Universidad Panamericana
Master's in Data Sciences
Natural Language Processing

Final Project: *Sentiment Analysis of FIFA World Cup Tweets*

Enrique Ulises Báez Gómez Tagle

October 5, 2025

Contents

Abstract

This project aims to perform sentiment analysis on tweets related to the FIFA World Cup 2022 using both traditional machine learning and deep learning models. The dataset contains 30,000 tweets collected on the first day of the tournament using the `Snscrape` library and pre-labeled with the `cardiffnlp/twitter-roberta-base-sentiment-latest` model from Hugging Face. We compare classical approaches (e.g., Logistic Regression, SVM) using TF-IDF features against neural architectures such as LSTM, CNN, and Transformer-based models. Evaluation metrics include Accuracy, Precision, Recall, and F1-score. The results highlight how deep learning models outperform traditional ones in handling contextual nuances of tweets.

1 Introduction

The FIFA World Cup is one of the largest and most influential sporting events worldwide, attracting billions of viewers and generating massive engagement across digital platforms. During the 2022 edition in Qatar, social media—especially Twitter—became a real-time forum for fans to express opinions, emotions, and reactions to matches, players, and events. This global activity creates a rich source of textual data that can be leveraged to analyze public sentiment and collective emotional dynamics surrounding major sports events.

Understanding the sentiment behind these tweets is important for several reasons. First, it provides valuable insights into fan behavior, public perception of teams and players, and the emotional impact of sporting events on different audiences. Second, sentiment analysis can assist broadcasters, sponsors, and organizations such as FIFA in gauging audience engagement and reputation trends. Finally, from a data science perspective, it represents a challenging Natural Language Processing (NLP) problem due to the informal, multilingual, and often sarcastic nature of social media text [1, 2, 3].

Sentiment analysis, or opinion mining, aims to classify text into categories such as positive, negative, or neutral, based on the emotional tone conveyed by users. Previous research has explored a range of approaches—from traditional machine learning models such as Logistic Regression, Support Vector Machines (SVM), and Random Forests [1] to deep learning architectures like Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks [2]. More recently, transformer-based models such as BERT and RoBERTa have achieved state-of-the-art results in sentiment classification tasks on Twitter data [3, 4].

Building upon this body of work, this project focuses on the sentiment analysis of tweets related to the 2022 FIFA World Cup. Tweets are categorized into three sentiment classes—positive, neutral, and negative—using both traditional machine learning and modern deep learning models. The goal is to evaluate how different text representations and architectures perform in understanding contextual nuances, emotion, and sarcasm in sports-related tweets. Through this comparison, the study contributes to the broader understanding of how NLP techniques can extract meaningful insights from real-world social media discourse.

2 Related Work

Previous research on sentiment analysis of sports-related tweets has evolved significantly over the past decade, progressing from traditional machine learning approaches to deep neural and transformer-based architectures.

Early works (2016–2018) commonly employed statistical text representations such as Bag-of-Words and TF-IDF combined with models like Logistic Regression, Linear SVC, Random Forest and Bayesian Logistic Regression. For instance, Barnaghi et al. [1] analyzed FIFA World Cup 2014 tweets using TF-IDF with Bayesian Logistic Regression, achieving competitive accuracy but later works noted limitations in capturing sarcasm and emoji-driven sentiment.

From 2018 onward, deep learning architectures such as CNN and LSTM began outperforming traditional classifiers in sentiment analysis tasks. Venkatesh et al. [2] proposed a CNN-LSTM hybrid for classifying sports tweets, demonstrating that combining convolutional and recurrent layers improved contextual understanding.

The next major shift came with transformer-based models like BERT and RoBERTa, which leverage self-attention mechanisms and large-scale pretraining. Barbieri et al. [3] introduced the TweetEval benchmark,

fine-tuning RoBERTa for multiple Twitter NLP tasks and achieving state-of-the-art sentiment accuracy. More recent studies (2021–2025) fine-tuned RoBERTa specifically on sports tweets [4], reporting superior F1-scores and robustness compared to CNN and LSTM models.

