

---

# BSDA2001P: Introduction to DL and GenAI Project

---

**Author: Shipra**

## 1. Abstract

This project tackles the multi-label emotion classification challenge from the DLGenAI Project competition, which requires predicting five emotions: anger, fear, joy, sadness, and surprise, from short English text snippets. As multiple emotions may co-occur within the same text, the task is formulated as a multi-label classification problem, with performance measured using the Macro F1-score averaged across all labels.

To address this, the final system employs an ensemble of two large-scale pretrained language models: DeBERTa and RoBERTa, combined at the prediction level to exploit their complementary strengths. Data cleaning, augmentation and preprocessing, tokenization, and multi-label sigmoid outputs were used, and threshold-based decision rules were applied for label assignment. The ensemble achieved a Macro F1-score of 0.883 on the kaggle competition leaderboard, outperforming individual models.

## 2. Introduction

Automatic emotion detection from text has become an important problem in natural language processing, with applications in psychological well-being assessment, customer experience analysis, social media monitoring, and conversational AI systems. The project focuses on this challenge by providing a curated dataset of short English text entries, each annotated with five possible emotion labels: anger, fear, joy, sadness, and surprise. Submissions are evaluated using the Macro F1-score, with a baseline performance cutoff of 0.70 to encourage effective modeling strategies.

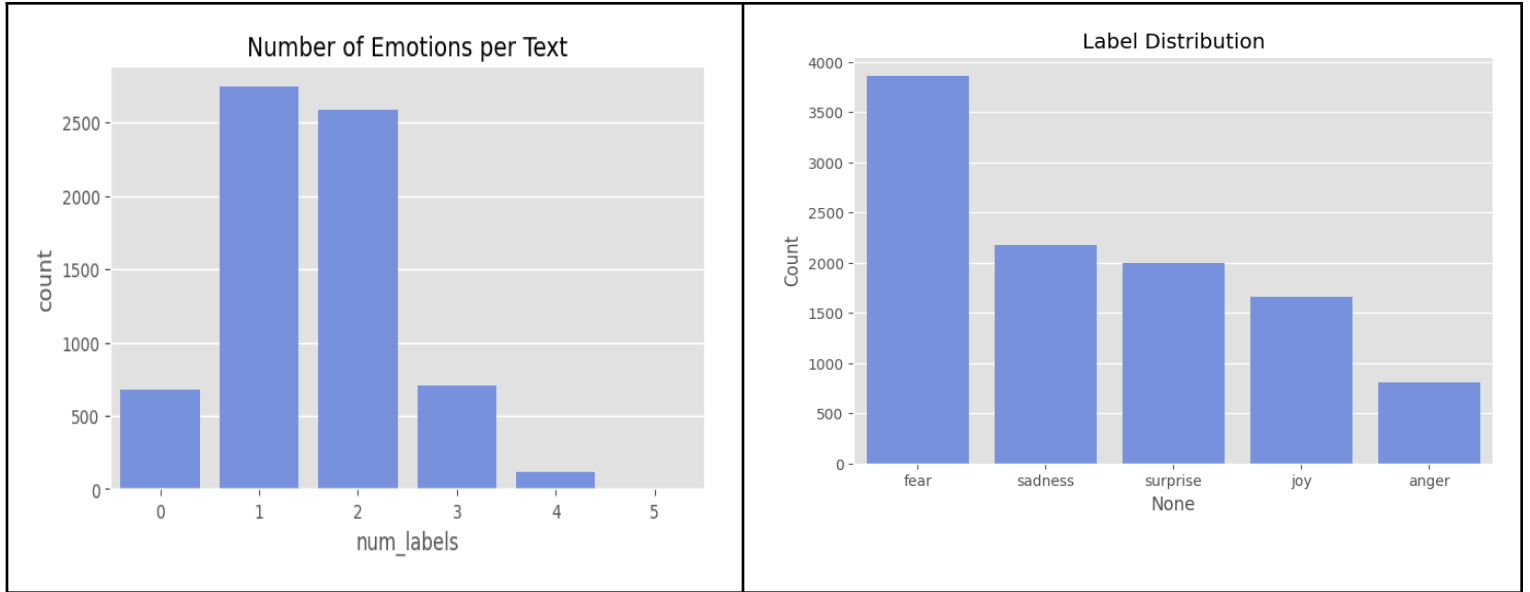
The goal of this project was twofold. First, to fine-tune state-of-the-art transformer models to leverage their contextual representation capabilities for improved emotion detection. Second, to explore classical deep learning approaches by building one model from scratch, experimenting with architectures such as LSTMs. The remainder of this report is organized as follows: the Dataset and Preprocessing section describes the data characteristics and cleaning steps; the Methodology section details model architectures, fine-tuning strategies, and scratch-model design; the Experiments and Results section covers evaluation procedures and performance outcomes; and finally, the Conclusion summarizes the main insights and future work directions.

