Advanced Deep Learning

AIGC 5500

Final Project

Comparative Analysis of Deep Learning Models for Sentiment Analysis on Yelp Reviews using **LSTM** and **DistillBert**

Submitted To:
Hossein Pourmodheji

Submitted By:

Balwinder Kaur - N01716139	DivyabharathiKanapan - N01691451
Jiya John - N01706490	Pooja Hosamane - N01603746

Introduction

Overview

This project focuses on the sentiment classification of Yelp restaurant reviews into three categories: **Positive**, **Negative**, and **Neutral**.

Goals & Significance

Sentiment analysis plays a crucial role in understanding customer opinions and guiding business strategies in the restaurant industry. To achieve this, two deep learning models were implemented and evaluated:

- A custom **LSTM**-based classifier
- A fine-tuned **DistilBERT** transformer model

The primary goal was to assess and compare the performance, interpretability, and generalization capabilities of these models on real-world review data, providing insights into the most effective approach for handling nuanced textual feedback.

Dataset Description & Preprocessing

Dataset

The dataset used for this project is Yelp Review Full from HuggingFace Datasets, which contains over 650,000 Yelp restaurant reviews labeled with star ratings from 0 to 4. To adapt the dataset for multi-class sentiment classification, we manually relabeled the reviews into three sentiment categories:

- 0–1 stars → **Negative**
- 2 stars → **Neutral**
- 3–4 stars → **Positive**

This relabeling was done to better reflect sentiment polarity, capturing a neutral midpoint while balancing the class distribution.

Preprocessing Overview

Different preprocessing pipelines were used for the LSTM and DistilBERT models to suit their architectural requirements.

LSTM Preprocessing

- **Text cleaning**: Removed punctuation, stopwords, and special characters to reduce noise.
- **Manual tokenization**: Used NLTK to tokenize and maintain control over the vocabulary.
- **Vocabulary construction**: Built from training data to convert words into integer IDs.
- **Padding**: All sequences were padded to a fixed length of 128 tokens to ensure consistent input dimensions for the LSTM.

Rationale: LSTM networks are sensitive to sequence lengths and word order, so controlled tokenization and consistent input shapes help stabilize training.

DistillBert Preprocessing

- Used **DistilBertTokenizerFast** from HuggingFace, which automatically lowercases and tokenizes input.
- Encoded text using token IDs and attention masks.
- Extracted the **[CLS]** token representation for classification, as it's designed to capture the overall context.

Rationale: Transformer models like DistilBERT are pre-trained on raw text using specific tokenization schemes, so minimal preprocessing ensures compatibility and preserves contextual integrity.

Model Description & Architecture

LSTM Classifier

The LSTM model architecture consists of:

- An **embedding layer** (dimension = 100) to learn dense vector representations of tokens.
- A **bidirectional LSTM** with 128 hidden units to capture both forward and backward contextual dependencies in the text.
- A **fully connected output layer** producing logits for three sentiment classes.
- **Dropout** of 0.5 was applied to reduce overfitting.
- **Optimizer**: Adam

DistilBERT Classifier

The transformer-based model uses the pretrained **distilbert-base-uncased**, with:

- A **classification head** added to the **[CLS]** token output.
- **Dropout** of 0.3
- **Optimizer**: Selected via Optuna, based on best validation performance (e.g., AdamW)

Hyperparameter Tuning

LSTM tuning was performed manually, experimenting with various configurations to optimize performance. Key observations included:

Hyperparameter	**Observations**
Embedding Dim	An embedding size of 100 provided better accuracy than smaller sizes by capturing richer semantic information
Hidden Dim	128 hidden units offered a strong balance of expressiveness and generalization. Larger sizes showed no clear gain and risked overfitting.
Number of Layers	A single-layer LSTM performed best, increasing layers did not yield improvement and sometimes hurt generalization.
Dropout	A dropout rate of 0.5 was optimal. It effectively prevented overfitting while maintaining high model accuracy.

Best LSTM Configuration

- Embedding Dim: 100
- Hidden Units: 128
- Layers: 1
- Dropout: 0.5
  Evaluation was based on validation loss with early stopping across multiple runs.

DistilBERT Tuning (via Optuna)

A limited Optuna trial budget was used to tune key parameters:

Hyperparameter	**Observations**
Learning Rate	Around 3e-5 provided stable, fast convergence. Higher rates caused divergence.
Dropout	A moderate value (~0.2) yielded best generalization; higher dropout degraded learning.
Weight Decay	Slight weight decay (~0.008) improved generalization on validation/test sets.
Optimizer	AdamW outperformed Adam, especially in early convergence and validation accuracy.
Batch Size	A batch size of 16 helped stabilize gradients; larger values reduced performance due to GPU memory limits and noise.