This project builds upon these advancements by directly comparing traditional machine learning models (Logistic Regression, Linear SVC, Random Forest) with deep learning architectures (LSTM, CNN) and transformer-based methods (RoBERTa as both feature extractor and fine-tuned model) on FIFA World Cup 2022 tweets. By systematically evaluating across these paradigms, this work highlights the evolution of sentiment analysis methods and quantifies the contextual advantages of transformer-based approaches for real-world, event-driven social media data.

3 Methodology

3.1 Dataset Description

The dataset used in this project is titled **"FIFA World Cup 2022 Tweets"**, consisting of approximately **30,000 tweets** collected during the first day of the tournament in Qatar 2022.

Data was obtained using the `Snscrape` library for scraping and labeled automatically with the pre-trained RoBERTa-based sentiment model `cardiffnlp/twitter-roberta-base-sentiment-latest` from Hugging Face. The dataset is stored as a CSV file named `fifa_world_cup_2022_tweets.csv` and contains the following columns:

- **Date Created**: Timestamp of tweet publication.
- **Number of Likes**: Engagement measure.
- **Source of Tweet**: Platform or device used to post.
- **Tweet**: Raw text content of the tweet.
- **Sentiment**: Pre-assigned label (Positive, Neutral, or Negative).

The dataset is publicly available under the **CC0 1.0 Universal (Public Domain)** license, allowing unrestricted reuse and modification.

3.2 Preprocessing

The preprocessing pipeline was implemented in Python using regular expressions to robustly clean and normalize tweet text while preserving sentiment cues. All text is first converted to lowercase. Mentions (e.g., @usernames) are replaced with the token `USR`, and URLs with `URL`. Hashtags are preserved as semantic tokens by replacing the symbol # with the prefix `hash_`. Additionally, a curated list of emoticons (e.g., `:-)`, `XD`) and emojis (e.g., 😊, 🔥, 😭) is kept intact to retain emotional context. A whitelist of characters ensures that only meaningful alphanumeric and sentiment-relevant symbols (e.g., `! ? ( ) : -`) are preserved, avoiding the loss of informal markers common in social media text. Finally, redundant spaces are removed, resulting in clean and normalized text ready for tokenization and embedding generation.

3.3 Feature Extraction

After preprocessing, the data was split with a stratified 80/20 train–test partition (`random_state=42`).

Statistical representations. For traditional ML models, we use a TF–IDF vectorizer with *max_features* = 50,000, n-grams $(1, 3)$, *min_df* = 2, and English stopwords. The resulting sparse matrix feeds Logistic Regression, Linear SVM, and Random Forest.

Semantic representations (word embeddings). For sequence models, we use **GloVe-Twitter-200** (via `gensim`). A Keras *Tokenizer* (*num_words*= 50,000, explicit `<OOV>`) builds the vocabulary; sequences are padded/truncated to $L = 180$. The embedding matrix is aligned to this vocab, with zero vector at the padding index and the `<OOV>` vector set to the mean of known embeddings.

Transformer representations. We employ **cardiffnlp/twitter-roberta-base-sentiment-latest** in two modes:

1. **Feature extractor:** we freeze the backbone and compute sentence embeddings with masked mean-pooling over the last hidden states (max length 96 tokens, batch size 64, `no_grad`). Train/test embeddings are cached to `.npy`.
2. **Fine-tuning:** we tokenize to 128 tokens and train a classification head end-to-end using `Trainer`.

3.4 Modeling

3.4.1 Traditional Machine Learning Models

Logistic Regression (TF–IDF). Multinomial Logistic Regression with `max_iter`= 2000, `C`= 1.0, and *class_weight*=`balanced`.
Linear SVM (TF–IDF). `LinearSVC` with `C`= 1.0 (hinge loss).
Random Forest (TF–IDF). `n_estimators`= 400, `max_depth=None`, `min_samples_split`= 2, *class_weight*=`balanced_subsample`, `n_jobs`= -1.
All models are trained on the TF–IDF features and evaluated with macro-averaged accuracy, precision, recall, and F1.