## 3. Dataset & Preprocessing

### 3.1 Exploratory Data Analysis

The dataset comprises 6,827 English text segments annotated with five emotion labels: anger, fear, joy, sadness, and surprise. The majority of samples contain only one or two labels. Exploratory analysis revealed a class imbalance, with Fear being the dominant class and Anger the minority.

While the dataset contained no missing values, we identified significant redundancy; only 4,986 entries were unique. Texts are notably concise, with a 95th percentile length of 38 word, indicating that the models must perform classification based on short contexts. Words such as "head," "eyes," "heart," and "hands" appear frequently across multiple emotion categories, suggesting that the dataset heavily relies on physical descriptions of emotional states.



### 3.2 Data Cleaning

To ensure data quality, we implemented a targeted cleaning pipeline. First, exact text duplicates were removed from the training set to prevent model overfitting, reducing the dataset to 4,986 unique entries. We subsequently stripped specific noise artifacts, including HTML entities (e.g., & # x 26; nbsp;), special characters (<, >), and placeholders like "#name?" and "##url##". Post-cleaning validation identified a negligible number of resulting empty strings (2 in training, 3 in testing) where the text consisted solely of these artifacts.

### 3.3 Data Augmentation

To address severe class imbalance and improve generalization, we implemented a two-stage augmentation pipeline targeting the minority classes (Anger, Joy, Sadness, Surprise). The dominant class, Fear, was excluded to prevent over-representation : Utilizing Helsinki-NLP opus-mt models, samples were translated from English to Hindi and back to English. This introduced lexical diversity while preserving semantic integrity. We employed the nlpaug library with bert-base-uncased to replace words with context-aware synonyms.

Augmentation intensity was inversely proportional to class frequency; the minority class Anger received a 5x multiplier per technique, while others received a 1x multiplier. The exact multiplier was derived from trying various numbers and monitoring the balance. This process expanded the training set to 15k unique samples, shifting the Anger class from a minority to approximately 19% of the dataset.

## 4. Tokenizer

We utilized the Byte-Level Byte-Pair Encoding (BPE) tokenizer associated with the respective model architecture. The tokenizer employs a BPE-based subword approach, allowing rare or unseen words to be split into known subwords, reducing out-of-vocabulary issues and ensuring compatibility with the model's pre-trained embeddings and attention mechanism. Texts were converted to strings and labels to numeric arrays, then tokenized in batches with truncation to respect the model's maximum input length. Labels were mapped directly to the tokenized dataset for training compatibility. We applied dynamic padding per batch using DataCollatorWithPadding, padding only to the longest sequence in each batch to optimize GPU memory and computation. Tokenized words are converted to IDs and then embeddings,

which model attention processes to capture both semantic and positional information. This pipeline efficiently converts raw text into a model-readable format while fully leveraging transformer architecture, supporting high performance on multi-label emotion classification.

