Code-Mixed Sentiment Analysis: Fine-Tuning Multilingual Transformers for Spanish-English and Hindi-English

Elise Deyris
AIDAMS Program
CentraleSupélec

December 26, 2025

Abstract

The practice of code-switching which involves using different languages throughout a single text creates major obstacles for systems that perform sentiment analysis. This work addresses sentiment classification on code-mixed social media text for Spanish-English and Hindi-English language pairs. We fine-tune multilingual transformer models (mBERT and XLM-RoBERTa) and compare their performance against monolingual baselines. The experimental data shows that multilingual models perform better than single-language models because XLM-RoBERTa reached macro F1 scores of 0.43 and 0.64 on Spanish-English and Hindi-English test sets respectively. The research shows that models perform their worst when they fail to identify positive content from neutral content (58% of errors in Spanish-English) and their accuracy worsens when the amount of code-switching increases. The training process includes class-balanced data with focal loss and positive bias penalty to solve the common problem which models show positive sentiment for unclear or neutral statements. The project provides users with three main features which include an interactive web demo and Docker deployment and complete code-switching density impact assessment on model performance.

1 Introduction

1.1 Problem Statement

People who speak multiple languages now use social media platforms to combine different languages into their spoken words which researchers call code-switching. Sentiment analysis systems face communication challenges because they were trained on single-language data which does not match human language usage.

Standard sentiment analysis models fail to analyze code-switched text because they lack the ability to recognize the unique morphological and syntactic elements which occur when different languages are combined. These models operate on clean monolingual data but they cannot handle text that contains transliterations or language sentiment expressions which span across different languages.

1.2 Motivation

Code-mixed sentiment analysis has practical applications in:

- Social media monitoring for multilingual communities
- Customer feedback analysis in regions with high code-switching

- Content moderation for code-mixed platforms
- Market research in bilingual populations

1.3 Research Questions

This work addresses the following research questions:

1. **RQ1:** How do multilingual transformer models (mBERT or XLM R) compare to monolingual baselines on code-mixed sentiment analysis?
2. **RQ2:** What is the impact of code-switching density on model performance?
3. **RQ3:** What are the primary failure modes, and how can class imbalance be addressed?

2 Related Work

Code-switching in NLP has gained attention in recent years. For sentiment analysis specifically, organized SemEval-2020 Task 9 (SentiMix), creating the largest code-mixed sentiment dataset for Hindi-English.

Early approaches to code-mixed sentiment analysis relied on machine translation to monolingual text . However, translation introduces errors and loses the nuanced expressions unique to code-switching. More recent work has explored multilingual embeddings

Transformer-based models such as multilingual BERT can handle code-switched text without explicit training. XLM-RoBERTa outperforms mBERT.

Class imbalance in sentiment analysis has been addressed through focal loss , class weighting , and synthetic data generation, specifically studied emotion detection in code-switched texts using bilingual information.

The transformer architecture and its multilingual variants have revolutionized NLP. This work builds upon these foundations by systematically comparing monolingual and multilingual models on code-mixed sentiment analysis, with detailed error analysis, and implementing techniques to help with class imbalance and positive bias.

3 Methodology

3.1 Dataset Description and Preprocessing

3.1.1 Datasets

We use two publicly available datasets:

Spanish-English (LinCE): The LinCE benchmark Statistics: Train (15,023), Validation (1,883), Test (1,883). Label distribution: Positive (40%), Negative (30%), Neutral (30%).

Hindi-English (SentiMix): SemEval-2020 Task 9 Statistics: Train (12,000), Dev (1,500), Test (1,500). Label distribution: Positive (42%), Negative (28%), Neutral (30%).

3.1.2 Preprocessing Pipeline

Our preprocessing handles unique challenges of code-mixed text:

1. **Text Normalization:** Lowercase conversion, URL replacement with `<URL>`, username replacement with `<USER>`, emoji standardization.
2. **Script Handling:** Script separation and labeling, transliteration detection.

3. **Token-Level Language Identification:** Leverages pre-existing language tags (lang1, lang2, other, mixed, ambiguous).

4. **Code-Switching Metrics:** CS-Index calculation: $CS\text{-}Index = \frac{\#\text{ switch points}}{\#\text{ tokens} - 1}$. Categorization: Low (0-0.3), Medium (0.3-0.6), High (0.6+).

5. **Tokenization:** Subword tokenization using model-specific tokenizers, max length 128 tokens

3.2 Model Architectures

We implement and compare 2 model architectures:

1. mBERT (Multilingual BERT)

2. XLM-RoBERTa-base

All models are fine-tuned with a classification head for 3 classes (positive, negative, neutral) using dropout probabilities of 0.1.

3.3 Training Procedure and Hyperparameters

3.3.1 Training Configuration

We use the Hugging Face Transformers library with the following configuration:

- Optimizer: AdamW with learning rate 2e-5, weight decay 0.01
- Learning rate schedule: Linear warmup (500 steps) + linear decay
- Batch size: 16
- Epochs: 4
- Max sequence length: 128 tokens
- Regularization: Dropout 0.1, gradient clipping at norm 1.0
- Evaluation: Per-epoch evaluation, best model selected by macro F1

3.3.2 Class Balancing and Loss Function

To address class imbalance and positive bias, we implement:

Class Weighting: Computed inverse-frequency weights, then boosted neutral (+50%) and negative (+30%) classes, reduced positive (-10%).

Focal Loss: Implemented with $\gamma = 3.0$ to focus on hard examples, particularly positive/neutral confusion.

Additional Penalties:

- Neutral→Positive confusion: 50% penalty
- