NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES
(KARACHI CAMPUS)
Department of Computer Science
SPRING 2025

Sentiment Analysis with Deep Learning and Transformers

PROJECT REPORT

Group Members:

ZOHAIB SAQIB	21K-3215
MUHAMMAD	21K-3192
ONAIS ALI SHAH	21K-4691

Objective:

The primary objective of this project is to develop and compare deep learning models for sentiment classification on Twitter data. The models aim to classify tweets into two sentiment categories: positive and negative. We evaluate traditional architectures such as Bidirectional LSTM, alongside state-of-the-art transformer-based models like DistilBERT and DistilBART-CNN-12-6, to determine the most effective approach for text-based sentiment analysis.

Problem Statement:

With the explosion of user-generated content on social media platforms, understanding public sentiment at scale has become increasingly valuable for businesses, researchers, and policymakers. Tweets, in particular, are short and often unstructured, making them challenging to analyze using conventional natural language processing methods. This project addresses the challenge of accurately classifying tweet sentiments using modern deep learning approaches.

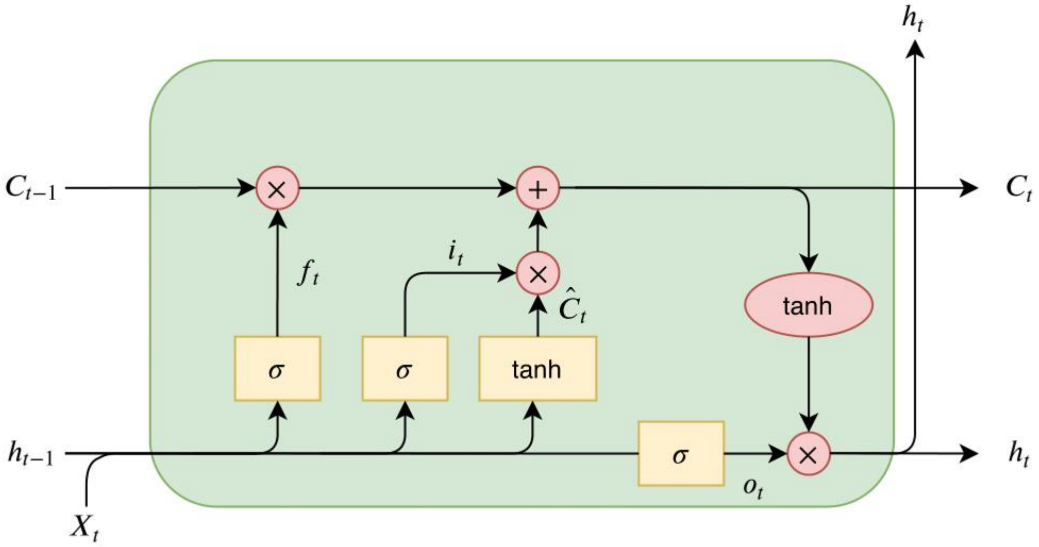
Methodology:

1. **Dataset:** We used the Sentiment140 dataset, which consists of 1.6 million tweets labeled for sentiments: positive and negative. For LSTM, we used the entire 1.6 million tweets data, for BERT we used 800,000 tweets data, and for BART we used 250,000 tweets data because of the computation and time constraints.\

2. **Data Preprocessing:** For pre-processing the data for the models, a simple data cleaning step took place first, which removed usernames, URLs, HTML entities, and unnecessary whitespace from the tweets using RegEx. Data was split into a ratio of 80/10/10. As the data was huge, we felt that a 70/15/15 split would result in larger-than-necessary validation and test splits. So, we tried to allocate the maximum data possible for the training split. During the cleaning step, the sentiment labels were remapped to a 2-class format, the negative sentiment was mapped to 0 label, and the positive sentiment was mapped to 1 label.

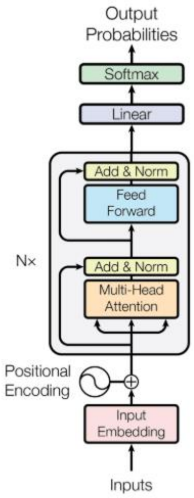
3. **Tokenization:** Tokenization was performed differently depending on the model. For LSTM,Tensorflow's Tokenizer and pad_sequences were used. For BERT and BART, Hugging Face's AutoTokenizer and BartTokenizer were used respectively, with padding and truncation to a maximum sequence length.

