Tweets Sentiment Analysis and Classification

Milestone 2 Report

By

Maryam AlBahri
albahri.m@northeastern.edu

Shan Lin
lin.shan1@northeastern.edu

Min Yao
yao.min1@northeastern.edu

July 7th , 2025

GitHub Repository

Repository Link: https://github.com/ariesslin/ie7500-g1-tweet-sentiment-nlp

Repository Status: Public

Milestone 2 Submission Overview

This report summarizes the work completed in Milestone 2 of our tweet sentiment analysis project, focusing on model development and implementation. In line with our proposal in milestone 1, we developed and compared multiple sentiment classification models for short, informal text.

We implemented three architectures: a TF-IDF + Logistic Regression as baseline, a Bidirectional LSTM for sequence modeling, and a DistilBERT transformer for contextual understanding. These models were implemented to investigate performance, complexity trade-offs, and generalization in tweet sentiment classification.

These implementations lay the groundwork for the next milestones, which will involve evaluation, error analysis, and final model selection.

Please refer to the GitHub repository for more details; we have included detailed analysis in the accompanying notebooks.

GitHub Repository Setup & Code Management

After completing Milestone 2, we found that handling all processes, such as EDA, preprocessing, model development, and evaluation in a single notebook, was inefficient and hard to manage. To improve clarity and workflow, **we decided to split the code into separate notebooks**, each dedicated to a specific stage of the project. This structure makes it easier to navigate, debug, and reuse components.

1. Repository Structure

```
.
├── data/ # Raw and external data
├── docs/ # Project documentation
├── models/ # Trained model files
├── processed_data/ # Cleaned datasets and splits
│ ├── preprocessed_tweets.csv
│ ├── train_dataset_comp.zip
│ ├── val_dataset.csv
│ ├── test_dataset.csv
│
├── scripts/ # Jupyter Notebooks
│ ├── 1-Exploratory-Data-Analysis-EDA.ipynb
│ ├── 2-Data-Preprocessing.ipynb
│ ├── 3-Model-Development.ipynb
│
├── utils/ # Helper functions
│ └── helper.py
├── requirements.txt # Dependencies
└── README.md # Project overview and guide
```

2. Version Control & Collaboration

- **Branching Strategy:** Trunk-based development, we commit in main branch
- **Commit History:** Regular commits with descriptive messages
- **Collaboration:** Team members working on separate components and then merge the notebooks
- **.gitignore:** Properly configured to exclude sensitive files and large datasets

3. Comprehensive Documentation Provided

- **README.md:** Complete project overview, methodology, and setup instructions
- **Project Proposal:** Milestone 1 project proposal in docs/ folder
- **Code Comments:** Extensive inline documentation in notebooks
- **Operation Instructions:** Environment setup and dependency installation in readme
- **Methodology Explanation:** Detailed explanation of model choices and evaluation approach in notebooks

Summary of Completed Tasks – Milestone 2

1. Define Objectives

Our NLP task focuses on **tweet sentiment classification**. A comprehensive definition of objectives and scope was outlined in Milestone 1. Please refer to the project proposal for details.

2. Literature Review

We conducted extensive research on state-of-the-art methods for sentiment analysis, such as:

Traditional Approaches:

- **TF-IDF + Logistic Regression:** Ramos (2003) demonstrated effectiveness for document classification
- **Bag-of-Words:** Wang & Manning (2012) showed strong baseline performance for text classification

Deep Learning Approaches:

- **LSTM Networks:** Hochreiter & Schmidhuber (1997) for sequential data processing
- **Bidirectional LSTM:** Zhou et al. (2016) for enhanced context understanding
- **BERT:** Devlin et al. (2019) for state-of-the-art transformer-based classification

Based on this research, we implemented three architectures: TF-IDF with Logistic Regression as a baseline, a Bidirectional LSTM for sequence modeling, and a DistilBERT transformer for contextual understanding. For more details, please refer to the project proposal and the code notebooks.

3. Dataset Preparation

Our data source is the **Sentiment140** dataset, which contains approximately 1.6M tweets.

