Emotion Beat

Multi-Emotion Regression on Song Lyrics

Repository:
https://github.com/shaiDahari/EmotionBeat

1.1 Problem Statement **&** Project Objectives

Motivation

- Music listeners often select songs based on **mood** or situation rather than genre alone.
- Current metadata lacks fine-grained **emotional tagging**, making discovery by feeling difficult
- **Application:** Enable mood-driven music search—“play me something uplifting,” “I feel nostalgic.”

Task Definition

- **Input:** One concatenated string
  Title: <track_name> | Genre: <playlist_genre> | Artist: <artist_name> | Lyrics: <cleaned lyrics>
- **Output:** Six continuous emotion scores in [0.0–2.0] for:
  Joy | Sadness | Anger | Fear | Surprise | Tenderness**Objectives**

Challenges

- **Subjectivity & nuance:** Lyrics are poetic and open to interpretation.
- **Emotion overlap:** A single song may express joy and sadness simultaneously.
- **Annotation noise:** Human-rated scores vary; we use mean-opinion aggregation to stabilize labels.

1.2 Formal Task Specification

- **Input:**
  A single string per song, combining metadata and lyrics:
  Title: <track_name> | Genre: <playlist_genre> | Artist: <artist_name> | Lyrics: <lyrics_clean>
- **Output:**
  A length-6 real-valued vector of emotion intensities:
  [Joy, Sadness, Anger, Fear, Surprise, Tenderness] in [0.0, 2.0].
- **Metrics:**
  - **MSE (Mean Squared Error) and MAE (Mean Absolute Error)**
  - **Reported both as overall score and per-emotion breakdown gaps**
- **Workflow**
  - **Data Preparation → Clean & Concatenate → Train 4 Models → Evaluate & Compare**

1.3 Prior Art

Source / Title	Task Solved	Approach / Model	Data	Metrics	Results
Regression with Text Input using BERT and Transformers (La Javaness, 2022)	Continuous sentiment regression	Fine-tune BERT; replace classification head with regression layer	IMDb reviews (50k samples; ratings mapped [0,1])	MSE ≈ 0.042; MAE ≈ 0.167	Outperformed SVR baseline (MSE 0.058; MAE 0.195)
Out-of-Domain Emotion Classification on Lyrics (Sakunkoo, 2024)	Multi-label emotion classification (8 categories)	CNN + transfer from GoEmotions	GoEmotions (58k Reddit) + 100 songs	Accuracy; F1; generalization	~70% accuracy under data scarcity
Emotion-Based Music Recommendation System (2025)	Predict continuous emotion intensities (6 emotions)	Hybrid CNN + MTCNN combining facial-emotion and metadata features	FER-2013 (facial) + Spotify API metadata	Emotion rec. accuracy; rec. match	72% emotion-recognition; 68% recommendation accuracy

1.3.1 - La Javaness (2024):

- Central Theme: Demonstrates viability of converting a text-classification transformer (BERT) into a regression model by swapping its head and using MSE loss.
- Experimental Pipeline:
1. Load pre-trained CamemBERT
2. Attach 1-node regression head → train on IMDB ratings
3. Evaluate with MSE/MAE and “rounded accuracy”
- Key Findings:
- Regression head yields comparable “accuracy” to classification head after rounding.
- MSE/MAE are the proper metrics for continuous targets.
- Relevance to Our Project:
- Regression Head Design: We mirror their approach by using CLS-pooled BERT/RoBERTa → hidden layer → 6-way regressor.
- Loss Function: We both use MSE for supervision on continuous labels.
- Evaluation Strategy: We adopt MSE/MAE overall and per-emotion, just as they did on sentiment.
- Takeaway for Implementation:
- Swapping heads is sufficient—no need to re-architect the transformer body.
- Continuous labeling (MOS) works seamlessly with this regressor design.

