**CS918 NLP Coursework Report**
**Sentiment Classifier for Twitter Data**

Aditya Mahamuni
5579267

# 1 Abstract

This report presents the development and evaluation of multiple sentiment analysis classifiers for analyzing Twitter data. The approach leverages traditional machine learning algorithms namely Logistic Regression (MaxEnt), Support Vector Machines (SVM), and Naive Bayes. Along with these traditional machine learning algorithms, advanced deep learning technique using a Long Short-Term Memory (LSTM) network enhanced with pre-trained GloVe embeddings is being employed for the classification task. The performance of each model is evaluated across three different test sets to ensure generalisability.

# 2 Introduction

Sentiment analysis of Twitter data has become an essential task in understanding public opinion on various topics. Twitter has become a base for expressions opinions and sharing personal experiences, or even discussing variety of topics. This makes it an interesting playground for sentiment analysis. The SemEval 2017 Task 4 (Rosenthal, Farra, and Nakov 2017) provides a platform for developing and benchmarking sentiment classifiers on a standardized dataset. The report focuses on Subtask A, aiming to classify the sentiment of tweets into positive, negative, or neutral. This report outlines our approach to tackle this classification problem, covering data preprocessing, feature extraction, model development, and evaluation.

# 3 Description

The dataset for this project, semeval-tweets, is sourced from Task 4 of Semeval 2017. It comprises five files: one for training (twitter-training-data.txt), one for development (twitter-dev-data.txt), and three subsets for testing (twitter-test[1-3].txt). These datasets are structured as tab-separated values (TSV) with each row representing a tweet. Each row contains the tweet ID, sentiment (positive, negative, neutral), and tweet text.

# 4 Methodology

## 4.1 Data Preprocessing

Initially, the TSV file is loaded into a pandas (McKinney et al. 2020) dataframe using pandas and then pre-processing is applied thereafter. The pre-processing stage involves cleaning the tweets to remove noise and normalize the text. This includes removing mentions (@), hashtags(#), URLs, non-alphanumeric characters, and single characters, which do not contribute to sentiment. After the pre-processing is finished further pre-processing is performed. This includes tokenizing the tweets, and removing the stopwords for reducing dimensionality. Lemmatization is then applied to these tweets to condense the words to their base forms, which ensures consistent representation across whole corpus.

## 4.2 Feature Engineering

Two main types of features are explored in our classifiers:

1. **Bag of Words (BoW) with TF-IDF:**

   This traditional NLP feature represents tweets as vectors indicating the frequency of words, weighted by their inverse document frequency across the corpus (Dauda Abubakar and Umar 2022).

2. **GloVe Word Embeddings:**

   For the LSTM model, we utilize pre-trained GloVe embeddings to convert words into dense vector representations that capture semantic similarities (Pennington, Socher, and C. Manning 2014). The glove.6B was used for the embedding task by download the pre-trained embeddings from 2014 English Wikipedia. This file contains 100-dimensional embedding vectors for 400,000 words (or non-word tokens) which can be used directly for the embedding task (Pennington, Socher, and C. D. Manning 2014).

### 4.3 Model Development

We develop three traditional machine learning models and a deep learning model:

1. **Logistic Regression and SVM:**

   These models are trained using BoW features with TF-IDF weighting as well as Glove features. These classifiers are optimized through grid search for hyperparameters. These models are implemented using scikit-learn library (Pedregosa 2024).

2. **Naive Bayes:**

   Given its probabilistic nature, this model is applied to BoW features, focusing on the probability of word occurrences in positive, negative, or neutral tweets. This model is also being implemented using scikit-learn library.

3. **LSTM with GloVe Embeddings:**

   A deep learning model that processes tweets as sequences of GloVe embeddings, capturing long-term dependencies and semantic nuances through LSTM layers (Guthrie 2024). LSTM is beiing implemented using pytorch.

### 4.4 Evaluation

Models are evaluated based on their macro weighted F1 score, across three separate test sets. This multi-test set evaluation helps assess the generalisability of our classifiers.

## 5 Results

The results are being performed at the earlier stage without pre-processing and a baseline evaluation is captured to judge the performance gain after pre-processing and hyper-parameter optimisations. Based on the macro weighted F1 scores for different classifiers on various test sets, the LSTM classifier outperforms SVM, Logistic Regression (LR), and Naive Bayes (NB). Here's a brief analysis and discussion on the results:

### 5.1 Performance without Pre-processing and Hyperparameter Optimisation

| Model | Testset 1 | Testset 2 | Testset 3 |
|---|---|---|---|
| SVM (Support Vector Machine) | 0.432 | 0.457 | 0.446 |
| Logistic Regression (MaxEnt) | 0.425 | 0.454 | 0.438 |
| Naive Bayes (NB) | 0.295 | 0.343 | 0.307 |
| LSTM (Long Short-Term Memory) | 0.453 | 0.469 | 0.472 |

Table 1: Model Performance on without Pre-processing or Hyperparameter tuning

1. **SVM (Support Vector Machine)**:

   The SVM classifier shows moderate performance across all three test sets, with the highest score on the second test set. SVMs are effective in high-dimensional spaces and when there is a clear margin of separation in the data.