3.4.2 Deep Learning Models

CNN with Squeeze-and-Excitation and attention. `TextCNN_TwitterSE_Attn` architecture:

- **Input:** token indices ($L = 180$) with `SpatialDropout1D`(0.2).
- **Embedding:** GloVe-Twitter-200 matrix (frozen in Stage 1; unfrozen in Stage 2).
- **Trunk:** five `Conv1D` branches (128 filters; kernel sizes $\{2, 3, 4, 5, 7\}$), each followed by `BatchNorm` and `ReLU`.
- **SE-block:** channel-wise recalibration with reduction ratio $r = 8$.
- **Attention pooling:** per branch, we concatenate global max-pooling, global average-pooling, and a softmax-weighted sum across time.
- **Head:** concatenation → `Dropout`(0.4) → `Dense`(192, ReLU, L2=10^{-4}) → `Dropout`(0.4) → `Dense`(#classes, softmax).
- **Training:** `sparse_categorical_crossentropy`, `Adam`. *Stage 1*: `lr`= 10^{-3}, batch 128, val split 0.1, `EarlyStopping(patience=3)` + `ReduceLROnPlateau`(factor 0.5, patience 2, `min_lr`= 10^{-5}). *Stage 2*: unfreeze embeddings and add class weights, `lr`= 10^{-4}, `clipnorm`= 1.0. *Stage 3*: `CosineDecayRestarts` with initial `lr`= 5×10^{-5}, batch 64; `EarlyStopping` only.

BiLSTM with attention. `BiLSTM_Attn` architecture:

- **Input/Embedding:** same as CNN (GloVe embeddings, $L = 180$, `mask_zero`).
- **Trunk:** `SpatialDropout1D`(0.2) → `BiLSTM`(96, `return_sequences`) → `BiLSTM`(64, `return_sequences`); no recurrent dropout in our runs.
- **Attention pooling:** tanh projection → 1D score → temporal softmax; we concatenate attention-weighted sum with global max/avg pooling.
- **Head:** `Dropout`(0.4) → `Dense`(128, ReLU, L2=10^{-4}) → `Dropout`(0.4) → `Dense`(#classes, softmax).
- **Training:** three stages mirroring the CNN: Stage 1 `Adam` (`lr`= 10^{-3}); Stage 2 unfreezes embeddings and adds class weights with `Adam` (`lr`= 10^{-4}, `clipnorm`= 1.0); Stage 3 uses `CosineDecayRestarts` (`lr`= 5×10^{-5}).

Transformers (RoBERTa).

- **Feature extractor + heads:** with the backbone frozen, we compute mean-pooled embeddings (96 tokens). On top, we train (i) Logistic Regression (`max_iter`= 2000, `C`= 1.0) and (ii) LinearSVC (`C`= 1.0).
- **Fine-tuning:** `AutoModelForSequenceClassification` with explicit `id2label/label2id`; tokenization to 128 tokens. `TrainingArguments`: `learning_rate`= $2 \cdot 10^{-5}$, `per_device_train_batch_size`= 16, `per_device_eval_batch_size`= 32, `num_train_epochs`= 3, `weight_decay`= 0.01, `save_total_limit`= 1, `report_to=none`. We report macro accuracy, precision, recall, and F1.

Ensemble. We aggregate per-example predictions using two strategies: (i) **weighted voting** with fixed weights $\{w_{\text{RoBERTa}} = 0.90, w_{\text{BiLSTM}} = 0.76, w_{\text{CNN}} = 0.75\}$ and a tie-breaker that favors RoBERTa; and (ii) **majority voting** (three equal votes with the same tie-break rule). We report both and retain the one with the best macro F1.

4 Experiments and Results

4.1 Evaluation Metrics

We evaluate models with *accuracy*, *precision*, *recall*, and *F1-score*. Let $\{(x_i, y_i)\}_{i=1}^{N}$ be labeled examples with $y_i \in \mathcal{C} = \{\texttt{negative}, \texttt{neutral}, \texttt{positive}\}$ and predictions $\hat{y}_i$.