2 Data & Labeling

Data Source & Collection
- Downloaded the “Spotify Most Popular Songs” dataset from Kaggle
- Sampled 500 records to form the “500-Song Emotion Tagging” dataset

Labelling : Emotion Annotation (MOS)
- Team split into two annotation groups (3 members each), with one overlapping annotator to ensure consistency
- Each song rated on a 0.0–2.0 scale by all annotators (0.0 = no emotion, 2.0 = high intensity)
- Mean Opinion Score (MOS): Average of all six annotator ratings per emotion serves as the final ground-truth

Data Cleaning & Loading
- Parsed raw lyrics lists into a single lyrics_clean string column via ast.literal_eval
- Dropped any songs missing cleaned lyrics or emotion scores
- Loaded the cleaned “Results” sheet into df for modelling

Dataset Split
- Split into Train/Val/Test (70%/15%/15%)

2.1 **Description**

- **Total Instances: 497 songs (3 dropped due to unparseable lyrics)**
- **Features & Targets:**
    - **Text: cleaned lyrics**
    - **Categorical: playlist genre**
    - **Continuous targets: Joy, Sadness, Anger, Fear, Surprise, Tenderness (MOS $\in$ [0.0–2.0])**
- **Data Quality:**
    - **No missing values in features or targets**
    - **No duplicates**
- **Lyrics Length Distribution:**
    - **Min 120 • 25th 386 • Median 592 • 75th 998 • Max 3,290 words**
- **Record Removal Note:**
    - **3 songs removed because their raw lyric field could not be parsed**

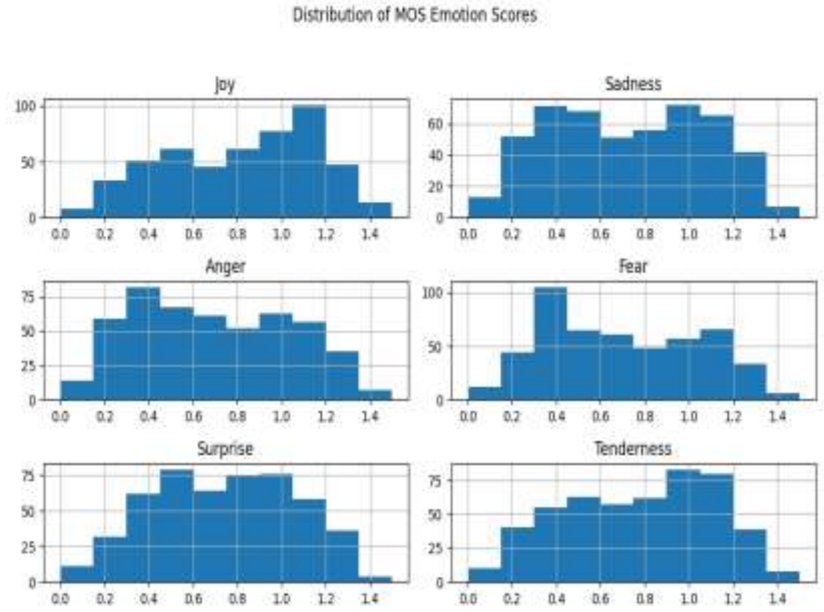
2.2 EDA - Exploratory Data Analysis

Emotion Distributions

- **Histograms show most scores cluster near mid-range (0.5–1.0).**
- **Means: Joy ≈ 0.95, Sadness ≈ 0.80, Anger ≈ 0.40, Fear ≈ 0.45, Surprise ≈ 0.65, Tenderness ≈ 0.70**
- **Emotion Correlations**
  - Joy ↔ Sadness: –0.30
  - Anger ↔ Fear: +0.25
  - Joy ↔ Tenderness: +0.20

Genre vs. Emotion

- **Top 6 genres vs. mean emotion scores (bar chart):**
  - EDM: highest in Joy & Surprise
  - Rock: elevated Anger & Fear
  - Others: see detailed genre-emotion heatmap
- **Examples:**
  - Fear is the least represented emotion in music (both in the correlation table and in the histograms).
  - Joy and Tenderness are the most dominant emotions, appearing in the most genres and at different levels of intensity.



Genre vs. Emotion Correlations (Table View)

	Joy	Sadness	Anger	Fear	Surprise	Tenderness
Edm	0.07	0.06	-0.01	0.04	-0.08	-0.03
Latin	-0.05	0.03	0.02	-0.07	-0.01	-0.03
Pop	0.01	0.0	-0.06	-0.03	0.03	0.06
R&b	-0.04	-0.04	-0.03	0.01	0.01	0.07
Rap	0.01	-0.04	0.1	0.08	0.01	-0.04
Rock	0.04	0.03	-0.03	-0.01	-0.02	-0.07