Best DistilBERT Parameters (Selected based on Validation accuracy**)**

- Optimizer: AdamW
- Learning Rate: 4.45e-5
- Dropout: 0.127
- Weight Decay: 0.00845
- Batch Size: 16

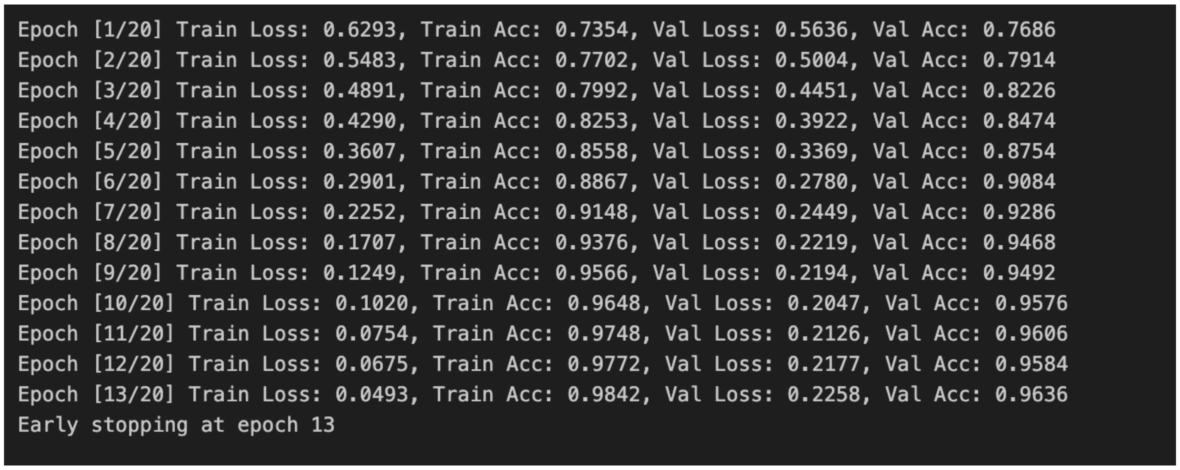
Results & Findings

Training Details

LSTM Training

- **Epochs**: 20
- **Early Stopped at** : Epoch 13
- **Final Training Accuracy:** 98.42%
- **Final Training Loss**: 0.0493
- **Final Validation Accuracy**: 96.36%
- **Final Validation Loss**: 0.2258
- **Final Test Accuracy**: 96.51%
- **Final Test Loss**: 0.2184

LSTM Training Progress:



```
Epoch [1/20] Train Loss: 0.6293, Train Acc: 0.7354, Val Loss: 0.5636, Val Acc: 0.7686
Epoch [2/20] Train Loss: 0.5483, Train Acc: 0.7702, Val Loss: 0.5004, Val Acc: 0.7914
Epoch [3/20] Train Loss: 0.4891, Train Acc: 0.7992, Val Loss: 0.4451, Val Acc: 0.8226
Epoch [4/20] Train Loss: 0.4290, Train Acc: 0.8253, Val Loss: 0.3922, Val Acc: 0.8474
Epoch [5/20] Train Loss: 0.3607, Train Acc: 0.8558, Val Loss: 0.3369, Val Acc: 0.8754
Epoch [6/20] Train Loss: 0.2901, Train Acc: 0.8867, Val Loss: 0.2780, Val Acc: 0.9084
Epoch [7/20] Train Loss: 0.2252, Train Acc: 0.9148, Val Loss: 0.2449, Val Acc: 0.9286
Epoch [8/20] Train Loss: 0.1707, Train Acc: 0.9376, Val Loss: 0.2219, Val Acc: 0.9468
Epoch [9/20] Train Loss: 0.1249, Train Acc: 0.9566, Val Loss: 0.2194, Val Acc: 0.9492
Epoch [10/20] Train Loss: 0.1020, Train Acc: 0.9648, Val Loss: 0.2047, Val Acc: 0.9576
Epoch [11/20] Train Loss: 0.0754, Train Acc: 0.9748, Val Loss: 0.2126, Val Acc: 0.9606
Epoch [12/20] Train Loss: 0.0675, Train Acc: 0.9772, Val Loss: 0.2177, Val Acc: 0.9584
Epoch [13/20] Train Loss: 0.0493, Train Acc: 0.9842, Val Loss: 0.2258, Val Acc: 0.9636
Early stopping at epoch 13
```

Key Observations:

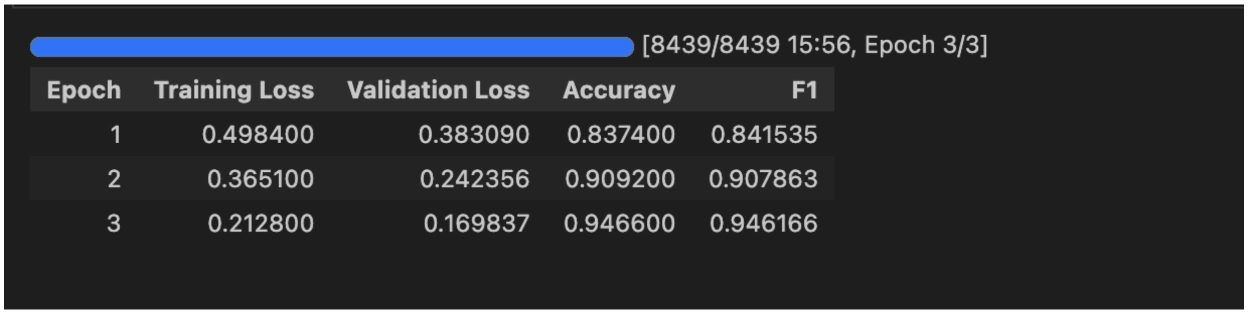
- Steady convergence over 13 epochs with early stopping
- Final Test accuracy (96.51%) and low test loss (0.2184) indicate strong generalization performance
- No signs of overfitting — training and validation accuracy improved constantly.