Our preprocessing pipeline included the following steps:

1. **Text Cleaning:** Remove URLs, HTML entities, special characters
2. **Tokenization:** Word-level tokenization with NLTK
3. **Normalization:** Lowercasing, stemming, stop word removal

4. **Feature Engineering:** TF-IDF vectors, word embeddings, BERT tokenization
5. **Data Splitting:** Stratified split ensuring class balance

Our Data Split and Main Statistics:

- **Training Set:** 1,120,000 tweets (70%)
- **Validation Set:** 240,000 tweets (15%)
- **Test Set:** 240,000 tweets (15%)
- **Class Balance:** ~50% positive, ~50% negative

4. Framework/Library Selection

We explored and selected appropriate frameworks based on model requirements:

Traditional ML:

- **scikit-learn:** For TF-IDF vectorization and Logistic Regression
- **pandas/numpy:** For data manipulation and numerical operations

Deep Learning:

- **TensorFlow/Keras:** For LSTM model implementation
- **Hugging Face Transformers:** For BERT model fine-tuning
- **PyTorch:** Backend for BERT implementation

Supporting Libraries:

- **NLTK/spaCy:** For text preprocessing and tokenization
- **matplotlib/seaborn:** For visualization and evaluation plots

5. Preliminary Experiments

To ensure development feasibility, we conducted several small-scale tests:

- **Sanity Check:** LSTM model tested on 100 samples to verify overfitting capability
- **Hyperparameter Tuning:** GridSearchCV for Logistic Regression with 96 parameter combinations
- **Data Validation:** Ensured proper train/validation/test splits (70%/15%/15%)

6. Model Development Key Parts

Model 1: TF-IDF + Logistic Regression

```
# Implementation in scripts/3-Model-Development.ipynb
pipeline = Pipeline([
  ('tfidf', TfidfVectorizer(max_features=10000, ngram_range=(1, 2))),
  ('clf', LogisticRegression(max_iter=1000, random_state=42, class_weight='balanced'))
])
```

Model 2: Bidirectional LSTM

```
# Implementation in scripts/3-Model-Development.ipynb
LSTMmodel = Sequential([
  Embedding(input_dim=vocab_size, output_dim=embedding_dim,
weights=[embedding_matrix]),
  Bidirectional(LSTM(265, return_sequences=True)),
  Dropout(0.5),
  Bidirectional(LSTM(128)),
  Dropout(0.3),
  Dense(1, activation='sigmoid')
])
```

Model 3: DistilBERT

```
# Implementation in scripts/3-Model-Development.ipynb
BERTmodel = DistilBertForSequenceClassification.from_pretrained(
  'distilbert-base-uncased', num_labels=2
)
```

7. Training & Fine-tuning

Below are the key details from the training and fine-tuning of our models:

Logistic Regression:

- **Hyperparameter Tuning:** GridSearchCV with 96 combinations
- **Best Parameters:** C=1, penalty='l2', class_weight=None
- **Training Time:** ~5 minutes

LSTM:

- **Optimizer:** Nadam with learning rate 0.001
- **Regularization:** Dropout (0.5, 0.3), BatchNormalization
- **Training:** 20 epochs with early stopping
- **Training Time:** ~2 hours

BERT:

- **Fine-tuning:** DistilBERT base model
- **Training Arguments:** 1 epoch, batch size 16, learning rate 2e-5
- **Training Time:** ~4 hours

8. Evaluation & Metrics

We used the following evaluation metrics to assess model performance:

- **Accuracy:** Overall classification correctness
- **Precision:** Positive predictive value
- **Recall:** Sensitivity/true positive rate
- **F1 Score:** Harmonic mean of precision and recall
- **Confusion Matrix:** Detailed error analysis

9. Benchmarking

We compared models based on multiple criteria: Training Time, Interpretability, Computational Cost, Model Complexity. Please refer to the next section to see our findings.

Key Findings & Results

Model Performance Summary Observations

Based on our model development and preliminary training, we have successfully implemented three distinct approaches for tweet sentiment analysis. The advanced models (LSTM and BERT) demonstrate improved performance capabilities compared to the traditional Logistic Regression baseline, which is expected given their ability to capture complex linguistic patterns and contextual information.

Model Selection Trade-offs Analysis

Aspect	Logistic Regression	LSTM	BERT
Speed	High	Medium	Low
Accuracy	Medium	High	High
Interpretability	High	Medium	Low
Resource Usage	Low	Medium	High

Next Steps & Future Work

The following are our planned improvements for the next milestone:

1. **Comprehensive Model Evaluation:** Detailed performance analysis on test dataset
2. **Final Model Selection:** Complete model selection using validation data
3. **Error Analysis:** Identify common mistakes and failure cases
4. **Performance Metrics Comparison:** Detailed comparison with baselines and benchmarks
5. **Model Refinement:** Hyperparameter tuning and architecture optimization