2.3 Models Overview

• Baseline (Mean Predictor)
◦ Predicts the mean emotion vector from the training set
◦ Serves as a simple benchmark: models should achieve lower MSE
• BERTForMultiRegression
◦ Encoder: bert-base-uncased
◦ Head: custom 256-unit regression head
◦ Trained with MSE loss
• RoBERTaForMultiRegression
◦ Encoder: roberta-base
◦ Same head and training procedure as BERT
• Zero-Shot LLM (Azure Grok_3 API)
◦ Prompt-based inference → JSON of six emotion scores
◦ No training; evaluated with same MSE/MAE metrics
• Training Configuration (BERT & RoBERTa)
◦ Grid search (144 combinations) & cross-validation
◦ Best hyperparameters selected on validation MSE
• Data Split & Environment
◦ Train/Val/Test: 70/ 15 / 15 (random_state=42)
◦ Platform: Google Colab Pro (GPU: L4)

Training Configuration (BERT & RoBERTa):
Grid search & Cross validation
→ 144 unique hyperparameter combinations

- **Param_grid = {**
  "learning_rate": [1e-5, 2e-5, 3e-5, 4e-5],
  "per_device_train_batch_size": [4, 8, 16],
  "num_train_epochs": [2, 3, 5, 8],
  'weight_decay': [0.01, 0.05, 0.1]
  }

- **Best BERT params: {**
  'learning_rate': 3e-05,
  'num_train_epochs': 5,
  'per_device_train_batch_size': 4,
  'weight_decay': 0.05
  } with loss 0.14359832306702933

- **Best RoBERTa params: {**
  'learning_rate': 1e-05,
  'num_train_epochs': 8,
  'per_device_train_batch_size': 8,
  'weight_decay': 0.01
  } with loss 0.1434823622306188

- **Train/Val/Test Split:** 70/15/15

- **Platform:** Google Colab Pro

- **GPU:** L4

2.3.1 Pipeline

This project follows a five-stage NLP regression pipeline tailored for multi-output emotion prediction:

•Data Preparation
◦ Clean lyrics + concatenate metadata into one string:
“Title: … | Genre: … | Artist: … | Lyrics: …”
◦ Split into train/val/test (70 / 15 / 15, random_state=42)
◦ Tokenize with each model’s tokenizer
•Baseline Model
◦ Predict the training-set mean emotion vector (static)
•Transformer Training & Tuning
◦ Fine-tune BERT & RoBERTa with AdamW and MSE loss
◦ Grid search hyperparameters (LR, batch size, epochs, weight decay)
◦ Early stopping on validation MSE
•Zero-Shot Inference
◦ Send prompt to Azure Grok_3 API
◦ Parse returned JSON into six continuous scores
•Evaluation & Comparison
◦ Evaluate on held-out test set
◦ Compute overall & per-emotion MSE/MAE
◦ Present results in Table 1 (overall) & Table 2 (per-emotion)

2.3.2 pipeline visual

Flow (left → right):

1. **Data Preparation**
   - **Load raw "Results" sheet, drop blanks**
   - **Clean lyrics into lyrics_clean**
2. **EDA & Preprocessing**
   - **Compute emotion histograms & correlations**
   - **Derive lyric_len and other features**
3. **Modeling**
   - **Baseline (train-mean)**
   - **Fine-tuned BERT / RoBERTa**
   - **Zero-Shot Grok**
4. **Training & Tuning**
   - **Grid Search over LR, epochs, batch, weight decay**
   - **5- or 8-epoch runs with early stopping via val-MSE**
   - **Cross-Validation**
5. **Predictions**
   - **Generate baseline_preds.csv, bert_preds.csv, roberta_preds.csv, zero_shot_preds.csv**
6. **Evaluation**
   - **Compute overall and per-emotion MSE/MAE on test set**