DistilBert Training

- **Epochs**: 3
- **Final Training Accuracy:** 96.29%
- **Final Training Loss**: 0.2128

- **Final Validation Accuracy**: 94.66%
- **Final Validation Loss**: 0.1698
- **Final Test Accuracy**: 80.74%
- **Final Test Loss**: 0.6395

DistilBERT Training Progress:



```
[8439/8439 15:56, Epoch 3/3]
Epoch  Training Loss  Validation Loss  Accuracy        F1
    1       0.498400         0.383090  0.837400  0.841535
    2       0.365100         0.242356  0.909200  0.907863
    3       0.212800         0.169837  0.946600  0.946166
```

Key Observations:

- Pre-trained weights enabled quick convergence
- Minimal overfitting, consistent accuracy across epochs

Evaluation

Confusion Matrix Analysis

Both models were evaluated using confusion matrices to understand class-wise performance. These matrices revealed insightful trends:

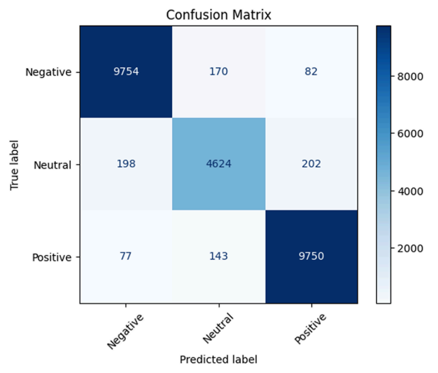
LSTM Model:

- Achieved high precision and recall for both Positive and Negative classes.
- Moderate confusion observed between Neutral and its neighboring classes.
- Tends to misclassify borderline-neutral reviews, which often share vocabulary with polarized opinions.

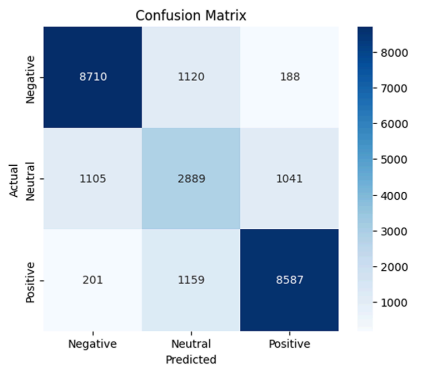
DistilBERT Model:

- Performed reasonably well on Positive and Negative classes.
- Showed significantly lower precision and recall for the Neutral class compared to LSTM.
- Struggled with subtle or overlapping sentiment, indicating a reliance on strongly polarized tokens for decision-making.

LSTM

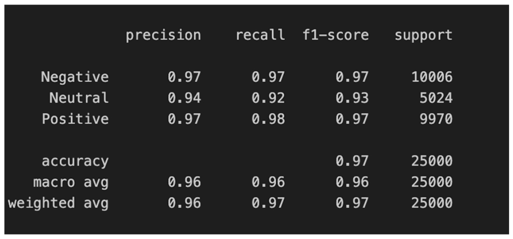


DistillBert



Classification Report

LSTM

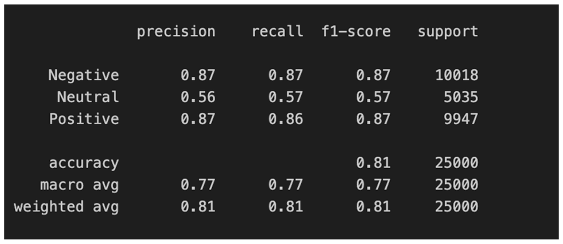


```
              precision    recall  f1-score   support

    Negative       0.97      0.97      0.97     10006
     Neutral       0.94      0.92      0.93      5024
    Positive       0.97      0.98      0.97      9970

    accuracy                           0.97     25000
   macro avg       0.96      0.96      0.96     25000
weighted avg       0.96      0.97      0.97     25000
```

DistillBert



```
              precision    recall  f1-score   support

    Negative       0.87      0.87      0.87     10018
     Neutral       0.56      0.57      0.57      5035
    Positive       0.87      0.86      0.87      9947

    accuracy                           0.81     25000
   macro avg       0.77      0.77      0.77     25000
weighted avg       0.81      0.81      0.81     25000
```

Evaluation Metrics (When run on **A100 GPU** with High-RAM for each epoch)

Model	**Test Accuracy**	**F1 Score**	**Test Loss**	**Training Time**
LSTM	96.51%	0.96	0.2184	~ less than 30 Sec
DistilBERT	80.74%	0.81	0.6395	~16 min

⚠️ ❗ Data Sampling & Resource Constraints:

Although the original Yelp dataset contains over 6.5 million reviews, we limited our training to a random sample of 50,000 examples due to computational constraints. This subset was further split into 90% training and 10% validation, with an additional 25,000 random samples used for testing.