Accuracy.

$$\text{Acc} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}\{y_i = \hat{y}_i\}.$$

Per-class precision/recall/F1. For each class $c \in \mathcal{C}$, define true positives TP_c, false positives FP_c, and false negatives FN_c. Then

$$\text{Prec}_c = \frac{\text{TP}_c}{\text{TP}_c + \text{FP}_c}, \quad \text{Rec}_c = \frac{\text{TP}_c}{\text{TP}_c + \text{FN}_c}, \quad \text{F1}_c = \frac{2\,\text{Prec}_c\,\text{Rec}_c}{\text{Prec}_c + \text{Rec}_c}.$$

Macro-averaging. To avoid dominance by frequent classes, we report *macro* metrics:

$$\text{Prec}_{\text{macro}} = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} \text{Prec}_c, \quad \text{Rec}_{\text{macro}} = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} \text{Rec}_c, \quad \text{F1}_{\text{macro}} = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} \text{F1}_c.$$

All metrics are computed with `scikit-learn` using the label order `[negative, neutral, positive]` and `zero_division=0`. Confusion matrices follow the same class order.

4.2 Results Comparison

Model leaderboard. We compare traditional, deep, and transformer-based models on the test split. Table 1 summarizes macro metrics; Figure 1 visualizes macro-F1 across models.

Table 1: Model comparison on the test set. Macro-averaged metrics; higher is better.

Model	Accuracy	Precision	Recall	F1
RoBERTa fine-tuned (twitter-roberta-base-sentiment-latest)	0.897800	0.897500	0.901100	0.899000
RoBERTa feature extractor (LogReg head)	0.886900	0.889200	0.889000	0.889100
RoBERTa feature extractor (LinearSVC head)	0.884200	0.886700	0.886100	0.886300
Ensemble (Weighted Voting)	0.809000	0.806300	0.815200	0.809000
Ensemble (Majority Voting)	0.809000	0.806300	0.815200	0.809000
BiLSTM+Attention (GloVe-Twitter)	0.764100	0.762100	0.769300	0.764200
CNN (Twitter SE+Attn)	0.759200	0.757100	0.766900	0.759300
Logistic Regression (TF-IDF)	0.706800	0.705000	0.711900	0.707800
LinearSVC (TF-IDF)	0.698600	0.700100	0.699000	0.699400
RandomForest (TF-IDF)	0.677300	0.686900	0.671900	0.677400

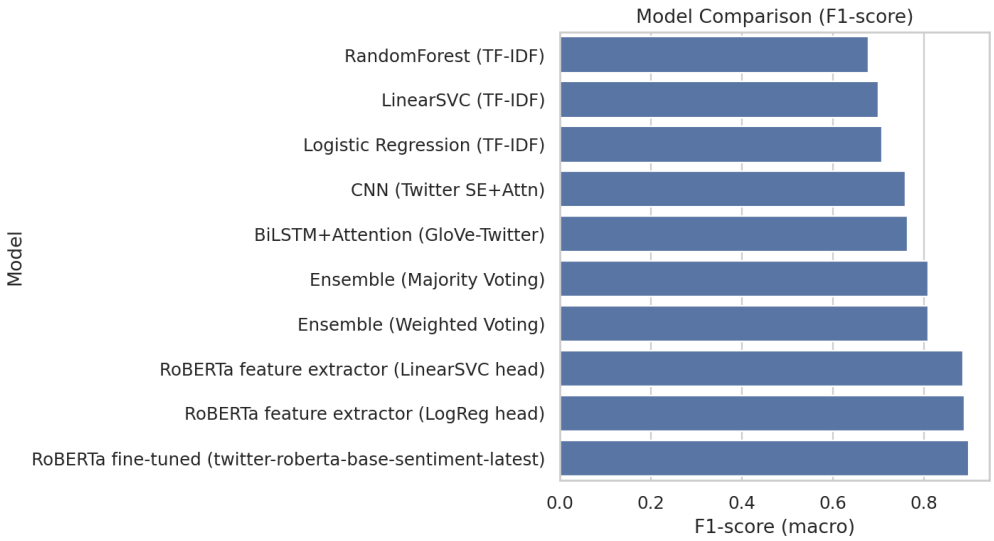

Figure 1: Macro-F1 across all models.