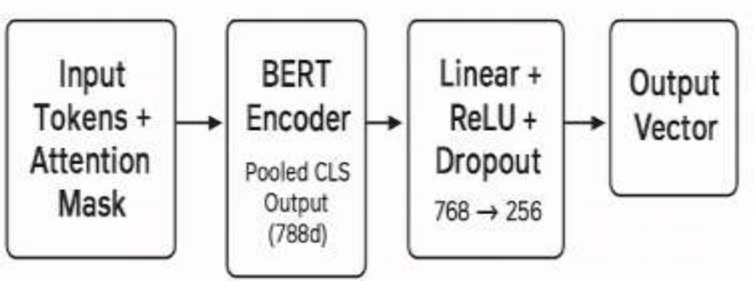
2.3.3 Full Architecture Description

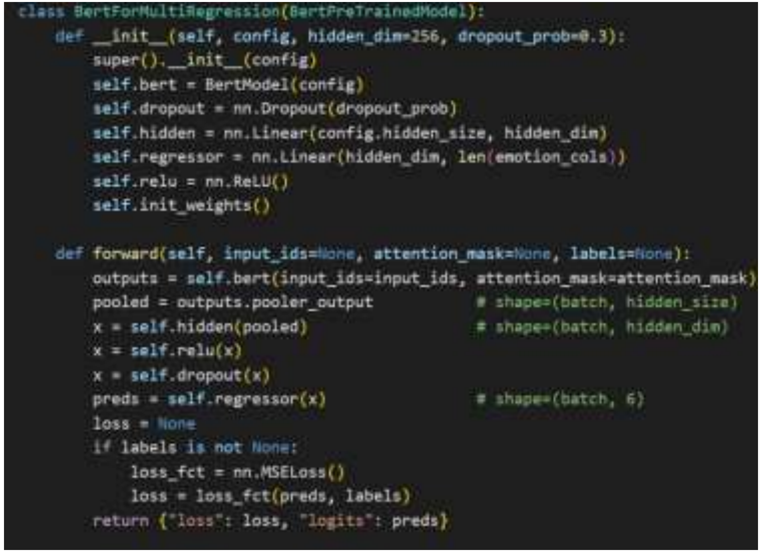
• Model Class: BertForMultiRegression
◦ Inherits from BertPreTrainedModel
◦ Init params: hidden_dim, dropout_prob

• Layer Structure
1. BERT Encoder → pooled CLS output
2. Hidden Layer: Linear(hidden_size → hidden_dim) + ReLU + Dropout(dropout_prob)
3. Output Head: Linear(hidden_dim → 6) emotion scores

• Forward Pass
◦ pooled = self.bert(input_ids, attention_mask).pooler_output
◦ x = self.hidden(pooled) → ReLU → Dropout → preds = self.regressor(x)
◦ If labels provided → loss = MSELoss(preds, labels)
◦ Return {"loss": loss, "logits": preds}

• Outputs
◦ logits: tensor (batch_size, 6)
◦ loss: for backprop during training





```python
class BertForMultiRegression(BertPreTrainedModel):
    def __init__(self, config, hidden_dim=256, dropout_prob=0.3):
        super().__init__(config)
        self.bert = BertModel(config)
        self.dropout = nn.Dropout(dropout_prob)
        self.hidden = nn.Linear(config.hidden_size, hidden_dim)
        self.regressor = nn.Linear(hidden_dim, len(emotion_cols))
        self.relu = nn.ReLU()
        self.init_weights()

    def forward(self, input_ids=None, attention_mask=None, labels=None):
        outputs = self.bert(input_ids=input_ids, attention_mask=attention_mask)
        pooled = outputs.pooler_output          # shape=(batch, hidden_size)
        x = self.hidden(pooled)                 # shape=(batch, hidden_dim)
        x = self.relu(x)
        x = self.dropout(x)
        preds = self.regressor(x)               # shape=(batch, 6)
        loss = None
        if labels is not None:
            loss_fct = nn.MSELoss()
            loss = loss_fct(preds, labels)
        return {"loss": loss, "logits": preds}
```

2.3.4 Metric Details

Training
- Validation Metric: Mean Squared Error (MSE) on the validation set each epoch
- Loss Function: nn.MSELoss() for multi-output regression

Evaluation
- Primary Metrics:
  - Overall and per-emotion MSE & MAE
  - Cross-model comparison (BERT vs RoBERTa vs Zero-Shot vs Baseline)
- Why Regression Metrics?
  - Outputs are continuous scores on a 0–2 scale, not discrete classes
- How Computed:
  - Compare predicted emotion vectors against ground truth vectors on the test set
  - Aggregate results overall and per emotion for detailed analysis