## 5. Modelling & Experimentation

To handle class imbalance, we used Custom Focal Loss, which down-weights easy examples and focuses on harder, misclassified samples, with  $\alpha = 0.25$  and  $\gamma = 2.0$ . This replaces the standard BCE loss in a CustomTrainer, where `compute_loss` calculates Focal Loss on model logits against true labels.

Predictions pass through a sigmoid, and per-label thresholds are optimized by scanning validation probabilities from 0.2 to 0.8 to maximize F1 scores for each emotion. Macro and per-label F1 metrics are tracked during training to monitor performance.

By combining Focal Loss with label-wise thresholding, the model mitigates bias toward dominant classes, improves recall for minority emotions, and ensures robust multi-label classification performance on imbalanced datasets.

### 5.1 DeBerta - Large

We fine-tuned DeBERTa-large (microsoft/deberta-large) [He et al., 2021], a transformer-based encoder that advances the standard BERT architecture by introducing disentangled attention and enhanced mask decoders. In disentangled attention, content and positional embeddings are processed separately, allowing the model to differentiate between the semantic meaning of a token and its position within a sequence. This separation enables more precise modeling of word dependencies and contextual relationships, which is especially important for capturing subtle emotional cues in text.

Multi-seed training (42, 123, 2025) was employed to ensure results are robust and not dependent on a single initialization. The model was fine-tuned for 7 epochs to balance sufficient learning with overfitting prevention. A batch size of 8 combined with gradient accumulation of 4 effectively simulates a larger batch size without exceeding GPU memory limits, stabilizing training. The learning rate of  $2e-5$  with a cosine scheduler and 10% warmup was chosen to allow gradual adaptation of the pre-trained weights while avoiding abrupt updates that could destabilize training. We used the AdamW optimizer, which combines Adam's adaptive learning rates with weight decay regularization to prevent overfitting.

Predictions were obtained via sigmoid activation, and macro and per-label F1 scores were tracked. Models were saved and logged with Weights & Biases for reproducibility. This approach leverages DeBERTa's architecture while addressing class imbalance for robust multi-label performance.

### 5.2 RoBERTa-Large

RoBERTa (roberta-large) [Liu et al., 2019] is an optimized variant of BERT that uses a robustly optimized pretraining approach. Unlike the original BERT, RoBERTa removes the next sentence prediction (NSP) objective, which was found to contribute minimally to downstream performance, thereby simplifying the pretraining task. The model is trained on significantly larger and more diverse corpora, including datasets such as CC-News, OpenWebText, and Stories, enabling it to capture richer linguistic patterns and broader contextual knowledge.

Multi-seed training (42, 123, 2025) was employed to ensure results are robust and not dependent on a single initialization. The model was fine-tuned for 10 epochs to balance sufficient learning with overfitting prevention. A batch size of 8 combined with gradient accumulation of 4 effectively simulates a larger batch size without exceeding GPU memory limits, stabilizing training. The learning rate of  $2e-5$  with a cosine scheduler and 10% warmup was chosen to allow gradual adaptation of the pre-trained weights while avoiding abrupt updates that could destabilize training. We used the AdamW optimizer, which combines Adam's adaptive learning rates with weight decay regularization to prevent overfitting.

Predictions were obtained via sigmoid activation, and macro and per-label F1 scores were tracked. Models were saved and logged with Weights & Biases for reproducibility. This approach leverages DeBERTa's architecture while addressing class imbalance for robust multi-label performance.

