AIGC 5500 Final Project

Team Members :

Balwinder Kaur - N01716139

DivyabharathiKanapan-N01691451

Jiya John - N01706490

Pooja Hosamane - N01603746

Final Report: Yelp Review Sentiment Classification Using LSTM and DistilBERT

1. Introduction

This project focuses on sentiment classification of Yelp restaurant reviews into three categories: Positive, Negative, and Neutral. Two deep learning models were implemented and evaluated:

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

The objective was to compare the performance, interpretability, and generalization capabilities of these models on real-world review data.

2. Dataset and Preprocessing

Dataset

The dataset was loaded from HuggingFace's yelp_polarity dataset and manually relabeled to three classes:

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

Preprocessing

Preprocessing Overview

- **LSTM**: Applied text cleaning (removed punctuation, stopwords, and special characters), followed by manual tokenization and padding.
- **DistilBERT**: Passed raw text directly to DistilBertTokenizerFast, which handles tokenization and lowercasing internally (since distilbert-base-uncased was used).

LSTM Preprocessing

- Manual tokenization using nltk
- Vocabulary built from training data
- Padding to fixed length (128 tokens)

DistilBERT Preprocessing

- Used DistilBertTokenizerFast, which automatically lowercases input (since using distilbert-base-uncased)
- Encoded text with attention masks
- Used [CLS] token for classification

---

3. Model Architectures

LSTM Classifier

- Embedding layer (Embedding Dim = 100)
- Bidirectional LSTM (128 hidden units)
- Fully connected layer with 3 output logits
- Dropout = 0.5
- Optimizer: Adam

DistilBERT Classifier

- Pretrained distilbert-base-uncased
- Classification head added to [CLS] token
- Dropout = 0.3
- Optimizer: Selected via Optuna hyperparameter tuning (e.g., AdamW, RMSprop)

---

4. Hyperparameter Tuning

LSTM Hyperparameter Tuning

- Tuning performed manually using multiple combinations of architecture and regularization parameters.

Hyperparameter	Observed Impact
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 configuration:

- Embedding Dim: 100
- Hidden Dim: 128
- Layers: 1
- Dropout: 0.5

Evaluation was based on validation loss over multiple runs using early stopping.

DistilBERT Hyperparameter Tuning

- Used Optuna for tuning with limited trial budget
- Search space:
  - Learning rate: [1e-6, 1e-4] (log scale)
  - Optimizer: [AdamW, Adam]
  - Dropout: [0.1, 0.5]
  - Weight decay: [0.0, 0.01]
  - Batch size: 16, 32

Hyperparameter	Observed Impact
Learning	Crucial for convergence; values around 3e-5 balanced speed and stability. Too high 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 parameters selected (based on validation accuracy):

- Optimizer: adamw
- Learning rate: 4.45e-5
- Dropout: 0.127
- Weight Decay: 0.00845
- Batch Size: 16

