Hussain Nathani - Individual Final Project Report

Detecting Sarcasm, Irony, and Figurative Language in Tweets using NLP

1. Introduction

The purpose of our group project was to develop an automated system capable of classifying tweets into four distinct categories of language use: **regular**, **sarcasm**, **irony**, and **figurative language**. Figurative language detection is a highly challenging NLP task because the meaning behind such expressions often differs significantly from the literal interpretation of the words. Unlike standard sentiment or topic classification, figurative language requires understanding of subtle pragmatic cues, speaker intent, and implicit contradictions. Given the informal, short, and noisy nature of Twitter data, the task becomes even more difficult.

Our project followed a complete machine learning pipeline involving several shared responsibilities: dataset exploration and cleaning, exploratory data analysis (EDA), multiple modeling approaches, model evaluation, error analysis, and deployment through a Streamlit web application. As a group, we implemented three model families that represent increasing levels of complexity: (1) TF–IDF + Logistic Regression, (2) a Bidirectional LSTM network, and (3) a fine-tuned DistilBERT transformer model. Each model helped us understand different aspects of the problem and the trade-off between classical NLP, deep learning, and transformer-based architectures.

My individual responsibilities within this larger project included performing a **comprehensive exploratory data analysis (EDA)**, developing and fine-tuning the **DistilBERT model**, evaluating its performance in depth, and integrating it properly into our Streamlit interface. This report documents the technical foundations behind my work, my implementation choices, the results I obtained, and my overall learning experience.

2. Description of My Individual Work (with Background Theory)

2.1 Exploratory Data Analysis (EDA)

The EDA process was essential for shaping our understanding of the dataset and for guiding several downstream choices, including preprocessing, tokenization length, and model selection. The dataset contained more than 89,000 tweets combined across training and test sets, each labeled as figurative, irony, sarcasm, or regular.

One of the first observations from EDA was the **short length of tweets**, with most containing fewer than 20 words. This insight justified using a relatively small token length (e.g., 64 tokens) during DistilBERT fine-tuning. A histogram of tweet lengths revealed a sharp peak in the 5–15 word range, confirming that the dataset primarily consists of concise, direct statements rather than long narratives.