We exhausted over 100 compute units during model training and had to repurchase additional credits. These constraints influenced our choice to use random sampling instead of sequential slicing, ensuring a more balanced and representative subset.

Analysis & Discussion

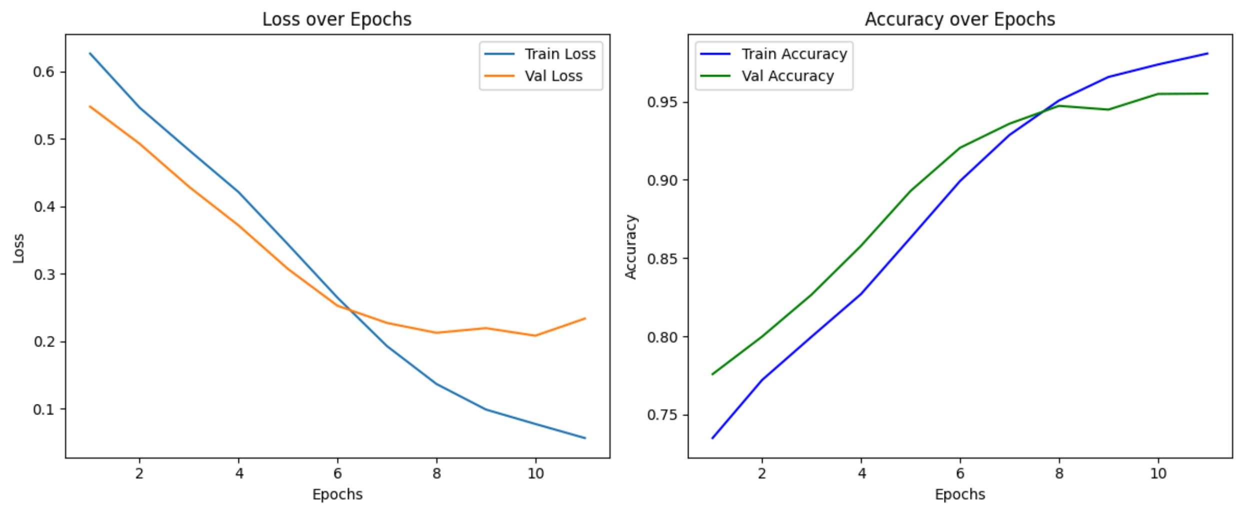
Models Comparison

Although DistilBERT achieved a higher validation accuracy (~94.66%), its test accuracy (80.74%) was lower than expected in real-world inference scenarios.

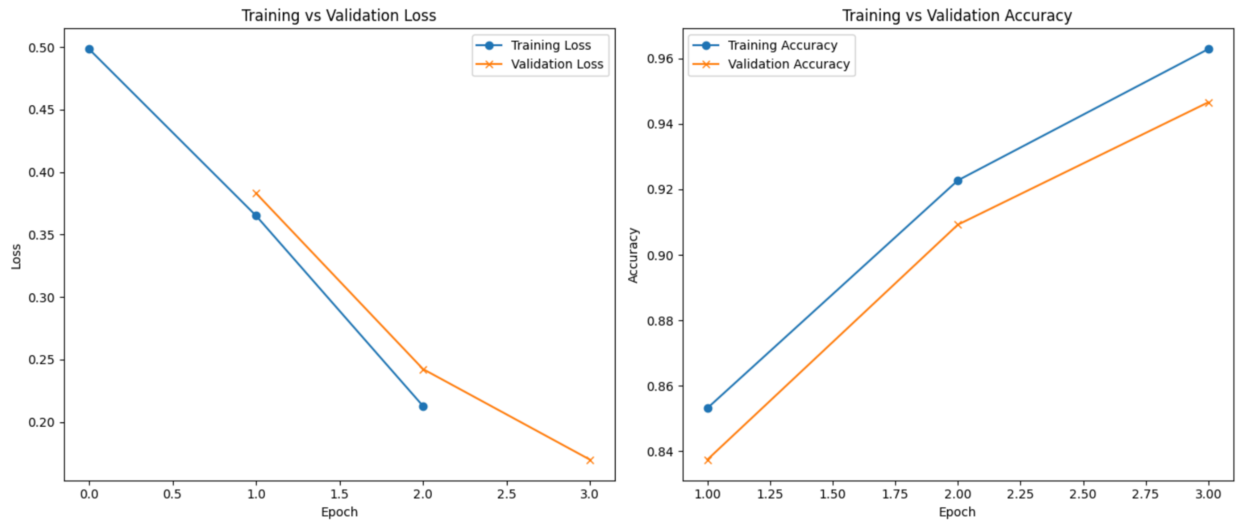
The LSTM model delivered robust and consistent performance, achieving 96.51% test accuracy and strong F1 scores across all sentiment classes.