4. **Model Architectures:**
   - **BiLSTM**



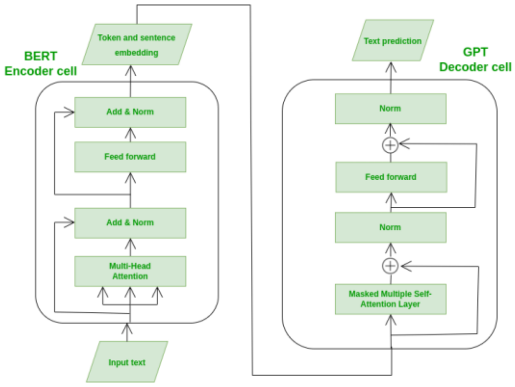
For the BiLSTM model, we trained our own embedding layer. Since, we kept the top 8000 most frequent words, our vocab size became 8000, and for each word, we generated a 100 dim embedding. The embedding layer was followed by a Bidirectional LSTM layer with 512 units. We also added Dropout to our model, as during testing we found it was overfitting. The dropout probability was set to 50%. Lastly, a Dense layer with sigmoid activation function for classification.

- **DistilBERT**



We used the pre-trained, distilled version of BERT, as it provided better computation efficiency with little loss of performance. Then, we finetuned it on our dataset with an additional classification head to perform binary classification.

- **DistilBART-CNN-12-6**



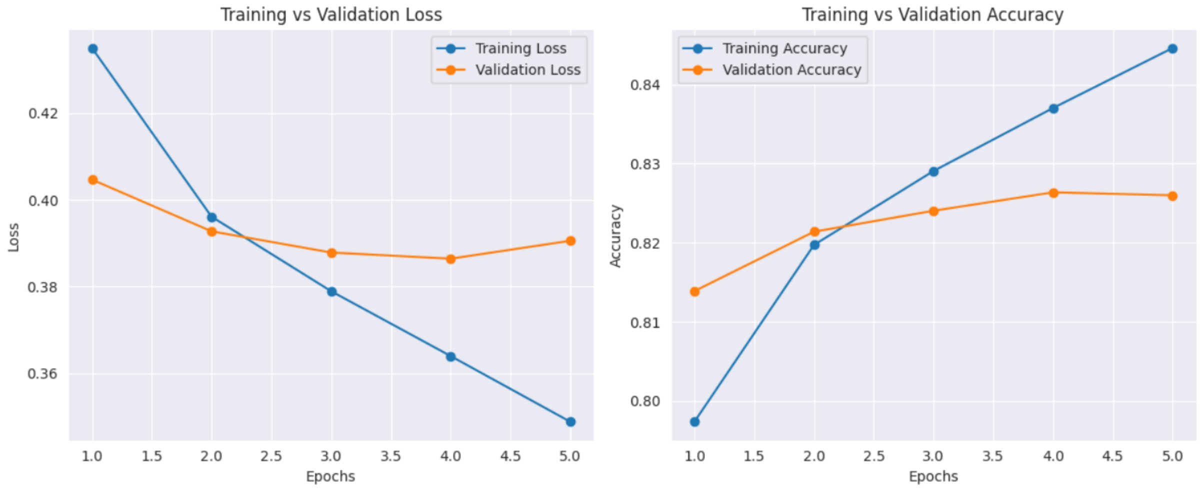
For BART, we also used a popular distilled version of BART named distilbart-cnn-12-6. For finetuning, we used PyTorch to setup the training and evaluation loop from scratch, as it provided better computational efficiency and more control.

5. **Training Strategy:** For batch sizes, the batch size for LSTM was 256, the batch size for BERT was 64, and the batch size for BART was 32. All the models used binary crossentropy as their loss function, as it was a binary classification task. The learning rates for BERT and BART were set manually, and were 1e-3 and 2e-5 respectively. Validation sets were used to evaluate the model's performance after each epoch during training to give an idea of how much the models are improving. Final evaluation was done on the test set.

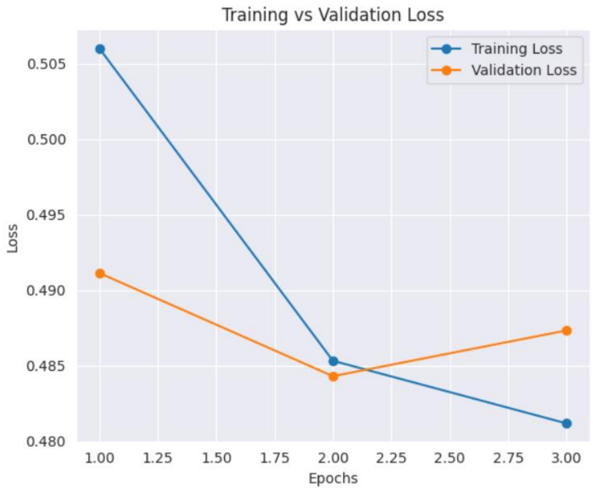
Results:

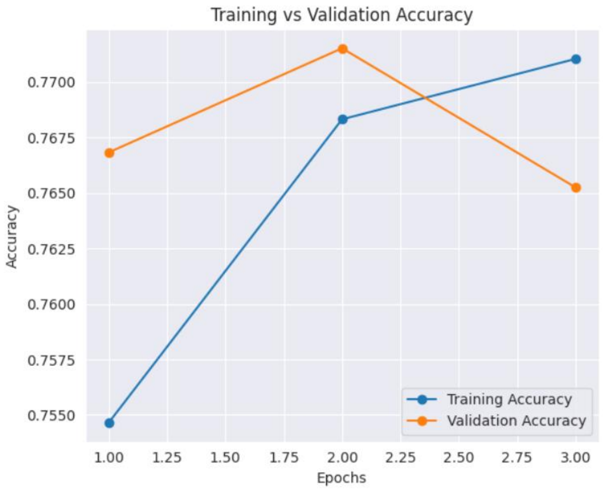
The performance of three models were evaluated on a sentiment classification task using the Sentiment140 dataset. LSTM model was trained for 5 epochs, and the rest of the models were trained for 3 epochs. They were evaluated using training loss, validation loss, validation accuracy, and final test accuracy.

1. **LSTM:**
   The LSTM model showed steady improvement over training epochs. Training loss decreases gradually from 0.4718 to 0.3446, and validation accuracy increases from 81.39**%** to 82.64**%**. The model also generalizes well, with minimal overfitting.

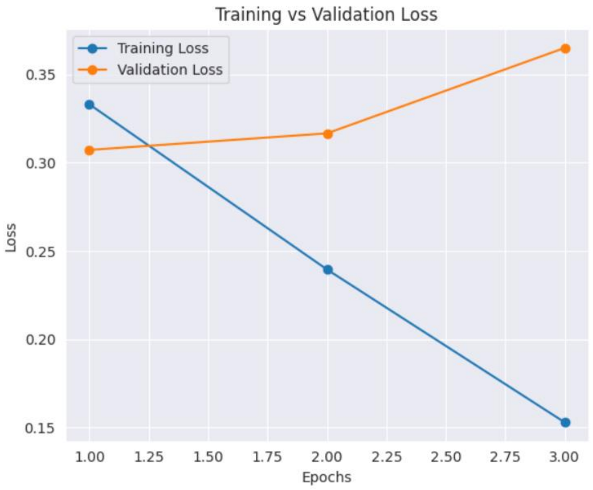


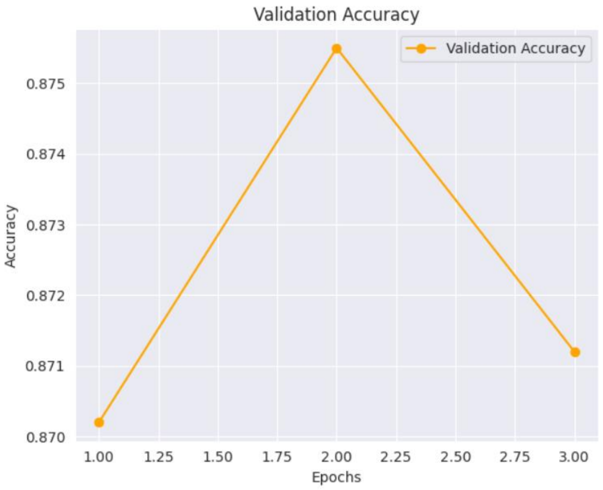
2. **BERT:**
Modest improvement between epoch 1 and 2, but validation accuracy drops in epoch 3 despite lower training loss, indicating some overfitting. Training loss drops from 0.5345 to 0.4824, but validation loss slightly increases in the final epoch. The model underperformed in the grand scheme of things, despite being a better architecture than LSTM.





3. **BART:**
Unsurprisingly, the strongest performer among all three. Training loss decreases sharply, but validation loss increases after epoch 1, suggesting slight overfitting. Despite that, validation accuracy remains consistently high.





Among the three models evaluated, BART demonstrated the highest validation accuracy, achieving 87.55%, and consistently outperformed both LSTM and BERT, making it the most effective model for this task. The LSTM model showed stable training and good generalization, making it a reliable baseline. In contrast, BERT underperformed.

References:

1. L. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer,
   "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension,"
   in *Proc. 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020, pp. 7871–7880.
2. V. Sanh, L. Debut, J. Chaumond, and T. Wolf,
   "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter,"
   arXiv preprint arXiv:1910.01108, 2019.
3. K. Hermann, T. Kočiský, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom,
   "Teaching Machines to Read and Comprehend,"
   in *Advances in Neural Information Processing Systems (NeurIPS)*, 2015, vol. 28.