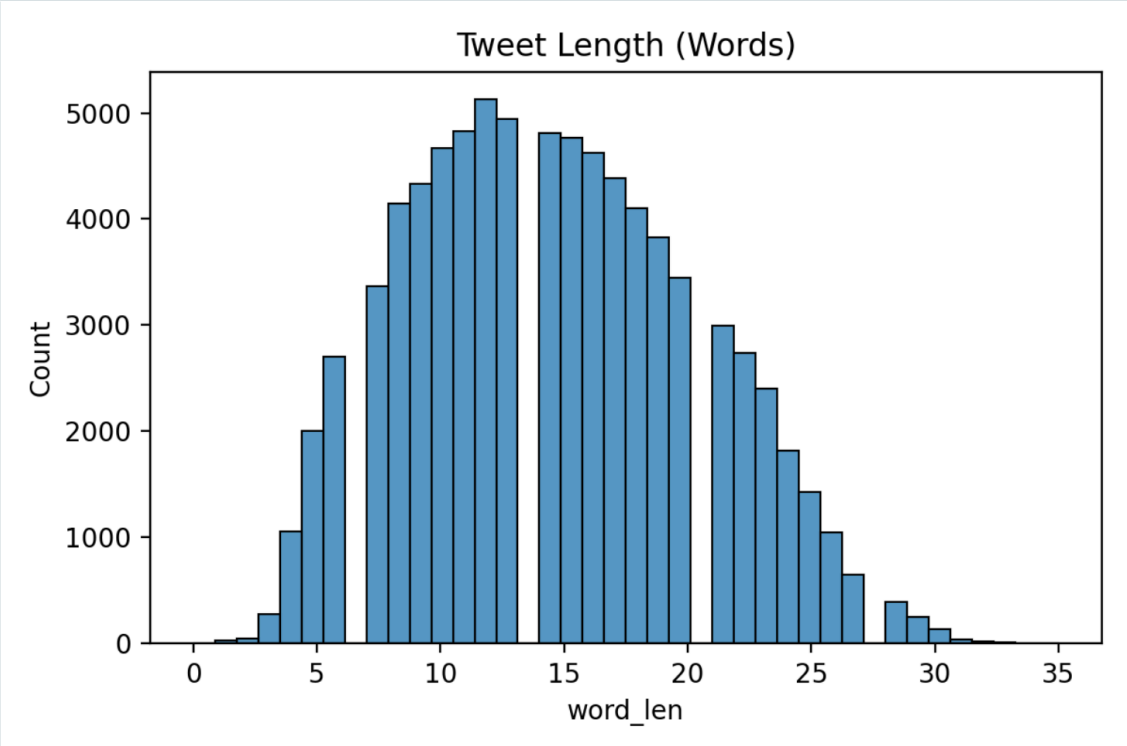

Fig 1: *Tweet length distribution histogram for the dataset.*

The next major insight came from studying **label distribution**. Although the dataset is not perfectly balanced, the counts across the four classes are reasonably close, meaning no aggressive resampling was needed. Figurative language, irony, and sarcasm appeared in nearly equal amounts, while regular tweets were slightly fewer. This balance helped ensure that model training was not biased toward any particular class.

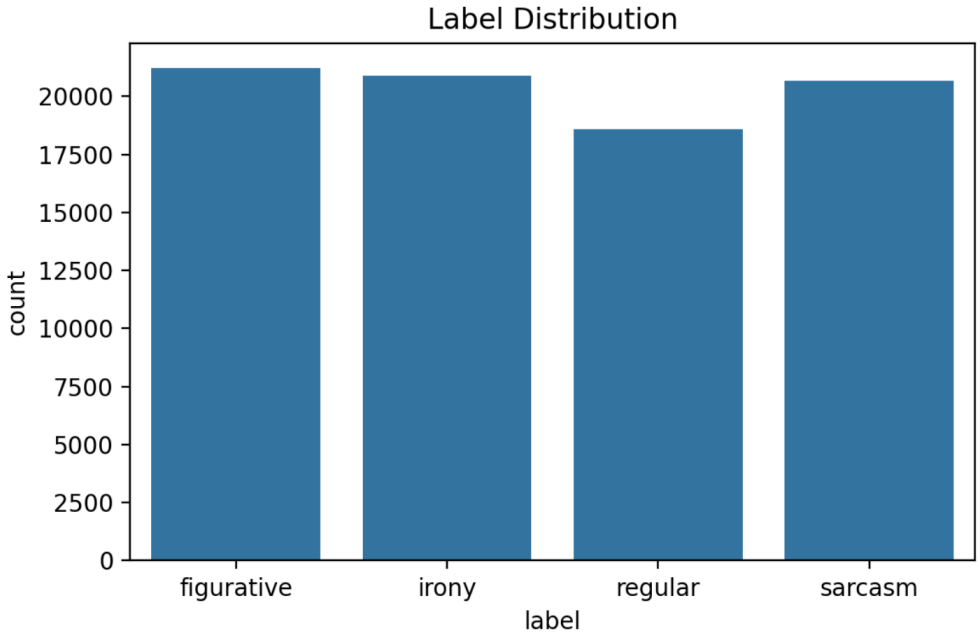

Fig 2: *Label distribution across the four tweet categories in the dataset.*

To better understand lexical tendencies, I generated **WordClouds** for each class. Sarcastic tweets commonly featured emotionally charged or exaggerated words, while ironic tweets included phrases expressing contradiction or subtle humor. Figurative tweets, however, did not exhibit a strong or unique vocabulary, indicating substantial overlap with sarcasm and irony. Regular tweets mostly contained factual or neutral terms. These lexical patterns revealed why figurative language is inherently difficult to classify: unlike sarcasm or irony, it does not consistently rely on specific keywords or markers.