---

5. Training Details

LSTM Training

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

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
```

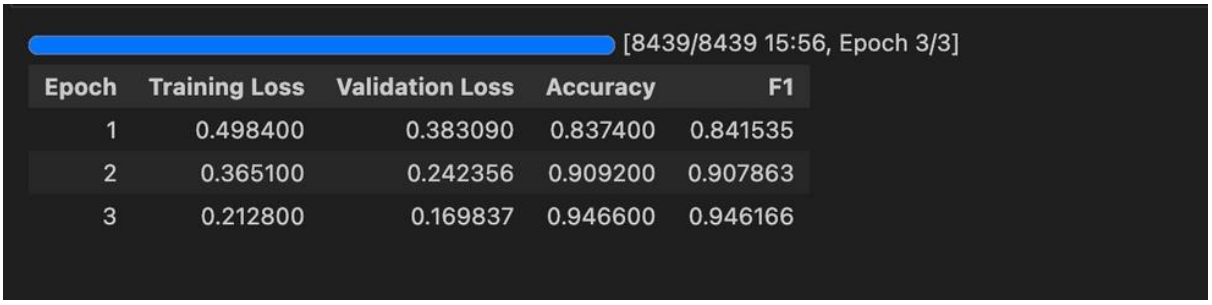
Key Observations:

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

DistilBERT Training

- **Epochs**: 3
- **Final Training Accuracy**: 96.29%
- **Final Validation Accuracy**: 94.66%
- **Final Training Loss**: 0.2128
- **Final Validation Loss**: 0.1698
- **Final Test Accuracy**: 80.74%
- **Final Test Loss**: 0.6395

Training Progress:



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

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

---

6. Evaluation Results

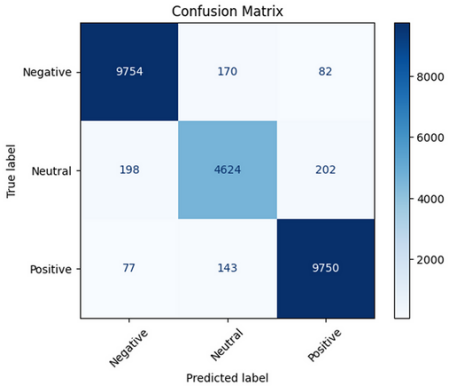
Confusion Matrix Analysis

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

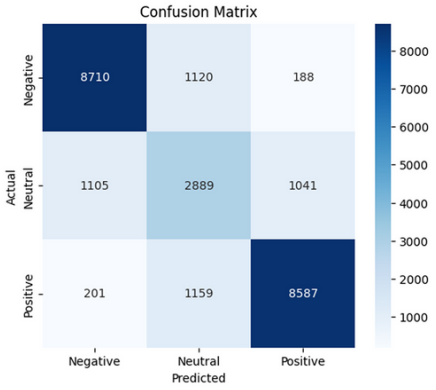
- **LSTM Model**:
  - Achieved high precision and recall for both Positive and Negative classes.