Both models showed challenges in accurately classifying neutral reviews, often confusing them with positive or negative sentiment. However, DistilBERT performed better on the Neutral class, indicating a better grasp of subtle and context-dependent expressions.

LSTM Performance over Epochs



DistilBERT Performance over Epochs



Computational Efficiency

- **LSTM**: Fewer parameters, more resource-efficient, and suitable for low-compute environments.
- **DistilBERT**: Higher memory usage but benefits from GPU parallelization; better suited for high-performance systems.

Training Dynamics

- **LSTM**: Required more training epochs (early stopped at 13), but demonstrated stable and consistent improvement throughout training.
- **DistilBERT**: Converged rapidly in just 3 epochs, leveraging pretrained language representations.

Sensitivity

Attention Insights(LSTM)

The LSTM model, augmented with an attention mechanism, allowed focused learning on key sentiment-bearing words. This significantly improved interpretability and reduced over-reliance on input length.

Hyperparameter Sensitivity

- LSTM was highly sensitive to learning rate and dropout settings.
- DistilBERT showed strongest sensitivity to dropout rate and batch size, impacting its generalization.

Common Error Patterns

- Misclassification of ambiguous or mixed-sentiment reviews
- Neutral reviews skewing slightly positive or negative

Interpretability Analysis

LIME Evaluation (Used Same input for All 3 cases)

To ensure a fair comparison of interpretability across models, the same reviews were used as inputs for LIME analysis across different sentiment categories.

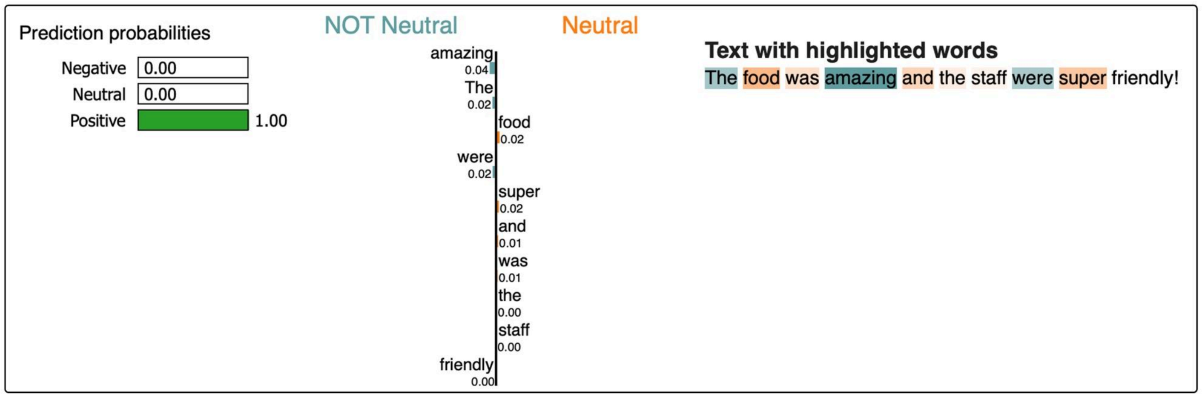
Positive Example

Input Text *: "The food was amazing and the staff were super friendly!"*

LSTM Prediction

- **Predicted**: Positive (1.00 probability)
- **Top Influential Tokens**: "amazing" (0.04), "The" (0.02), "food" (0.02), "were" (0.02)
- **Observation**: Attention mildly highlighted sentiment-related words like amazing, but weights were generally low across all tokens.

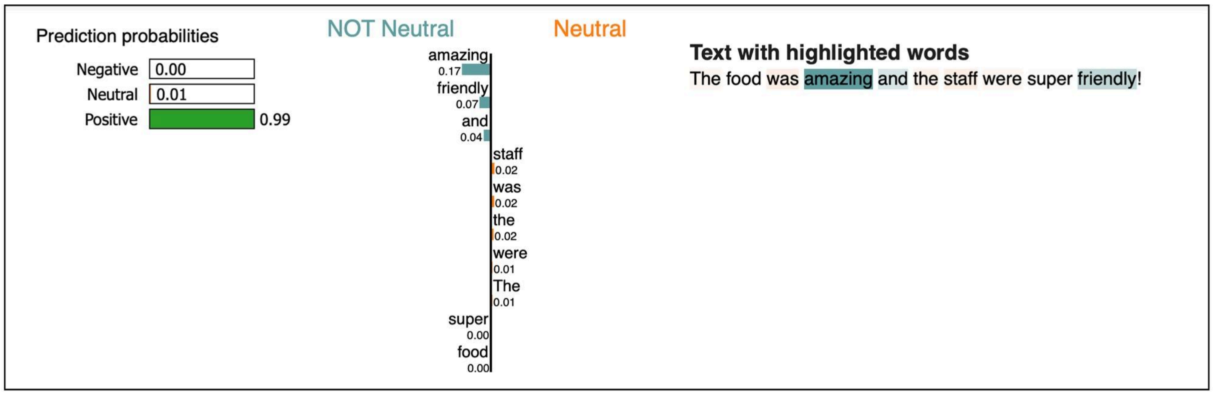
LSTM LIME Output



DistilBERT Prediction

- **Predicted**: Positive (0.99 probability)
- **Top Influential Tokens**: amazing" (0.17), "friendly" (0.07), "and" (0.04), "staff" (0.02)
- **Observation**: Attention mildly highlighted sentiment-related words like amazing, but weights were generally low across all tokens.