Per-class analysis. For the best model (top row in Table 1), we report its confusion matrix and per-class metrics:

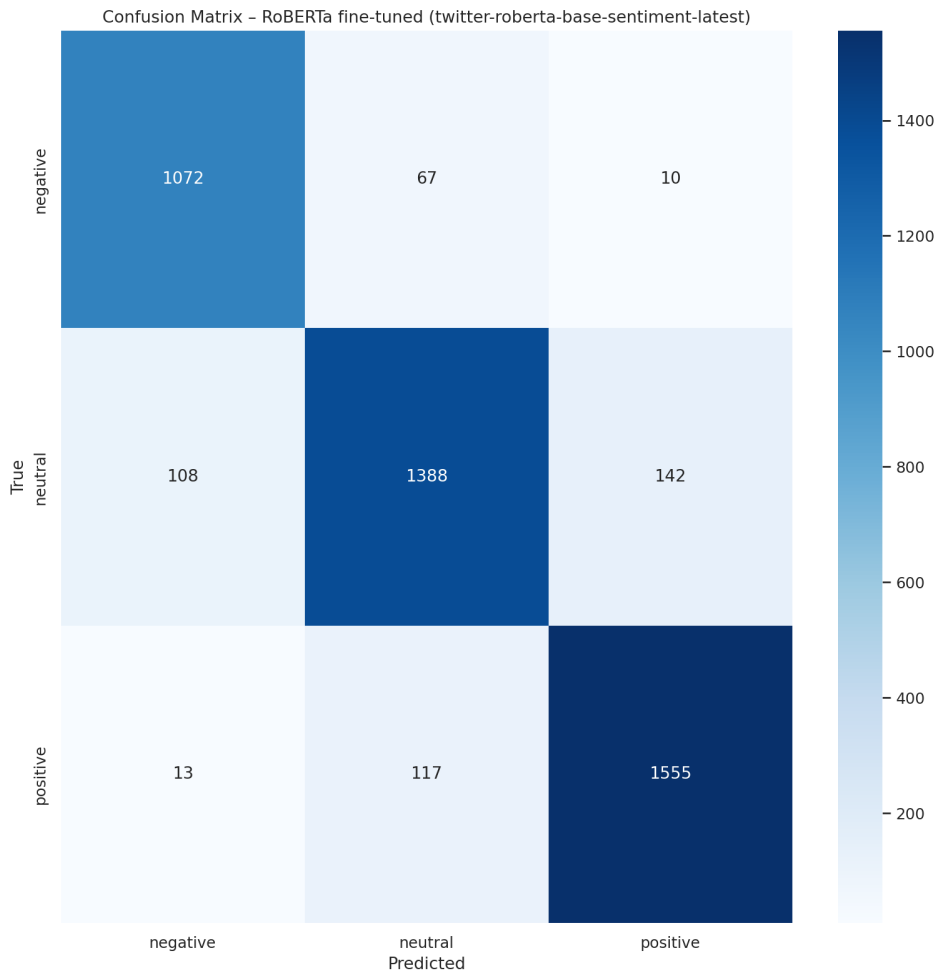

Figure 2: Confusion matrix for the best model (label order: negative, neutral, positive).

Table 2: Per-class report of the best model (precision, recall, F1, and support).

class	negative	neutral	positive	accuracy	macro avg	weighted avg
precision	0.898575	0.882952	0.910955	0.897809	0.897494	0.897517
recall	0.932985	0.847375	0.922849	0.897809	0.901070	0.897809
f1-score	0.915457	0.864798	0.916863	0.897809	0.899039	0.897431
support	1149.000000	1638.000000	1685.000000	0.897809	4472.000000	4472.000000

Probability-based curves. For models that output class probabilities (e.g., Logistic Regression, Random Forest, CNN, BiLSTM, RoBERTa heads and fine-tuned), we plot macro ROC and macro PR curves that are available in the Repository.
The figure below shows the macro ROC curve for the fine-tuned RoBERTa model.

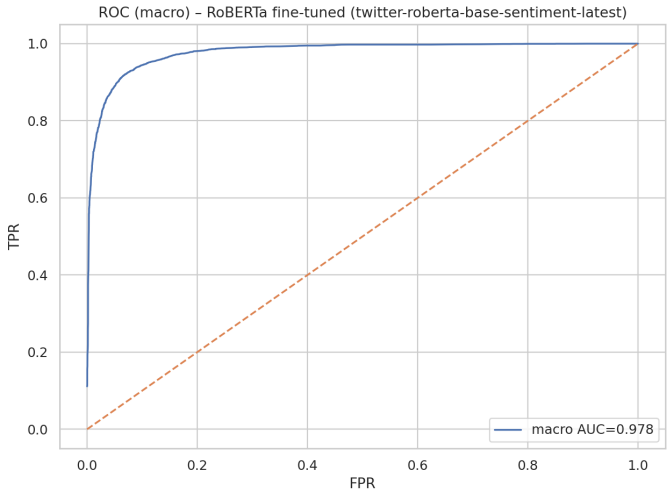

Figure 3: Macro ROC for a probability-capable model (example: fine-tuned RoBERTa).