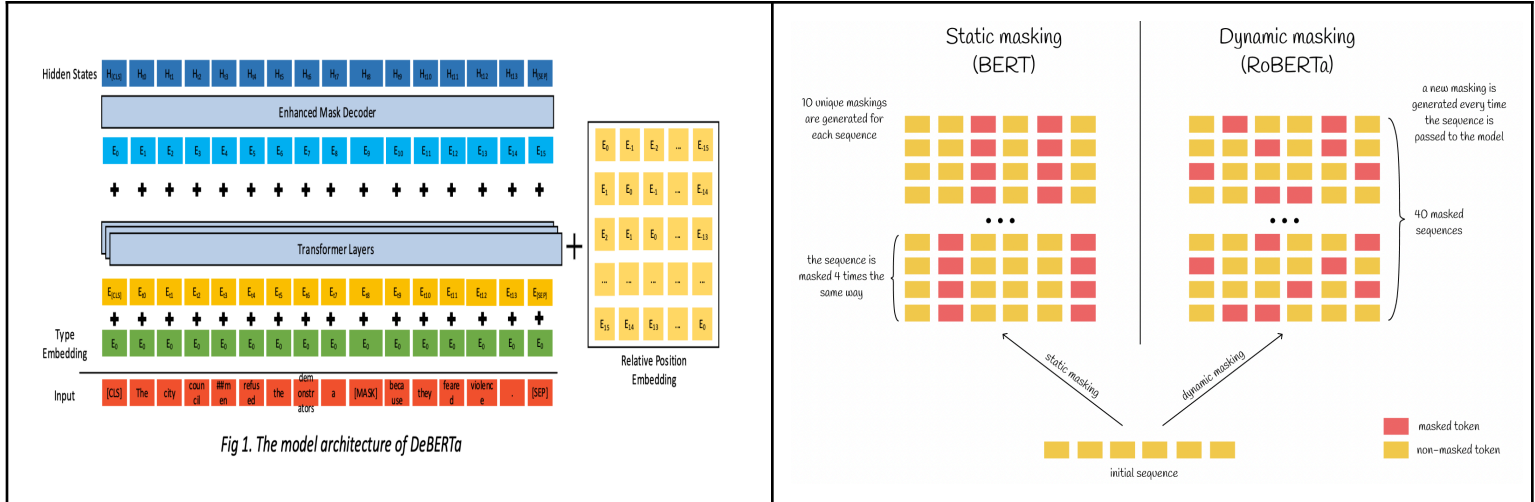


Figure: DeBERTa (distangled embedding) and RoBERTa (dynamic masking) architectures

### 5.3 Bi-Directional LSTM

The core of our model architecture is a bidirectional Long Short-Term Memory (LSTM) network, which is well-suited for sequential text data due to its ability to capture long-range dependencies and contextual information. Unlike standard recurrent networks, LSTMs incorporate gating mechanisms: input, forget, and output gates, that regulate the flow of information through the network, enabling it to retain relevant patterns and discard irrelevant ones across long sequences. The bidirectional configuration allows the model to access both past (forward states) and future (backward states) context for each token, which is particularly beneficial for emotion recognition where the sentiment or emotional cues may appear at different positions in the text.

We implemented a bidirectional LSTM model for multi-label emotion classification on five emotions: anger, fear, joy, sadness, and surprise. A fixed random seed (42) ensured reproducibility across PyTorch, NumPy, and CUDA. Texts were preprocessed using a cleaning pipeline that normalized Unicode characters, removed excessive whitespace, and replaced newlines and tabs with spaces. A SentencePiece BPE tokenizer with a vocabulary size of 28,000 was trained on the training corpus to convert text into subword token IDs, which were padded or truncated to a maximum length of 256 tokens.

The LSTMEmotion model consists of an embedding layer, a bidirectional LSTM with hidden size 128, batch normalization, dropout, and a final linear layer producing logits for all five labels. Inputs were moved to GPU when available, and training used the AdamW optimizer with a cosine annealing learning rate scheduler (learning rate  $2e-3$ ) for 10 epochs. BCEWithLogitsLoss handled multi-label outputs, and gradient clipping (1.0) stabilized training.