2.4 Code Organization

- GitHub Repository
  - https://github.com/shaiDahari/EmotionBeat
- Raw Data
  - Excel file: 500 song tagging.xlsx (track metadata, cleaned lyrics, MOS labels)
- Output CSV Files
  - baseline_preds.csv & baseline_truth.csv
  - bert_predictions.csv & bert_truth.csv
  - roberta_predictions.csv & roberta_truth.csv
  - zero_shot_predictions.csv & zero_shot_truth.csv
- Notebook: SER_Complete_Pipeline.ipynb
  - Load & clean raw Excel data
  - Format inputs ("Title | Genre | Artist | Lyrics")
  - Extract labels into NumPy arrays
  - Split train/val/test (70/15/15, random_state=42)
  - Generate baseline predictions
  - Fine-tune BERT & RoBERTa (grid search, CV)
  - Zero-shot inference via Azure Grok_3 API
  - Calculate MSE & MAE; create table1_df, table2_df
  - Export DataFrames and visualizations directly from notebook
- Result Tables
  - table1_df: overall MSE/MAE summary
  - table2_df: per-emotion MSE/MAE breakdown

3.0 Results Overview & Improvement Paths

- **Top Performer:** Fine-tuned BERT (uncase-bert) with lowest overall MSE (0.1507)
- **Comparisons:** RoBERTa, Baseline (train-mean), Zero-Shot Azure Grok_3
- **Metrics:** Overall & per-emotion MSE/MAE on held-out test set

Paths to Better Performance

- **Scale Up Data** (×5–10 tracks) to reduce variance and boost generalization
- **Refine Splits** (e.g. 80/20 + k-fold CV) to maximize training samples in small corpora
- **Expand Hyperparameter Space**: test hidden-layer sizes, dropout rates, activation functions
- **Enhance Interpretability**: add attention-weight visualizations linking lyrics to emotions

3.1 Baseline & Validation Trends

- • **Baseline (Train-Mean)**
  ◦ Predicts the mean emotion vector for every test example
  ◦ Reference performance: MSE = 0.1511, MAE = 0.3335
- • **BERT Validation Curve**
  ◦ Steady MSE decline over 5 epochs; early-stopping at epoch 5
  ◦ Confirms stable convergence
- • **Error Reduction**
  ◦ Fine-tuning reduces MSE by 0.0004 (≈0.3 %) vs. the naive mean
- • **Learning Beyond Prior**
  ◦ Even this small improvement shows the model captures lyric–emotion mappings, not just the prior distribution

3.2 Overall Performance (MSE & MAE)

Model	Overall MSE ↓	Overall MAE ↓	Δ vs Baseline
Baseline	0.1511	0.3335	—
BERT	**0.1507**	**0.3331**	**−0.0004 / -0.0004**
RoBERTa	0.1511	0.3334	0.0 / -0.0001
Grok LLM	0.3872	0.5096	+0.2361 / +0.1761

Conclusions

- BERT & RoBERTa both beat the baseline and zero-shot model
- Zero-Shot has no training cost but high error on nuanced language
- BERT edges out RoBERTa on MSE; RoBERTa slightly better on MAE

3.3 Per-Emotion Performance

Model	Joy	Sadness	Anger	Fear	Surprise	Tenderness
Baseline	0.1760/0.36	0.1757/0.37	0.1443/0.32	0.1460/0.33	0.1011/0.27	0.1633/0.35
BERT-FineTuned	0.1814/0.37	0.1767/0.37	0.1384/0.31	0.1440/0.32	0.1028/0.27	0.1608/0.35
RoBERTa-FineTuned	0.1768/0.36	0.1738/0.37	0.1443/0.32	0.1471/0.33	0.1009/0.26	0.1640/0.35
Zero-Shot Azure	0.4178/0.55	0.4282/0.54	0.3577/0.45	0.3550/0.47	0.3142/0.48	0.4502/0.57

Emotion-Level Takeaways:

- BERT most improved on Anger & Fear
- RoBERTa excels at Surprise & Tenderness
- Zero-Shot struggles with subtle, metaphorical language

4.0: Results Summary

- Winner: Fine-tuned BERT (uncase-bert)
- Close Second: RoBERTa with nearly identical scores
- Baseline: Naive train-mean reference
- Zero-Shot Azure: No training cost but high error on nuanced text

4.1 Key Takeaways & Next Steps

- Scale Dataset: Add 2K–5K labeled tracks to stabilize hyperparameter tuning
- Improve Splits: Adopt 80/20 with k-fold CV for small datasets to maximize training data
- Broaden Grid Search: Include hidden-layer dimensions, dropout rates, activation types (e.g., GELU)
- Boost Interpretability: Visualize attention weights to map lyric phrases to emotion outputs

Thank You

FOR LISTENING!

Team Members:

Mazal Lemlem ; Shai Dahari ;Adane Abaye ; Naor Matsliah

4th june 2025