NLP Project Report

Rahma Mourad 202201407 **Nourhan Deif 202201407**

1. Introduction

The goal of this project was to implement a sentiment analysis model using a Long Short-Term Memory (LSTM) network. The model was trained to classify movie reviews as either positive or negative. The project utilized a dataset of 15,000 IMDB reviews, and the aim was to evaluate the model's performance in terms of classification accuracy, precision, recall, and F1-score. Additionally, the model was deployed using **FastAPI** to allow real-time sentiment predictions via an API.

2. Methodology

2.1 Data Preprocessing

- **Text Cleaning**: HTML tags were removed from the reviews using **BeautifulSoup**.
- **Lemmatization**: The text was lemmatized using **spaCy** to reduce words to their base forms (e.g., "running" to "run").
- **Tokenization**: Tokenization: The text was processed into smaller units (tokens) using the **BERT tokenizer**
- **Padding**: Text sequences were padded to a fixed length (512 tokens) to ensure consistency in input size.
- **Data Splitting**: The dataset was split into training and test sets, where 80% of the data was used for training and 20% for testing.

2.2 Model Architecture

The LSTM-based architecture for sentiment analysis consists of:

- **Embedding Layer**: Converts input tokens into dense vectors representing their semantic meaning.
- **Bidirectional LSTM Layer**: Processes the sequence in both directions to capture contextual information from the entire sequence.
- **Fully Connected (Dense) Layers**: Learn higher-level representations and map the LSTM output to sentiment labels.

- **Dropout**: Applied to reduce overfitting by randomly dropping neurons during training.
- **Sigmoid Output Layer**: Outputs a probability for the positive class (0 for negative, 1 for positive sentiment).

2.3 Training

The model was trained with:

- **Optimizer**: The **Adam optimizer** was used for efficient training.
- **Loss Function**: **Binary Cross-Entropy Loss** was used for binary classification (positive/negative sentiment).
- **Batch Size**: A batch size of 1000 was used during training to improve convergence speed and make better use of GPU resources.

2.4 Evaluation

The model was evaluated on the test set, and performance metrics such as **precision**, **recall**, **F1-score**, and **accuracy** were calculated. The **classification report** was generated to assess the model's performance for both the positive and negative sentiment classes.

3. Results

The following **classification report** was obtained after evaluating the model on the test set:

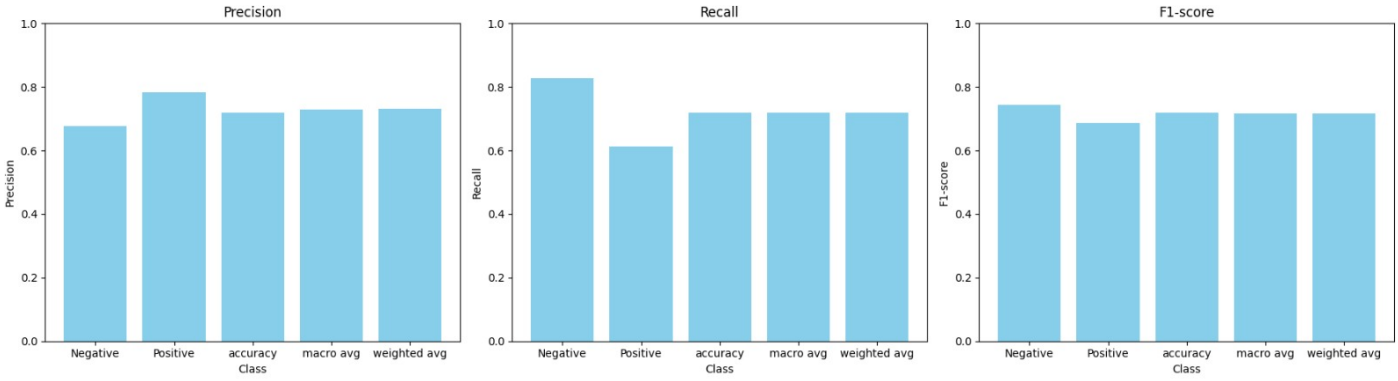
Metric	Negative	Positive	Macro Average
Precision	0.68	0.78	0.73
Recall	0.83	0.61	0.72
F1-score	0.74	0.69	0.72
Support	496	504	1000
Accuracy	0.72		

Precision: The negative class (0.68) precision was slightly lower than for the positive class (0.78), indicating that the model was slightly more accurate when predicting positive sentiment.

Recall: The recall for the negative class (0.83) was higher than for the positive class (0.61), suggesting that the model was better at identifying negative sentiments but struggled more with positive reviews.

F1-score: The F1-scores for both classes were reasonably balanced, with negative sentiment (0.74) performing better than positive sentiment (0.69).

Accuracy: The overall accuracy of the model was 72%, meaning it correctly classified 72% of the reviews in the test set.



Comparison to Related Work:
The LSTM-based model performed reasonably well, but more advanced models like BERT and transformers typically perform better, especially for the positive sentiment class, due to their ability to capture long-range dependencies and contextual information. The LSTM model is competitive but can be improved with hyperparameter tuning or by using more advanced models like BiLSTM or attention mechanisms.

What Worked Well:

- **Data Preprocessing**: Essential steps like cleaning, lemmatization, and tokenization prepared the text data effectively.
- **LSTM Architecture**: The LSTM network successfully captured sequential dependencies, providing a solid baseline.
- **Evaluation**: The model was evaluated comprehensively, revealing both strengths and weaknesses.

Challenges Faced:

- **Imbalanced Classes**: Though the dataset had an equal distribution of positive and negative reviews, the model performed better on negative sentiment.
- **Overfitting**: Despite regularization, overfitting occurred and could be addressed with hyperparameter tuning or advanced techniques.
- **Model Limitations**: The LSTM model is relatively simple compared to transformer-based models like BERT, which typically offer better performance.

Future Work:

- **Hyperparameter Tuning**: Optimizing hyperparameters like learning rate and batch size could enhance performance.
- **Advanced Architectures**: Using BERT or RoBERTa could improve results, particularly for positive sentiment classification.
- **Data Augmentation**: To reduce overfitting, data augmentation or a larger dataset could be explored.

Conclusion:
The LSTM-based sentiment analysis model achieved 72% accuracy, performing well on negative sentiment but struggling with positive sentiment. While effective, the model could benefit from the use of more advanced models and further optimization techniques.