2. **Logistic Regression (MaxEnt)**:

   Logistic Regression shows slightly lower performance compared to SVM on the first test set. LR is a good baseline model for classification problems and performs well when the data is linearly separable.

3. **Naive Bayes (NB)**:

   Naive Bayes shows the lowest performance among the classifiers for the test sets. While NB is known for its simplicity and efficiency, especially in text classification tasks, its assumption of feature independence might not hold true in practice, which might affect its performance.

4. **LSTM (Long Short-Term Memory)**

   The LSTM classifier achieves the highest macro weighted F1 score among the classifiers on the testsets. LSTMs are particularly suited for sequence prediction problems due to their ability to capture long-term dependencies. Their strong performance here suggests that capturing contextual information in tweet texts significantly contributes to sentiment analysis accuracy.

The models tested without pre-processing or hyperparameter tuning show varied performance. The SVM and Logistic Regression models provide moderate accuracy, highlighting their strength in high-dimensional spaces and linear separability, respectively. Naive Bayes underperforms, likely due to its assumption of feature independence, which may not hold in complex datasets. LSTM leads in performance, demonstrating its capability to capture long-term dependencies and contextual information, crucial for tasks like sentiment analysis in sequences of text. The running time for training and evaluation for all the classifiers took less than half hour.

### 5.2 Performance without Pre-processing and Hyperparameter Optimisation

| Model | Testset 1 | Testset 2 | Testset 3 |
|---|---|---|---|
| SVM (Support Vector Machine) | 0.502 | 0.516 | 0.474 |
| Logistic Regression (MaxEnt) | 0.535 | 0.553 | 0.504 |
| Naive Bayes (NB) | 0.318 | 0.371 | 0.312 |
| LSTM (Long Short-Term Memory) | 0.546 | 0.554 | 0.515 |

Table 2: Model Performance with Pre-processing and Hyperparameter Optimisation

1. **SVM (Support Vector Machine):**

   Exhibits moderate performance with scores that suggest it handles the datasets with some degree of variability, indicating it might be sensitive to the nuances of the test sets.

2. **Logistic Regression (MaxEnt):**

   Shows strong performance across all test sets, particularly standing out on Testset 2. This suggests a good fit for the data and possibly a better generalization capability compared to the other models.

3. **Naive Bayes (NB):**

   Demonstrates significantly lower performance compared to the other models, indicating it may struggle with the complexity or specific characteristics of the data in these test sets.

4. **LSTM (Long Short-Term Memory):**

   Delivers the best overall performance, especially on Testset 2 and 3, highlighting its capability to capture dependencies in the data that the other models may not leverage as effectively.

This results clearly shows that after performing valid pre-processing and performing hyperparameter optimisations, the performance of the classifiers has increased. LSTM is marginally outperforming SVM and Logistic Regression. Naive Bayes classifier has been showing the lowest results among all the classifiers. It is clear to see that the pre-processing steps and hyper-parameter tuning have contributed to the increase in the performance of the classifiers. The running time for training and evaluation for all the classifiers took around 24 minutes.

## 6 Discussion

The results suggest that for sentiment analysis tasks, models that can understand and leverage contextual and sequential information in text (like LSTM) tend to perform better compared to traditional machine learning models that rely on bag-of-words or TF-IDF feature representations (SVM, LR, NB).

In future work, we can explore more sophisticated neural network architectures such as Transformer-based models especially models such as GPT, that have shown state-of-the-art performance on various NLP tasks, including sentiment analysis. Additionally, fine-tuning and hyperparameter optimization, as well as advanced feature engineering, could further improve model performance.

## 7 Conclusion

In this report, a comprehensive analysis of different sentiment classification techniques for Twitter data is presented. The findings highlight the strengths and weaknesses of traditional machine learning models and the potential of deep learning approaches when combined with pre-trained word embeddings. Future work could explore more complex neural architectures and the integration of additional tweet-specific features.

# References


Dauda Abubakar, Haisal and Mahmood Umar (Aug. 2022). "Sentiment Classification: Review of Text Vectorization Methods: Bag of Words, Tf-Idf, Word2vec and Doc2vec". In: *SLU Journal of Science and Technology* 4, pp. 27–33. DOI: 10.56471/slujst.v4i.266.

Guthrie, Robert (2024). *Sequence Models and Long Short-Term Memory Networks*. https://pytorch.org/tutorials/beginner/nlp/sequence_models_tutorial.html. Accessed: 2024-03-21.

McKinney, Wes et al. (2020). *pandas: A Fast, Powerful, Flexible and Easy to Use Open Source Data Analysis and Manipulation Tool, Built on Top of the Python Programming Language*. Python version 3.10 or above required. URL: https://pandas.pydata.org/.

Pedregosa, F. et al. (2024). *scikit-learn: Machine Learning in Python*. https://scikit-learn.org/.

Pennington, Jeffrey, Richard Socher, and Christopher Manning (2014). "GloVe: Global Vectors for Word Representation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543.

Pennington, Jeffrey, Richard Socher, and Christopher D. Manning (2014). *GloVe: Global Vectors for Word Representation*. https://nlp.stanford.edu/projects/glove/.

Rosenthal, Sara, Noura Farra, and Preslav Nakov (2017). *SemEval-2017 Task 4: Sentiment Analysis in Twitter*. https://alt.qcri.org/semeval2017/task4/.