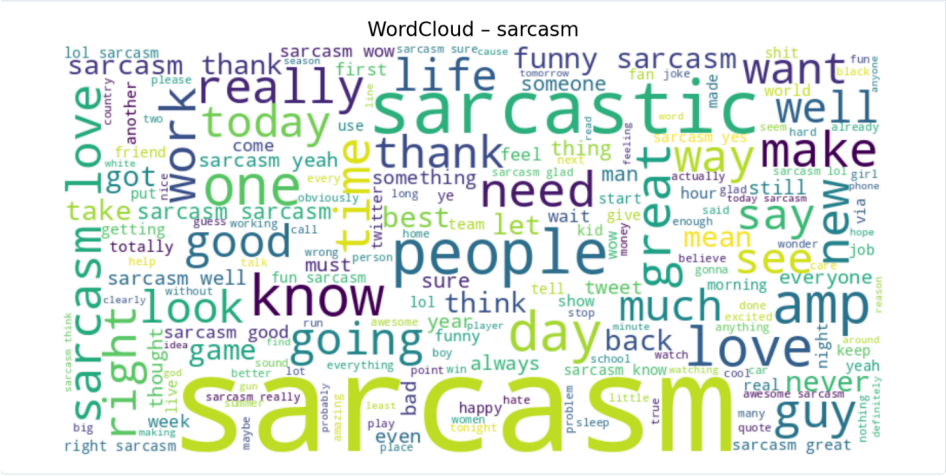

Fig 3: *Word cloud illustrating common lexical patterns in sarcastic tweets.*

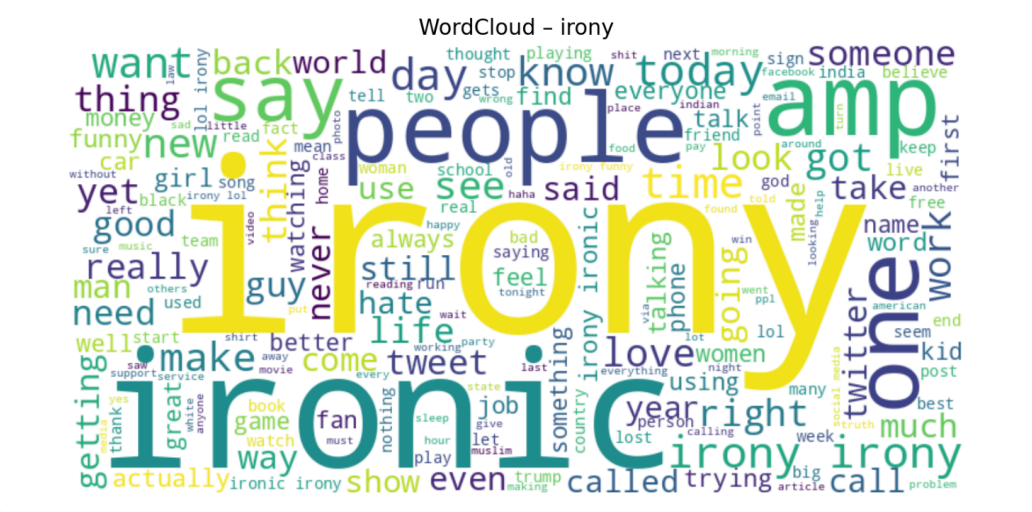

Fig 4: *Word cloud highlighting frequent terms found in ironic tweets.*

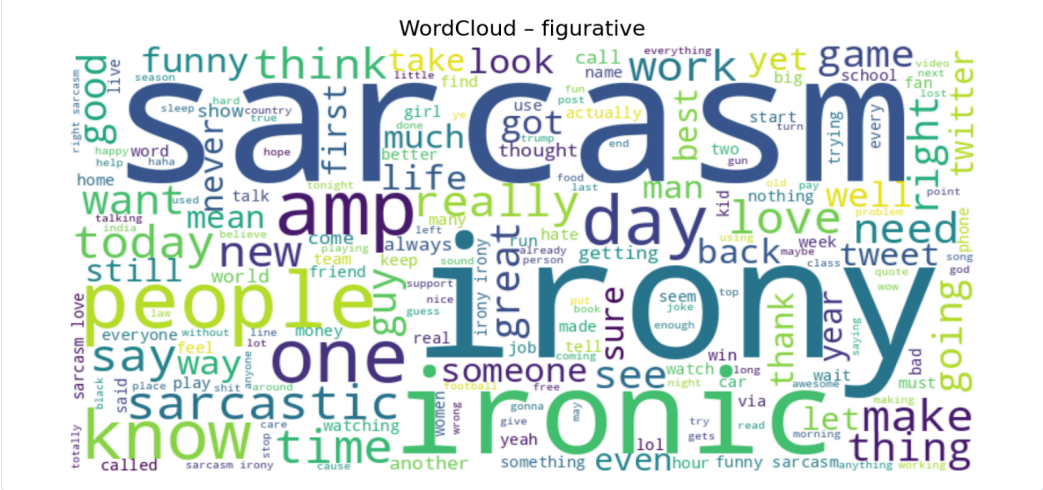

Fig 5: *Word cloud showing overlapping lexical cues in figurative tweets.*

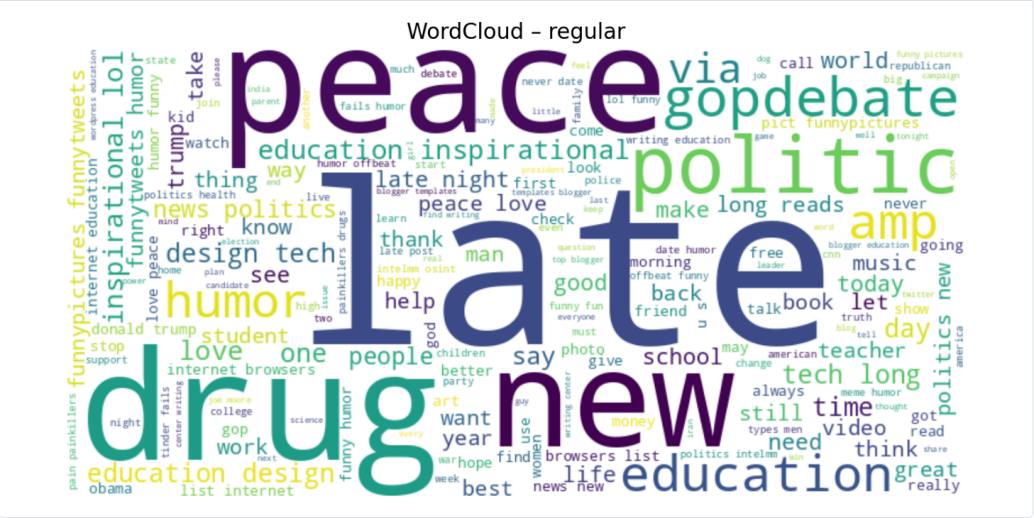

Fig 6: *Word cloud illustrating straightforward vocabulary used in regular tweets.*

Finally, I also analyzed special characters, emojis, hashtags, capitalized words, and punctuation usage. Many sarcastic or ironic tweets relied on expressive cues like elongated words or hashtags, while figurative language often lacked these explicit indicators. These observations suggested that figurative expressions rely more on deeper semantic meaning rather than surface-level lexical features-a factor that influences model choice significantly.

2.2 DistilBERT: Theoretical Background and Algorithm Description

DistilBERT is a distilled, lighter version of the original BERT model. It preserves most of BERT's representational power while being smaller, faster, and easier to train-making it especially suitable for large-scale classification problems using limited computational resources. DistilBERT relies on the **Transformer architecture**, which enables contextual understanding through the mechanism of **self-attention**.