DistilBERT LIME Output



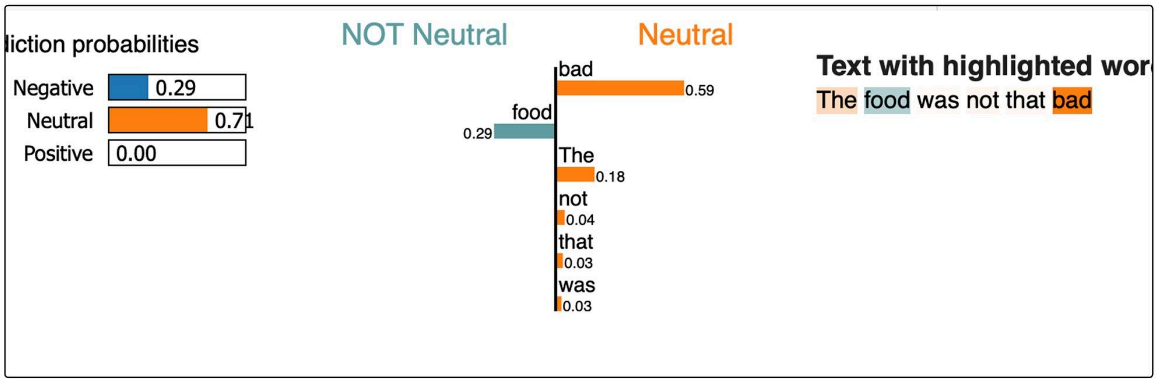
Neutral Example

Input Text *: "The food was not that bad."*

LSTM Prediction

- **Predicted**: Neutral (0.71 probability)
- **Top Influential Tokens**: "bad" (0.59), "food" (0.29), "The" (0.18)
- **Observation**: High attention on bad and food shows correct sensitivity to contrastive sentiment structure.

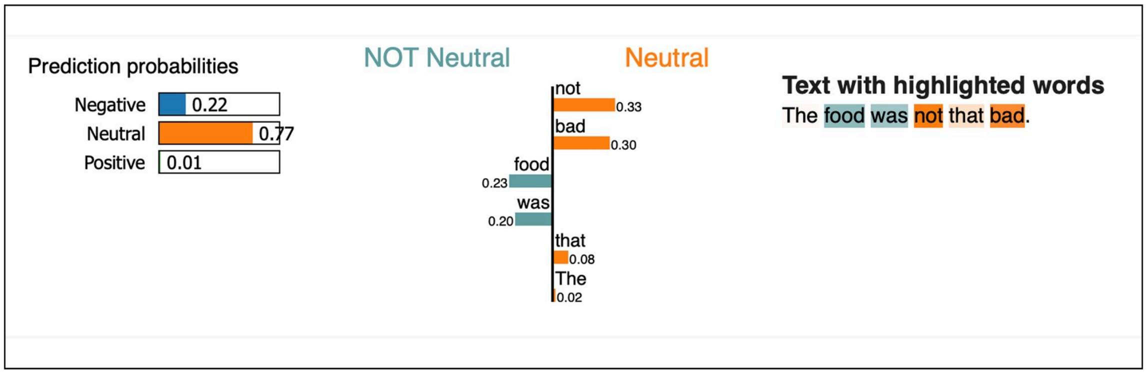
LSTM LIME Output



DistilBERT Prediction

- **Predicted**: Neutral (0.77 probability)
- **Top Influential Tokens**: "not" (0.33), "bad" (0.30), "food" (0.23), "was" (0.20)
- **Observation**: Clear emphasis on negation and sentiment-bearing tokens, demonstrating nuanced handling of neutral tone.