  - Moderate confusion observed between Neutral and its neighboring classes.
  - Tends to misclassify borderline-neutral reviews, which often share vocabulary with polarized opinions.
- **DistilBERT Model**: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

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

Model	Test Accuracy	F1 Score	Test Loss	Training Time (approx)
LSTM	96.51%	0.96	0.2184	~13 min
DistilBERT	80.74%	0.81	0.6395	~16 min

7. Analysis and Discussion

Model Comparison

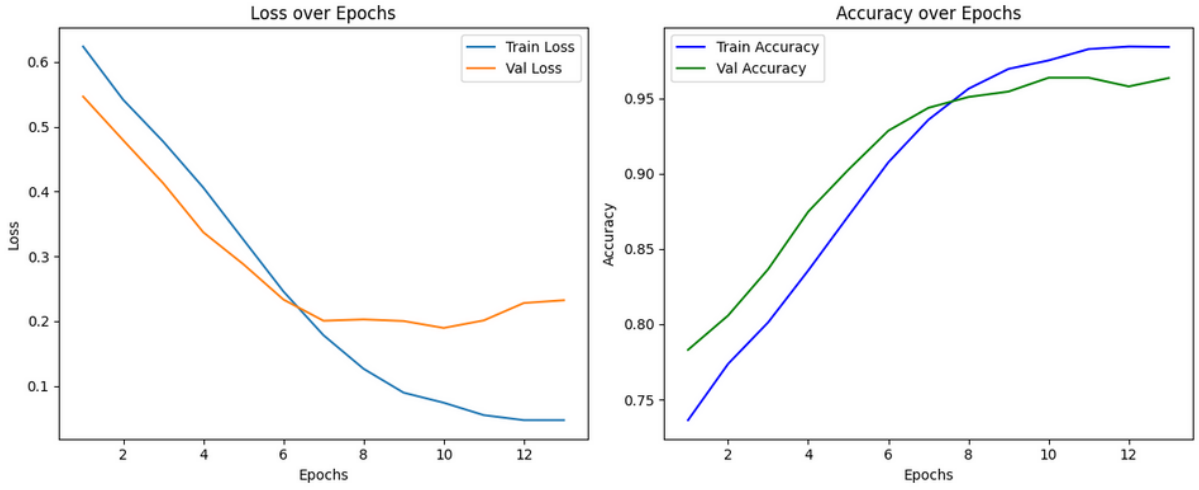
Accuracy and Classification Performance

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

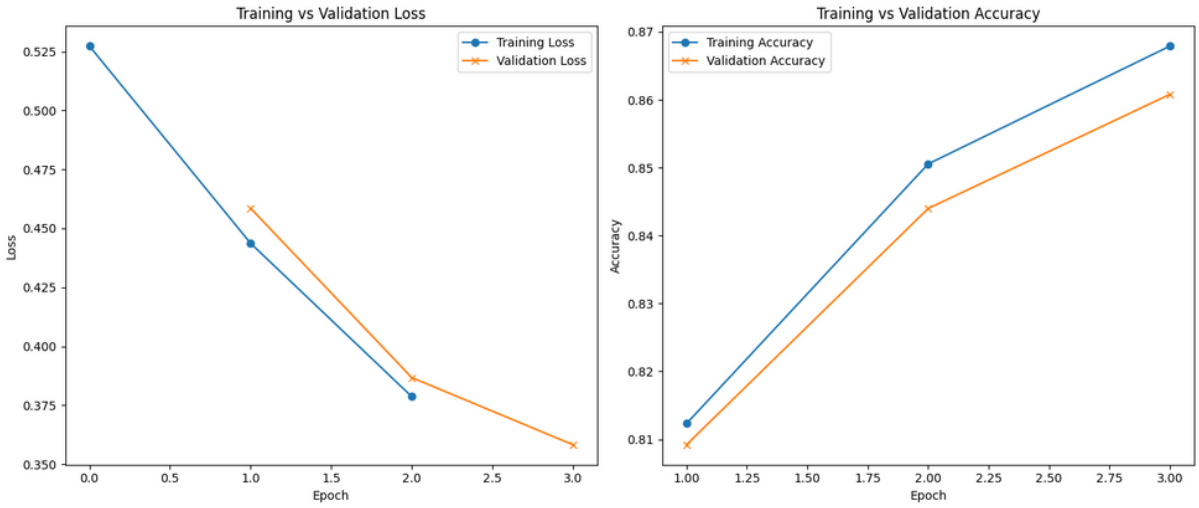
The LSTM model delivered robust and consistent performance, achieving 96.51% test accuracy and strong F1 scores across all sentiment classes.

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

- **LSTM Performance**



- **DistillBert Performance**



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