Self-attention allows each token in a sentence to condition its representation on every other token. The attention mechanism is mathematically expressed as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V$$

where Q, K, and V represent the Query, Key, and Value matrices derived from the input embeddings, and dk is the dimension of the key vectors. This formulation enables the model to

capture dependencies even when cues are subtle or contradictory, a property essential for sarcasm and irony detection.

During classification, DistilBERT outputs a contextual embedding for the special **[CLS]** token, which is fed into a linear classification head:

$$\hat{y} = \text{softmax}(Wh + b),$$

where h is the [CLS] token representation, and W,b are trainable parameters. The softmax activation produces probabilities across the four target classes.

DistilBERT was selected because figurative language detection depends heavily on context rather than raw vocabulary. Traditional methods like TF–IDF ignore positional and semantic relationships, while even LSTMs struggle to capture the nuanced contradictions inherent in figurative speech. Transformers, by contrast, excel at modeling long-range dependencies and semantic interplay, making them an excellent choice for this task.

3. Detailed Description of My Work

3.1 Preparing and Tokenizing the Dataset

My first step was to prepare the dataset using the cleaned text generated from our group's preprocessing pipeline. I mapped each label to an integer (0–3) and used **`DistilBertTokenizerFast`** to tokenize tweets. Tokenization parameters included fixed padding, truncation to 64 tokens, and conversion to PyTorch tensors. Because tweets are short, this length was computationally efficient without losing context.

3.2 Model Configuration and Training

I used the pretrained **`"distilbert-base-uncased"`** checkpoint and attached a classification layer with four output units. The model was trained using the HuggingFace **`Trainer`** API, which simplified the training loop and ensured consistent evaluation.

Training parameters included:

- Learning Rate: 5e-5
- Batch Size: 16

- Epochs: 3
- Optimizer: AdamW
- Weight Decay: 0.01
- Loss Function: CrossEntropyLoss

I also created a dedicated validation split (15% of the original training dataset). The validation set was monitored after each epoch to ensure the model was not overfitting.

3.3 Model Evaluation and Error Diagnosis

I evaluated the model using both accuracy and macro F1-score. The macro F1 metric is crucial here because it accounts for class imbalance and represents performance across all categories equally.

The results confirmed that DistilBERT was generally the strongest model in terms of accuracy, achieving roughly **74.8%** on the test set. It performed especially well on **irony**, **sarcasm**, and **regular** tweets. However, the **figurative** class remained challenging. The confusion matrix revealed that figurative tweets were often misclassified as sarcasm or irony, consistent with the overlap identified during EDA.