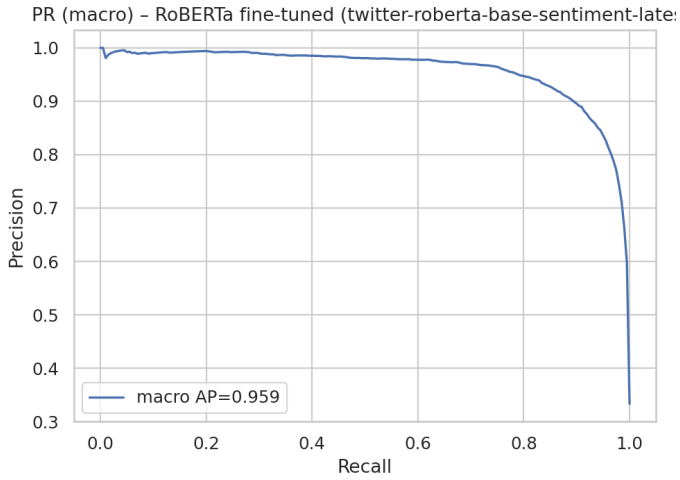

Figure 4: Macro PR for the same model as Figure 3.

The macro ROC curve hugs the top-left corner (AUC ≈ 0.98), reflecting strong class-averaged discrimination. Across reasonable thresholds, the model attains high true-positive rates at low false-positive rates for most classes.

Training dynamics. We report training/validation curves for neural models to show convergence and regularization behavior:

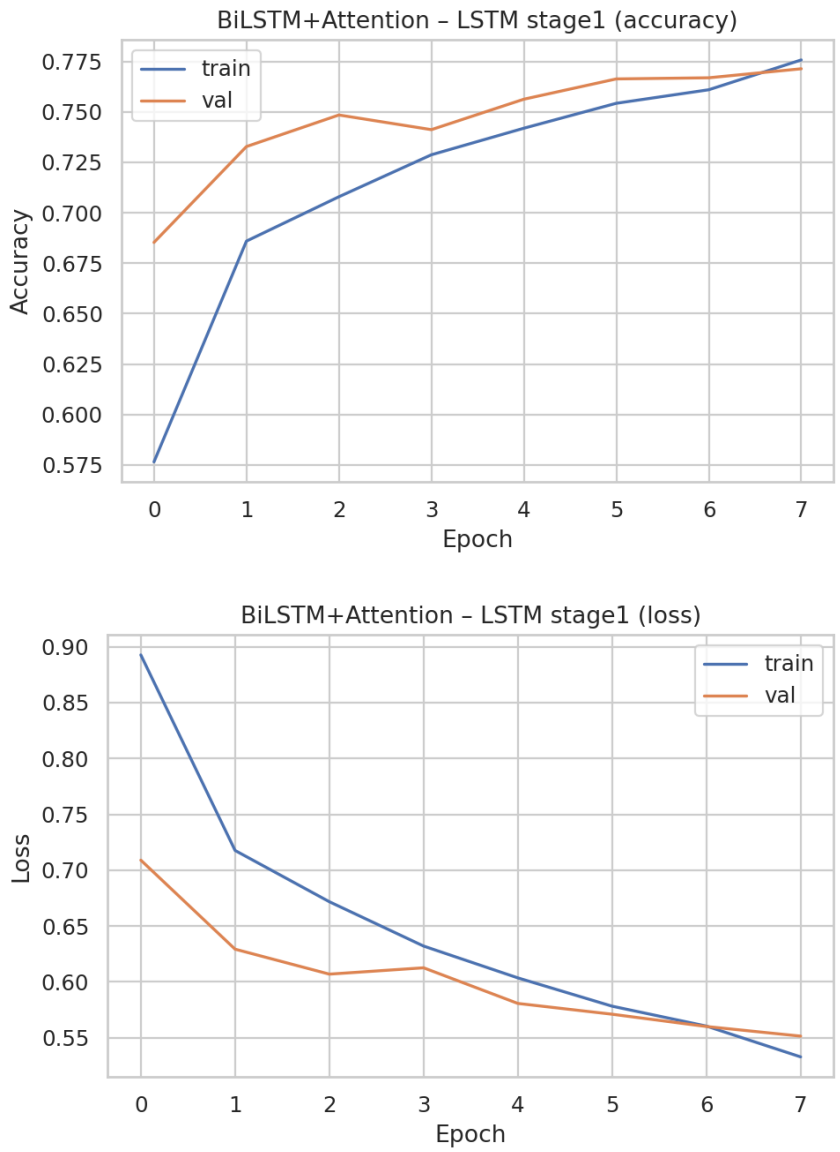

Figure 5: Example training curves (BiLSTM+Attention). Additional stages/architectures are in the Repository.

Stage-1 learning curves exhibit steady convergence: validation accuracy rises and validation loss falls over epochs. The train–val gap stays small, indicating regularization (dropout/L2) plus early stopping effectively limit overfitting.

Representation analysis When using RoBERTa as a frozen feature extractor, we visualize test-set embeddings with UMAP to inspect class separability:

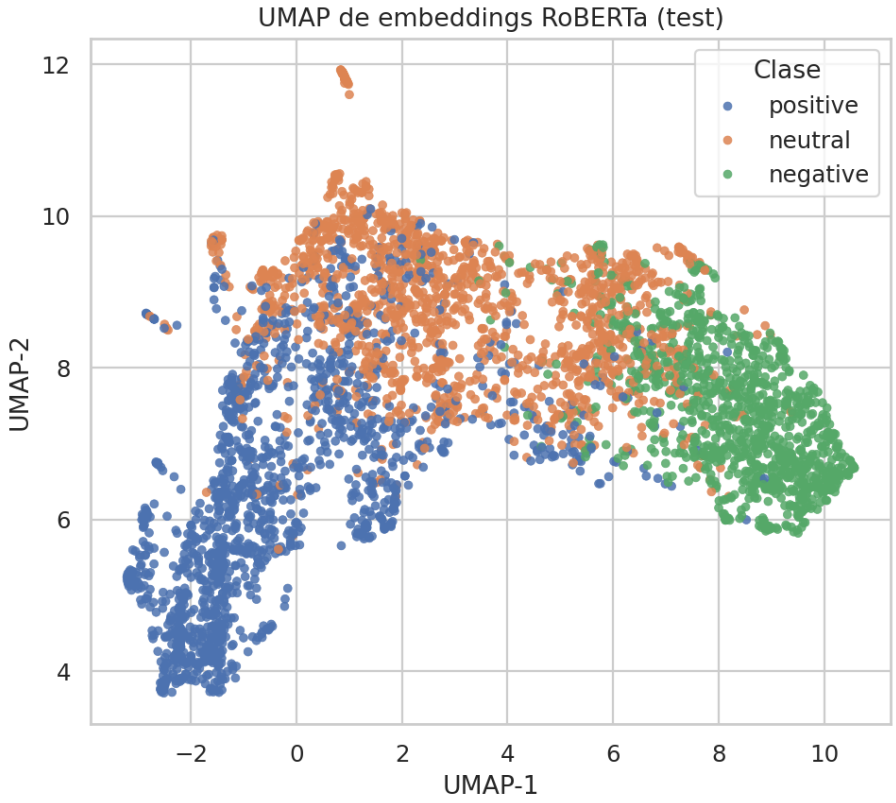

Figure 6: UMAP of mean-pooled RoBERTa embeddings (test split).

The UMAP of mean-pooled RoBERTa embeddings on the test set shows well-separated clusters for positive and negative, while neutral occupies a transitional region with more overlap. This suggests the transformer captures polarity clearly, with remaining ambiguity concentrated around the neutral boundary.