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

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

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

---

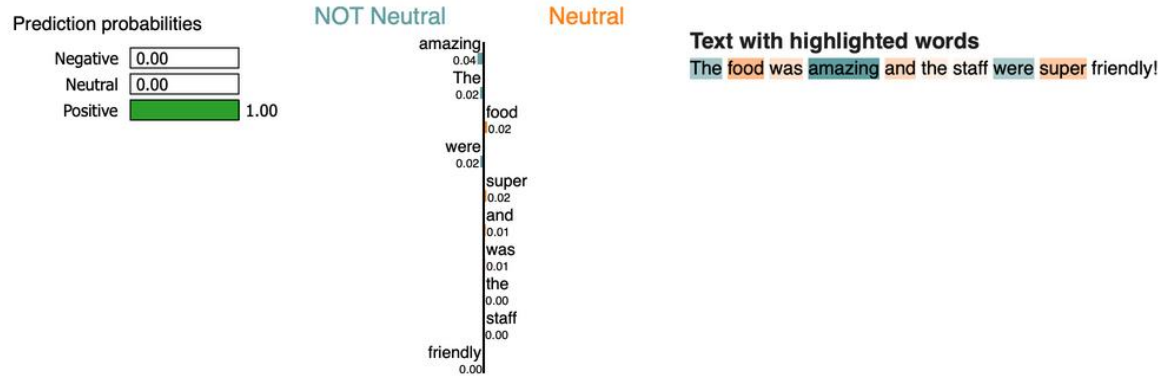
8. Interpretability and Analysis

LIME Evaluation (Same Input for All Cases)

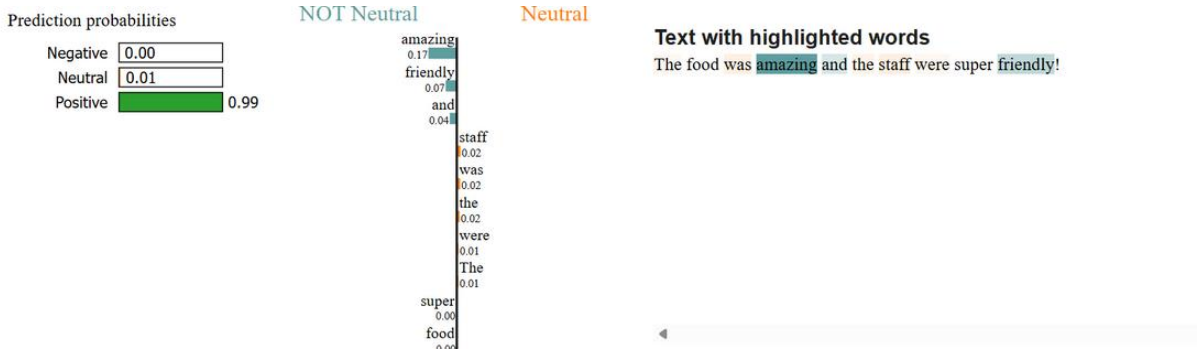
To ensure a fair comparison of interpretability across models, the same reviews were used as inputs for LIME analysis across different sentiment categories.

Positive Input Example - "The food was amazing and the staff were super friendly!"

- **LSTM LIME Output**:
  - **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.



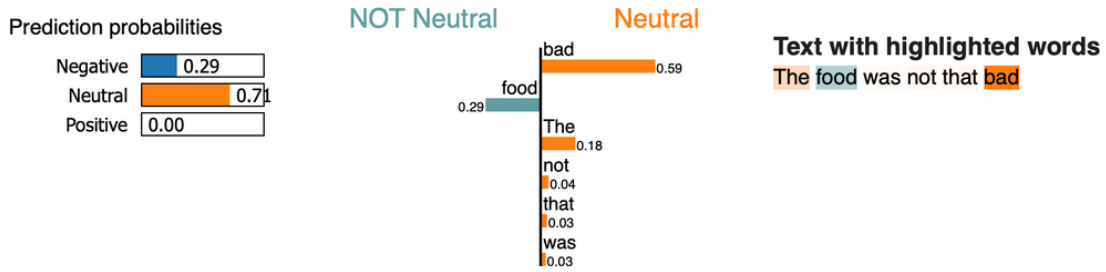
- **DistilBERT LIME Output**:
  - **Predicted**: Positive (0.99 probability)
  - **Top Influential Tokens**:"amazing" (0.17), "friendly" (0.07), "and" (0.04), "staff" (0.02)