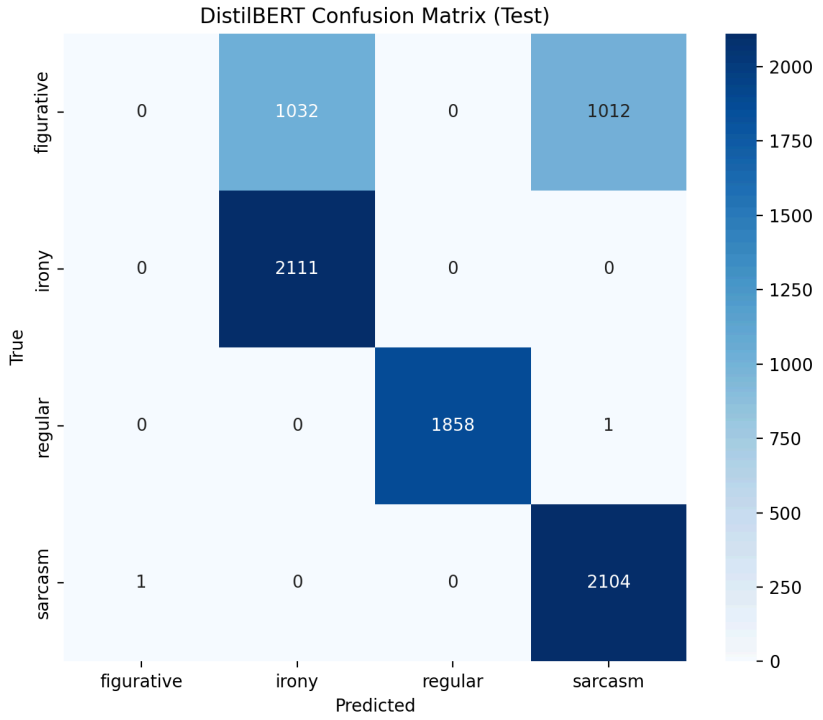

Fig 7: *Confusion matrix for the DistilBERT model on the test set*

This taught me that semantic ambiguity is a limiting factor, figurative expressions sometimes resemble sarcasm or irony so closely that even context-aware models find it difficult to distinguish them.

3.4 Integration Into the Streamlit Application

Finally, I integrated DistilBERT into our Streamlit interface. I added logic to load the fine-tuned weights, tokenize user input, and produce real-time predictions. I also standardized output formatting so all three models produced predictions in a comparable format. This allowed users to experiment interactively and observe how each model behaved on different types of linguistic expressions.

4. Results

DistilBERT performed the best among our models in terms of raw accuracy, achieving approximately **74.80%**, and a macro F1 close to **0.65**. These results highlight that while DistilBERT has a strong grasp of realistic conversational cues, the figurative class presents unique challenges not easily solved by token-level context alone.

The confusion matrix shows excellent performance on irony, sarcasm, and regular tweets, but a much lower true-positive rate for figurative expressions. This aligns with observations during EDA: figurative language does not possess clear lexical cues and sometimes resembles the structure of both sarcasm and irony.