During each epoch, training loss was accumulated, and validation was performed with sigmoid activation and a 0.5 threshold. Macro F1 was computed per epoch, logged via Weights & Biases, and the best model was saved. This setup efficiently combines tokenization, bidirectional LSTM encoding, and careful optimization to achieve robust multi-label classification performance

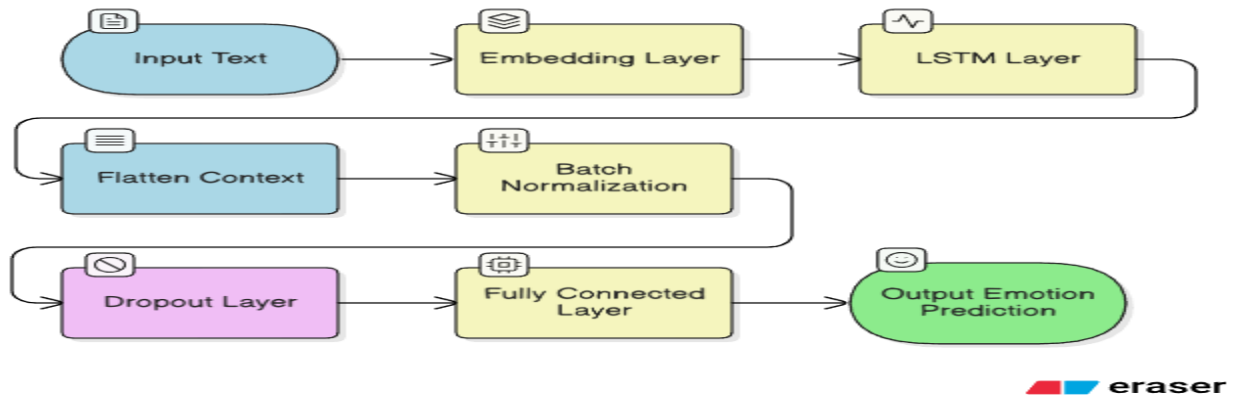
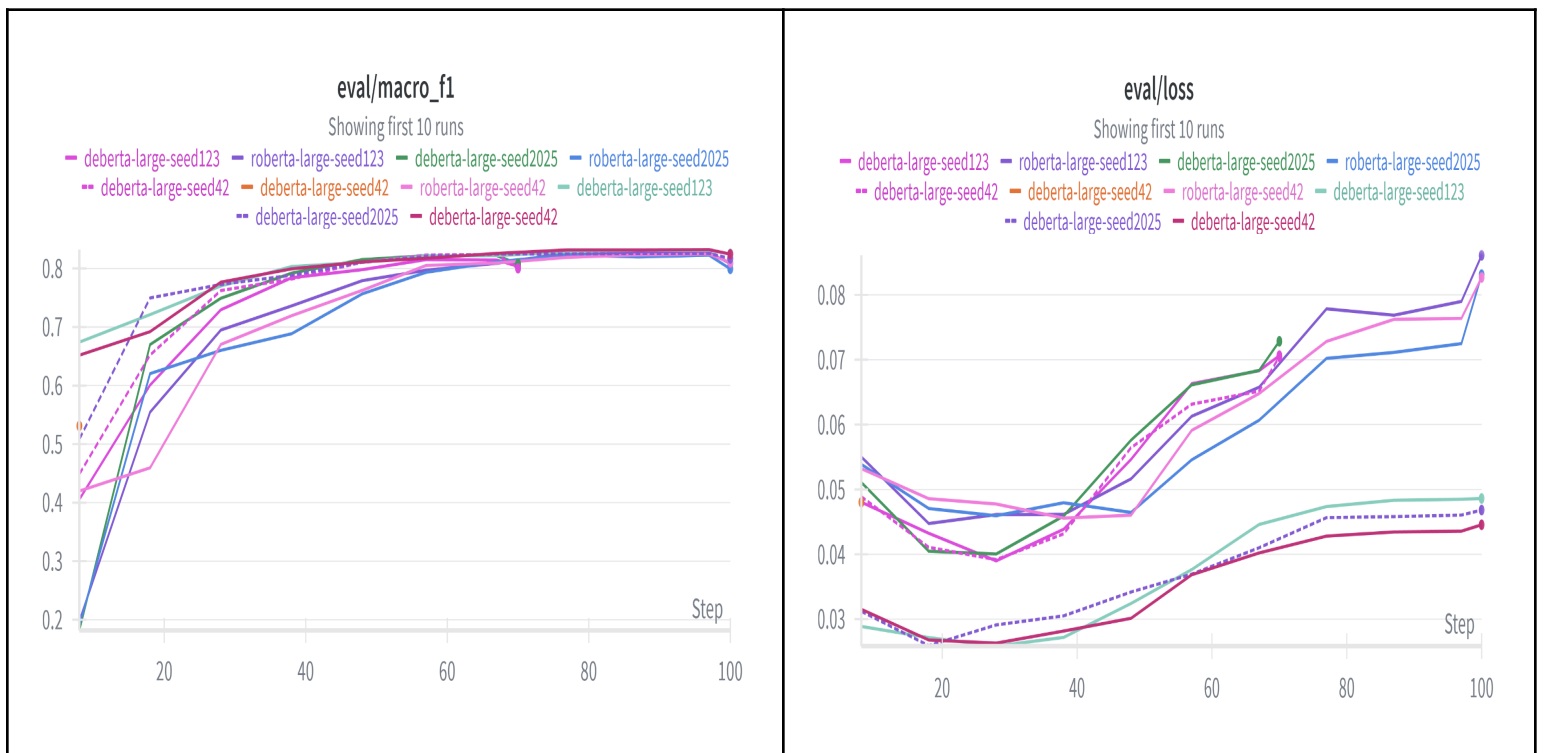