  - **Observation**: Strong focus on key sentiment-rich words such as *amazing* and *friendly*, enabling interpretable decision-making.



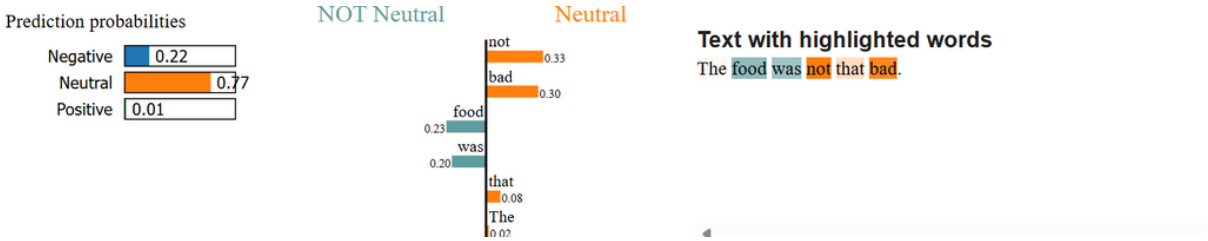
Neutral Input Example - "The food was not that bad."

- **LSTM LIME Output**:
  - **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.

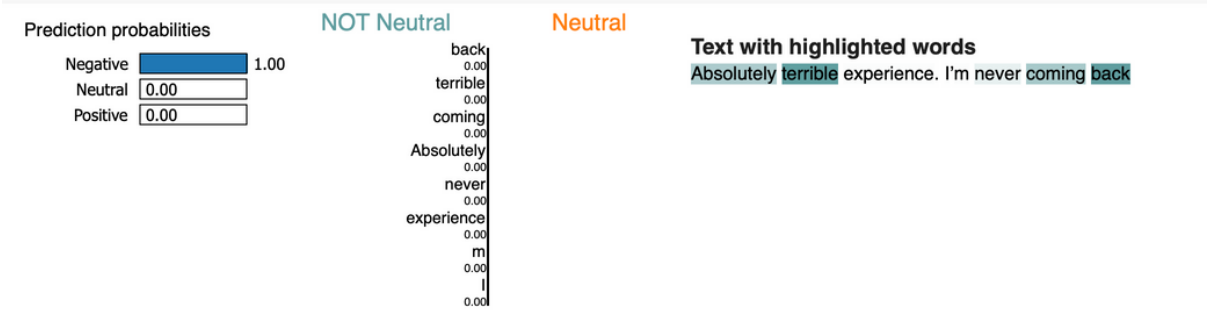


- **DistilBERT LIME Output**:
  - **Predicted**: Neutral (0.77 probability)
  - **Top Influential Tokens**:"not" (0.33), "bad" (0.30), "food" (0.23), "was" (0.20)
  - **Observation**: Clear emphasis on negation and sentiment-bearing tokens, demonstrating nuanced handling of neutral tone.

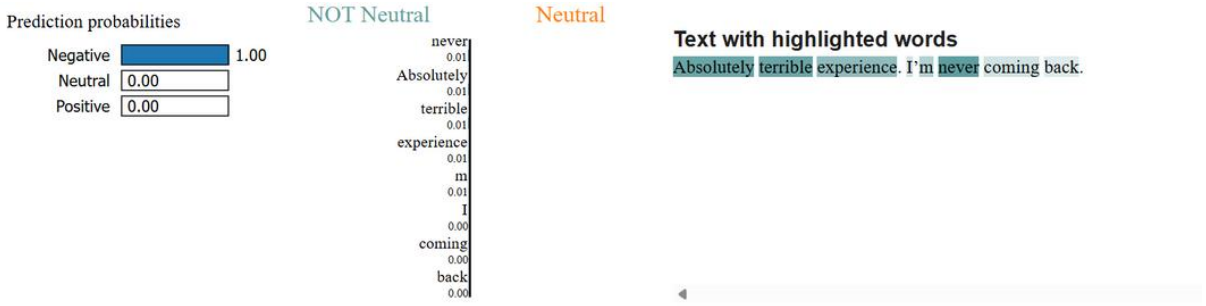


Negative Input Example - "Absolutely terrible experience. I'm never coming back"

- **LSTM LIME Output**:
  - **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.



- **DistilBERT LIME Output**:
  - **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.



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.