DistilBERT LIME Output



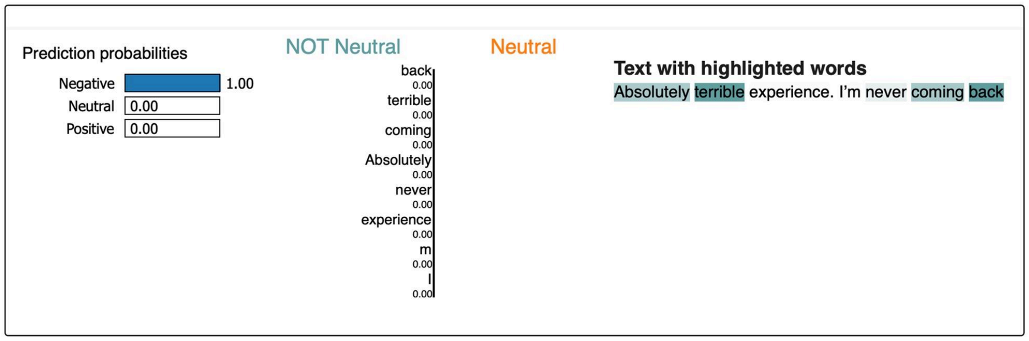
Negative Example

Input Text *: "Absolutely terrible experience. I'm never coming back"*

LSTM Prediction

- **Predicted**: Negative (1.00 probability)
- **Token Weights**: All attention weights were 0.00
- **Observation**: No meaningful token influence detected under current attention extraction setup, despite correct prediction.

LSTM LIME Output

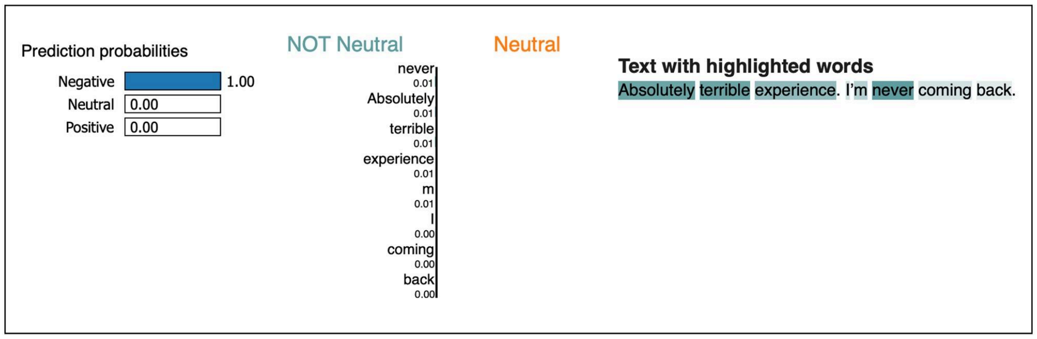


Text with highlighted words
Absolutely terrible experience. I'm never coming back

DistilBERT Prediction

- **Predicted**: Negative (1.00 probability)
- **Top Influential Tokens**: "never" (0.01), "Absolutely" (0.01), "terrible" (0.01), "experience" (0.01)
- **Observation**: Relatively flat distribution of attention across sentiment words, with the model still accurately recognizing strong negative sentiment.

DistilBERT LIME Output



Text with highlighted words
Absolutely terrible experience. I'm never coming back.

This shared evaluation approach helped reveal how each model interprets sentiment cues differently across strong, ambiguous, and neutral reviews. It shows how DistilBERT tends to rely on stronger lexical signals, while LSTM's attention behavior is more varied, especially in complex or ambiguous inputs.

Review Length Sensitivity

To assess how review length impacts model performance, the test set was divided into four categories based on word count:

Length Category	**Length Range**	**DistilBERT Accuracy**	**LSTM Accuracy**
Short	0–50 words	82.15%	96.64%
Medium	51–100 words	82.31%	96.38%
Long	101–200 words	81.11%	96.44%
Very Long	201–10000 words	76.48%	96.63%

Observations:

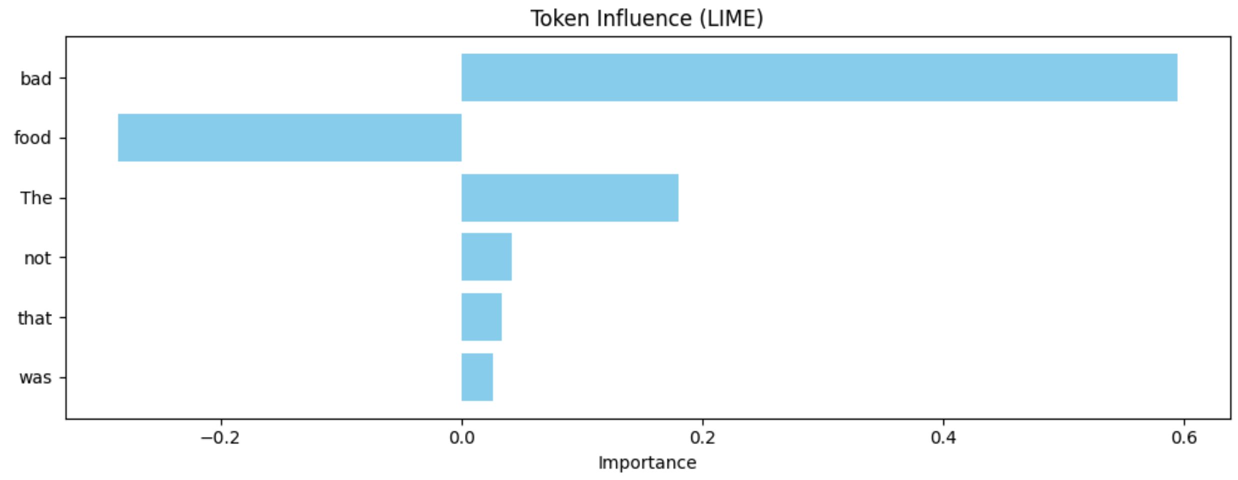
- LSTM maintained consistent high accuracy across all lengths, showing robustness to review size.
- DistilBERT performance declined significantly for longer texts, suggesting challenges in handling extended sequences without truncation or additional tuning.
- LSTM is better suited for length-diverse data without performance degradation.

