct predictions).
- Notable confusion between neutral and negative/positive classes.
- Misclassification is higher in the negative and positive categories, particularly mislabeling as neutral.





Transformer-based Models Performance

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

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

Discussion

LSTM is computationally cheaper but offers significantly lower performance.

DistilBERT offers a good balance between speed and accuracy, ideal for resource-constrained scenarios.

BERT and RoBERTa, while more resource-intensive (higher warmup steps, more accumulation steps), provide state-of-the-art results in sentiment classification.

Transformer-based models demonstrate significantly better generalization.

- BERT achieves the highest overall performance with 0.81 across accuracy, precision, recall, and F1 score, indicating balanced and consistent predictions.
- RoBERTa closely follows with 0.80, benefiting from deeper pretraining and longer training (7 epochs).
- DistilBERT, though lighter and faster, still outperforms LSTM by a wide margin.

Insights

- The performance gap clearly illustrates the limitations of traditional RNN-based models (like LSTM) for NLP tasks compared to modern transformers.
- Transformer architectures not only improve raw accuracy but also yield more reliable and balanced predictions across all sentiment classes.

CONCLUSION

Limitations

Resource Constraints

- Limited GPU/TPU availability prevented extensive hyperparameter sweeps
- Insufficient storage/memory “space” to ingest and process a much larger, more diverse dataset
- Insufficient variety in data led to overfitting common patterns and poor generalization to long-tail phenomena

Transformer Sensitivity

- Deep self-attention architectures require large, diverse corpora to unlock full representational power
- Support Vector Machines & Multinomial Naive Bayes (with engineered/statistical features) achieved 88–90 % accuracy on the same task

Future Improvements

Enrich & Expand Training Data

- Curate a large, high-quality Twitter-specific dataset
- Include slang, emojis, abbreviations, and hashtags for better domain coverage

Finer-Grained Emotion Labels

- Move beyond “positive/negative/neutral” to multi-class emotions (anger, joy, sadness, disgust, fear, surprise)
- Capture emotional nuance and mixed sentiments

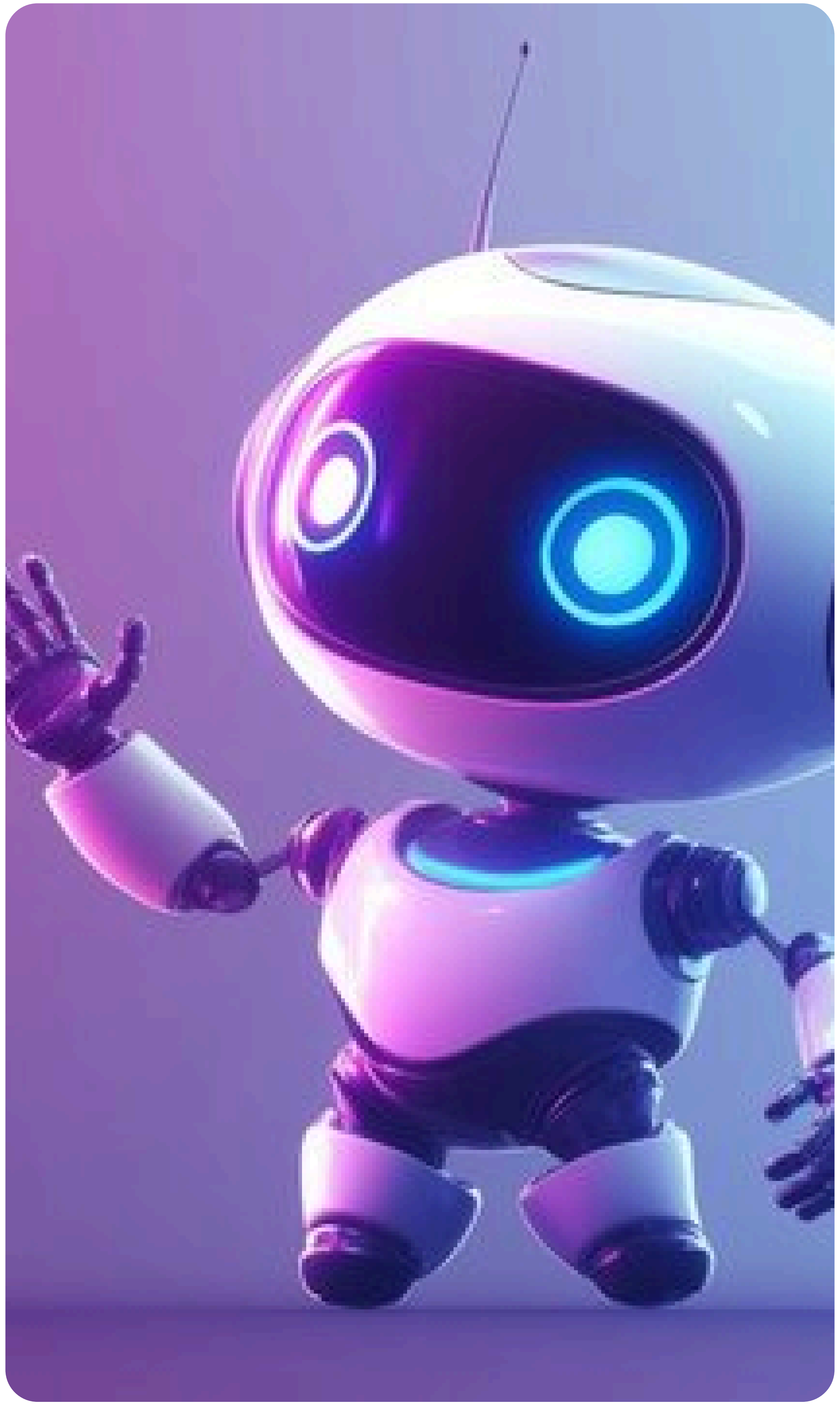
Multilingual & Code-Switching Support

- Integrate XLM-R or mBERT to handle multiple languages and mixed-language tweets
- Extend coverage to non-English and code-switched content

VINUNIVERSITY

THANK YOU

FOR YOUR ATTENTION !!!



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.

Q&A