---

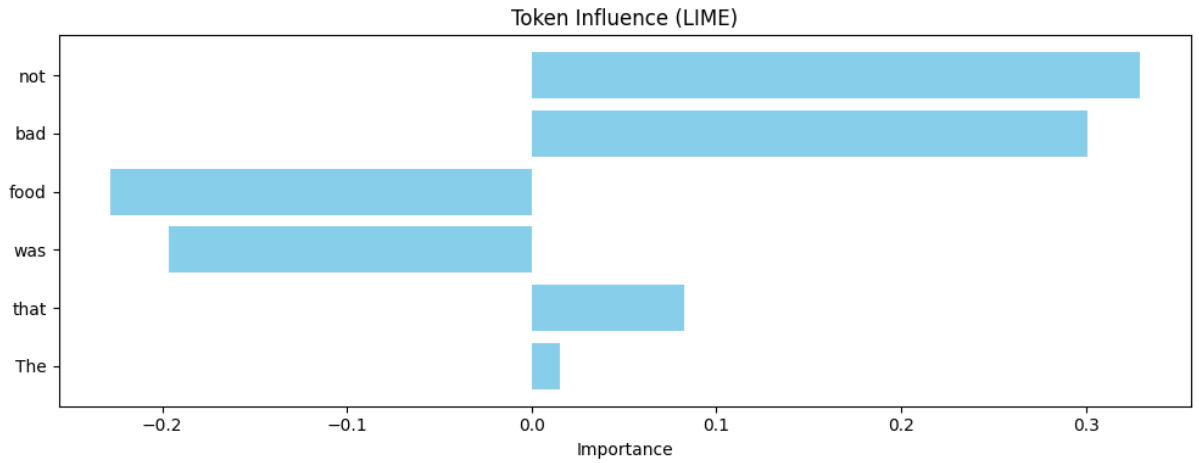
9. Visualization and Reporting

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."*

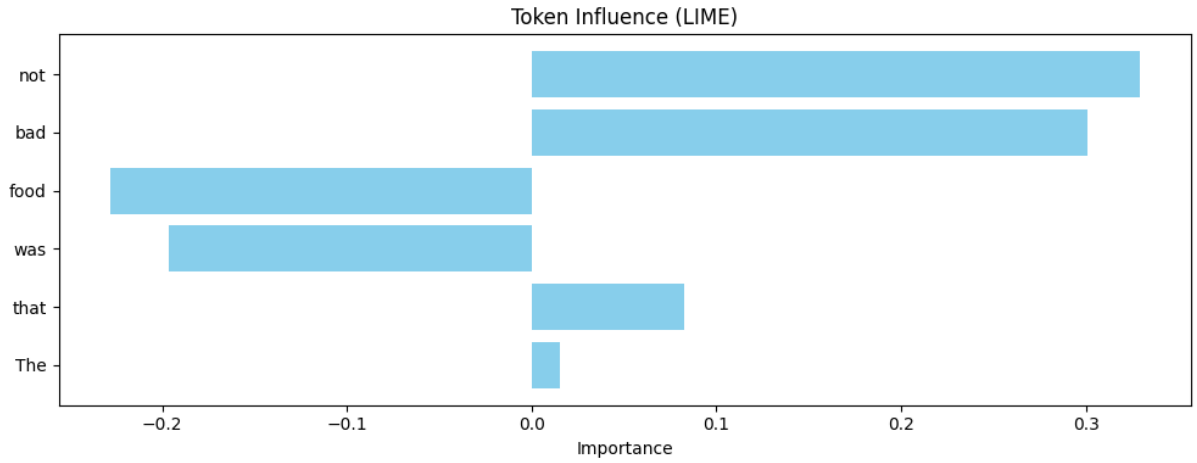
Ground Truth: Neutral

DistilBERT – Token Influence



- **Strong focus on**:
  - *not* (~0.33), *bad* (~0.30), *food* (~0.23)
- Indicates nuanced handling of contrastive sentiment

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

Summary

- Both models correctly emphasized negation (*not*) and sentiment-shifting words (*bad*, *food*) in predicting neutral sentiment.
- While DistilBERT presented 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.

---

10. 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 pretrained 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.

---

11. Contributions

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 Ramesh Babu and Divya Bharathi Kannappan

12. 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