Project Document: Twitter Sentiment Analysis Using LSTM

# Abstract

This project performs sentiment analysis on Twitter data using a Long Short-Term Memory (LSTM) neural network. The model classifies tweets as positive, negative, or neutral by analyzing their textual content. Implemented in Google Colab using TensorFlow and Keras, this document provides an in-depth explanation of the code and methodology used.

# 1. Introduction

Sentiment analysis is a key application of Natural Language Processing (NLP), often used to gauge public opinion on social media. LSTM models are well-suited for text classification tasks due to their ability to retain context over long sequences. This project focuses on classifying tweet sentiments using an LSTM model.

# 2. Dataset Description

The dataset consists of labeled tweets categorized as positive, negative, or neutral. It is preprocessed to remove noise such as URLs, mentions, hashtags, and special characters.

# 3. Code Explanation

\*\*Step 1: Import Required Libraries\*\*

Libraries like pandas, numpy, TensorFlow, Keras, matplotlib, and nltk are imported to handle data, build the model, and visualize results.

\*\*Step 2: Data Loading and Preprocessing\*\*

Tweets are loaded into a DataFrame. The preprocessing steps include:  
- Lowercasing text  
- Removing punctuation, URLs, mentions  
- Removing stopwords and performing tokenization using nltk  
- Encoding sentiments to numerical labels  
- Padding sequences to ensure equal input length

\*\*Step 3: Text Tokenization and Embedding\*\*

Keras Tokenizer is used to convert words into sequences of integers. These sequences are then padded to create fixed-length inputs. An Embedding layer is used in the model to convert these integers into dense vector representations.

\*\*Step 4: Define the LSTM Model\*\*

The LSTM model is defined using the Keras Sequential API:  
- Embedding layer for word vector representation  
- LSTM layer with dropout for sequence learning  
- Dense layers for classification using softmax activation  
This architecture captures long-term dependencies and classifies the sentiment effectively.

\*\*Step 5: Compile and Train the Model\*\*

The model is compiled using 'categorical\_crossentropy' loss and 'adam' optimizer. It is trained using the `fit()` method with validation data over several epochs.

\*\*Step 6: Evaluate the Model\*\*

Model performance is evaluated on test data using metrics such as accuracy, precision, recall, and F1-score. Confusion matrix and training/validation loss curves are plotted to assess the model's performance.

\*\*Step 7: Predict on New Data\*\*

The model is used to predict sentiments on new tweet inputs. These predictions are decoded back into human-readable labels.

# 4. Results

The model achieves good accuracy on sentiment classification. Visualization tools like confusion matrix, accuracy/loss plots, and sample predictions provide insights into the model's effectiveness.

# 5. Conclusion

LSTM networks are effective in handling sequential data like tweets for sentiment analysis. This project successfully demonstrates a deep learning pipeline from data preprocessing to model evaluation using Google Colab.

# 6. References

[1] Twitter Sentiment Dataset - Kaggle  
[2] TensorFlow/Keras Documentation - https://www.tensorflow.org/  
[3] NLTK Documentation - https://www.nltk.org/