Visualization & Reporting

To illustrate model interpretability, LIME was used to visualize how individual tokens contributed to sentiment classification. Below are the token importance plots for both DistilBERT and LSTM, using the same input review:

Input: *"The food was not that bad."*
Ground Truth: Neutral

LSTM - Token Influence

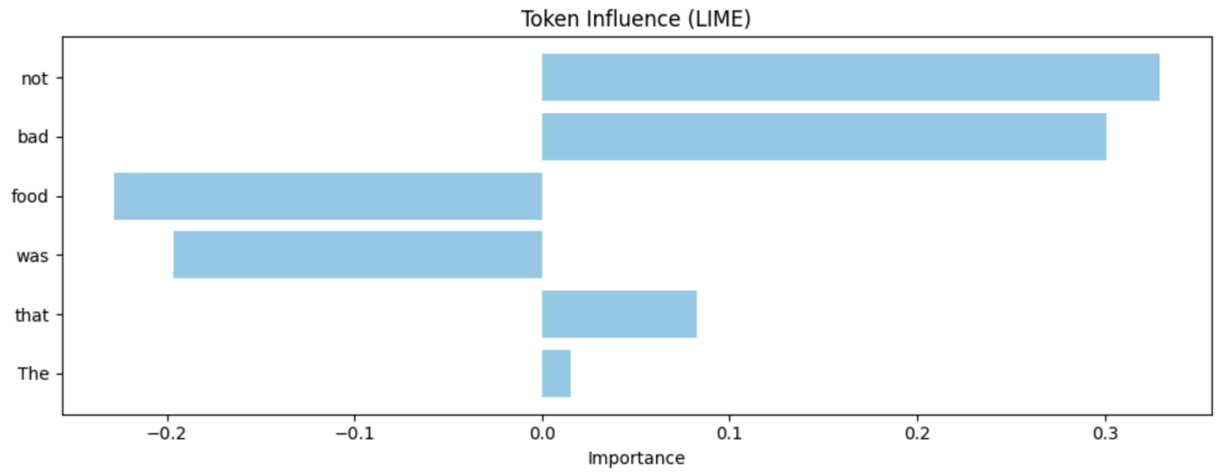


Strong token weights on:

- not, bad, food, with similar relative importance to DistilBERT

Slight negative influence from was and food, showing balanced understanding of sentiment context

DistilBERT - Token Influence



Strong token weights on:

- not (~0.33), bad (~0.30), food (~0.23)

Indicates nuanced handling of contrastive sentiment

Summary

- Both models correctly emphasized negation (not) and sentiment-shifting words (bad, food) in predicting neutral sentiment.
- While DistilBERT presented a slightly sharper focus, LSTM captured similar cues with a broader spread.
- These visualizations support the earlier vocabulary analysis and offer insight into how each model interprets subtle sentiment.

Conclusion

Summary of Key Findings

1. Both LSTM with attention and DistilBERT models delivered strong performance on Yelp sentiment classification tasks.
2. LSTM showed competitive results while being lightweight and interpretable, especially due to its attention layer.
3. DistilBERT achieved higher validation accuracy and converged quickly owing to its pre-trained language representation.
4. Both models struggled most with neutral sentiment detection, due to subtle contextual clues and overlapping language features.
5. Hyperparameter optimization had distinct impacts — learning rate critically influenced LSTM performance, while dropout was key for DistilBERT.
6. Evaluation on a common LIME input review showed different model perspectives on key sentiment phrases, highlighting their interpretability differences.

Final Assessment

- LSTM is efficient and interpretable, making it well-suited for resource-constrained deployments.
- DistilBERT is ideal for high-performance systems where accuracy and generalization matter most.
- Both models validated the value of deep learning in text sentiment classification.

Group Responsibility Breakdown

Task	**Contributor**
Data Preprocessing	Jiya John
LSTM Model Implementation	Pooja Hosamane Ramesh Babu
DistilBERT Fine-tuning	Divya Bharathi Kannappan
Evaluation and Analysis	Balwinder Kaur
Report Writing	Pooja Hosamane, Divya Bharathi

References

- HuggingFace Datasets & Transformers
- PyTorch & Transformers Documentation
- Ribeiro et al., "Why Should I Trust You?" (LIME)
- Yelp Dataset: https://huggingface.co/datasets/yelp_polarity
- https://github.com/LukeDitria/pytorch_tutorials/blob/main/section12_sequential/solutions/Pytorch6_LSTM_Text_Classification.ipynb