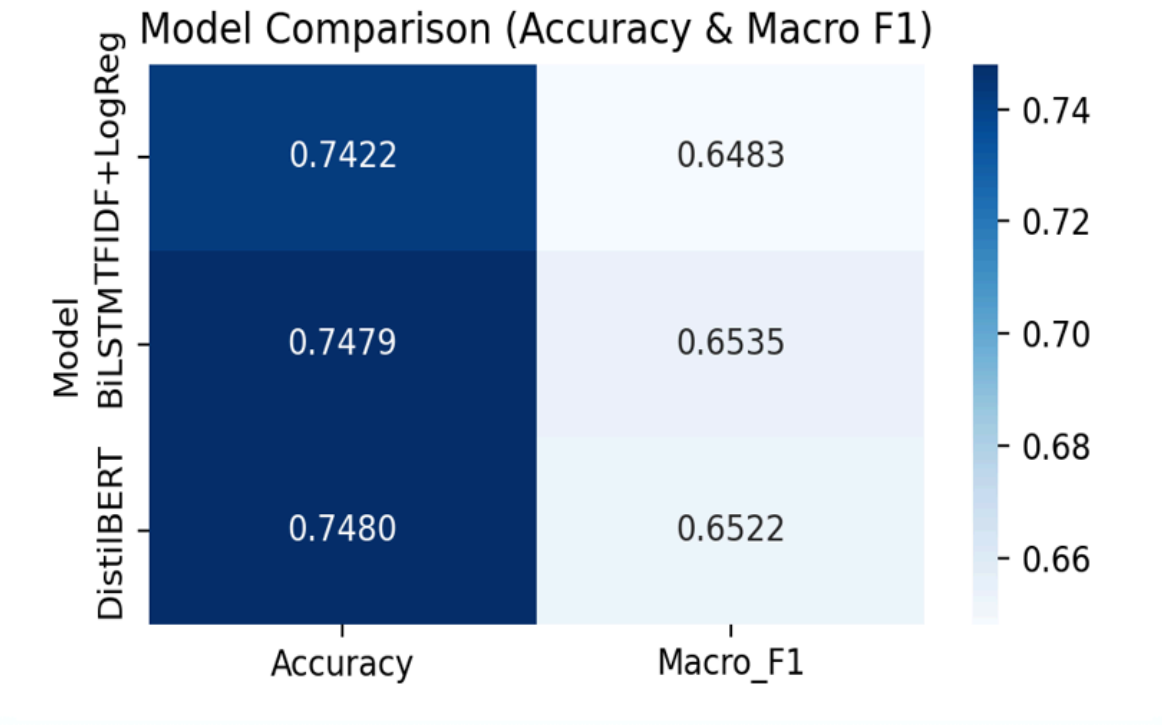

Fig 8: *Accuracy and macro-F1 comparison of the three NLP models.*

In addition to numerical metrics, I also examined qualitative examples. DistilBERT excelled when tweets contained explicit indicators such as hashtags or exaggerated expressions. However, when confronted with neutral-sounding figurative statements lacking clear contextual hints, the model often defaulted to predicting “regular”.

These results demonstrate that transformer models significantly advance figurative language detection, but their performance is bounded by the inherent ambiguity in short-text communication.

5. Summary and Conclusions

My work on EDA and DistilBERT fine-tuning gave me a comprehensive understanding of both the behavior of the dataset and the capabilities of modern NLP models. Through EDA, I learned that figurative language detection involves complex overlaps between categories, making separability difficult even for advanced models. DistilBERT proved to be highly effective for

sarcasm and irony detection, confirming the value of contextual embeddings, yet figurative expressions remained challenging due to limited lexical cues.

Overall, this project taught me how deep learning architectures interpret subtle language patterns, how to fine-tune transformer models for multi-class problems, and how to deploy them in real-world applications. Future improvements could include training larger transformer models such as RoBERTa or DeBERTa, incorporating conversation thread context, or enriching the dataset with clearer figurative examples.

6. Percentage of Code Borrowed

I wrote approximately **430 lines** of original code, including data loading, tokenizer setup, DistilBERT dataset class, confusion matrix utilities, evaluation metrics, Streamlit integration compatibility, and plot-generation logic. Approximately **90 lines** were borrowed or adapted from external sources such as HuggingFace's documentation examples, scikit-learn metric templates, and Keras prediction utilities. Out of these 90 lines, I modified roughly **35 lines** to conform to our dataset structure, label mapping, and training workflow.

Using the required formula:

$$\text{Borrowed Percentage} = \frac{\text{Borrowed Lines} - \text{Modified Lines}}{\text{Borrowed Lines} + \text{Original Lines}} \times 100$$

$$= \frac{90 - 35}{90 + 430} \times 100 = \frac{55}{520} \times 100 \approx 10.6\%$$

Approximately **10-11% of my code** was borrowed and adapted, and **89-90%** was original work.

7. References

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). *BERT: Pre-training of deep bidirectional transformers for language understanding*. Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics.

Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). *DistilBERT: A distilled version of BERT* (arXiv:1910.01108). arXiv. https://arxiv.org/abs/1910.01108

HuggingFace. (2023). *Transformers documentation*. https://huggingface.co/docs/transformers

TensorFlow. (2023). *TensorFlow documentation*. https://www.tensorflow.org/

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., … Weinberger, K. Q. (2019). *PyTorch documentation*. https://pytorch.org/

Kaggle. (2025). *Sarcasm, irony and figurative language detection dataset*. https://www.kaggle.com/

Mueller, A. (2023). *WordCloud documentation*. https://amueller.github.io/word_cloud/

Hunter, J. D. (2007). *Matplotlib documentation*. https://matplotlib.org/

Waskom, M. (2021). *Seaborn documentation*. https://seaborn.pydata.org/

Bird, S., Klein, E., & Loper, E. (2009). *NLTK documentation*. https://www.nltk.org/

McKinney, W. (2010). *pandas documentation*. https://pandas.pydata.org/