Traditional vs. deep vs. transformers. We observe three regimes:
(i) *Traditional* TF–IDF models are fast and lightweight but lag in macro-F1 (0.677–0.708), reflecting limited contextualization.
(ii) *Deep, non-transformer* models with GloVe (CNN / BiLSTM) recover part of the gap (0.759–0.764) yet still struggle on borderline *neutral* cases.
(iii) *Transformers* dominate: RoBERTa as a frozen feature extractor with a Logistic Regression head already reaches 0.889 macro-F1, while full fine-tuning yields 0.899 (+1.0 pp F1 and +1.2 pp macro-Recall), indicating that end-to-end adaptation mainly improves recall on ambiguous tweets.
Naïve voting ensembles (0.809) did not surpass the best single model; gains likely require calibrated stacking or error-aware weighting.
For transparency and reproducibility, we include per-model artifacts:

- Confusion matrices: `figures/*_cm.png`
- Per-class reports: `tables/classification_report__*.tex`

General findings. Fine-tuned **RoBERTa** achieved the best test performance, with $\mathrm{F1}_{\mathrm{macro}} = 0.899$ (Accuracy 0.898, Precision 0.898, Recall 0.901). Using RoBERTa as a frozen feature extractor + Logistic Regression head was a close second ($\mathrm{F1}_{\mathrm{macro}} = 0.889$), indicating that most of the gains come from pretrained Twitter-domain representations, while end-to-end fine-tuning adds a modest but consistent improvement ($\approx +1.0$ pp in $\mathrm{F1}_{\mathrm{macro}}$ and $+1.2$ pp in macro-Recall). Among non-transformer neural models,

BiLSTM+Attention (GloVe–Twitter) reached $F1_{macro} = 0.764$, slightly ahead of **CNN (SE+Attn)** at 0.759. Traditional TF–IDF baselines trailed behind (**Logistic Regression** 0.708, **LinearSVC** 0.699, **Random Forest** 0.677 macro-F1), underscoring the value of contextual representations for tweet sentiment. The two ensemble variants (weighted and majority voting) yielded 0.809 macro-F1, notably below the single best transformer models, suggesting that naive, fixed-weight voting did not exploit complementary errors; learned stacking/calibration might be required to help ensembles close the gap. Confusion matrices (best model vs. neural/classic baselines) illustrate that most residual errors concentrate around the *neutral* class boundaries, consistent with prior observations on short, informal texts.

5 Discussion

5.1 Analysis of Results

Discuss the overall findings from your experiments. Highlight how traditional models compared to deep learning ones and interpret why some performed better.

5.2 Strengths and Weaknesses

Describe what worked well in your approach (e.g., preprocessing pipeline, embeddings, model tuning) and what could be improved (e.g., dataset balance, training time, generalization).

5.3 Limitations

Acknowledge limitations of your dataset, models, or computational setup. For example, bias in tweets, lack of multilingual support, or restricted training epochs.

5.4 Future Work

Propose next steps — for instance, expanding to multilingual sentiment, fine-tuning larger transformers, or incorporating multimodal data (text + image).

5.5 Concluding Remarks

Briefly summarize the overall takeaway: what your results show and the main contributions of your project.

References

[1] Barnaghi, P., Ghaffari, P., & Breslin, J. (2016). Opinion mining and sentiment polarity on Twitter and correlation between events and sentiment. *Journal of Web Semantics*, 30, 1–11.

[2] Venkatesh, K., et al. (2019). Deep learning-based hybrid CNN-LSTM model for sentiment classification on sports tweets. *Procedia Computer Science*, 152, 341–348.

[3] Barbieri, F., Camacho-Collados, J., Neves, L., & Espinosa Anke, L. (2020). TweetEval: Unified benchmark and comparative evaluation for tweet classification. *Proceedings of EMNLP 2020*, 1644–1650.

[4] Fadel, A., et al. (2023). Fine-tuning RoBERTa for sentiment analysis of sports-related tweets. *IEEE Transactions on Affective Computing*, 14(2), 1–12.

[5] Barnaghi, P., Ghaffari, P., & Breslin, J. G. (2015). Text Analysis and Sentiment Polarity on FIFA World Cup 2014 Tweets. En *Proceedings of the KDD'15 Workshop on Large-Scale Sports Analytics*.

6 Appendix

6.1 GitHub Repository

https://github.com/enriquegomeztagle/MCD-NLP-SentimentAnalysisOfFIFATweets-FinalProject