Figure: LSTM Model Architecture

## 6. Evaluation & Metrics

To evaluate the multi-label classification performance, we monitored Binary Cross-Entropy Loss and Macro-Averaged F1 scores across both training and validation sets for all model variants.

Macro F1 was chosen as the primary metric to provide an unweighted mean of per-label scores, ensuring that performance on minority classes was represented equally alongside frequent classes.



Model	LeaderBoard Score
DeBERTa Baseline	0.87
DeBerta Advanced with focal loss, optimal threshold and seed ensemble	0.877
RoBERTa Baseline	0.865
RRoBerta Advanced with focal loss, optimal threshold and seed ensemble	0.87
Ensemble of RoBERTa and DeBERTa	0.883

## 7. Conclusion

This project tackled multi-label emotion classification using a combination of strong preprocessing, class balancing, and modern transformer models. After cleaning and augmenting the dataset, DeBERTa-large and RoBERTa-large were fine-tuned with focal loss, optimal thresholding, and multi-seed training, significantly improving performance on minority classes. A Bi-LSTM model was also built from scratch, offering useful insights but performing below transformer baselines.

The best results came from an ensemble of DeBERTa and RoBERTa, which achieved a Macro F1-score of 0.883 on the leaderboard—higher than any individual model. This demonstrates that complementary transformer architectures and careful optimization provide strong advantages for multi-label emotion detection. Overall, the project shows that combining preprocessing, balanced training strategies, and advanced language models leads to reliable and competitive performance.

## 8. References

- He, P., Liu, X., Gao, J., & Chen, W. (2021). *DeBERTa: Decoding-enhanced BERT with Disentangled Attention*. Microsoft Research.
- Liu, Y., Ott, M., Goyal, N., et al. (2019). *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. Facebook AI.
- DLGenAI Project Competition Dataset, IITM BS Data Science Program
- Kaggle Platform (Leaderboard evaluation environment).
- Hugging Face Transformers — model fine-tuning, tokenization, and training utilities.
- PyTorch — deep learning framework used for custom LSTM and trainer modifications.
- SentencePiece — subword tokenizer training for LSTM model.
- nlpaug — data augmentation toolkit for contextual synonym replacement.
- Helsinki-NLP Opus-MT models — translation-based augmentation.
- Weights & Biases (wandb) — experiment tracking and logging.
- Scikit-learn — evaluation metrics (F1-score, threshold tuning).
- Python 3.10 — development environment.
- Google Colab / GPU runtime — training infrastructure.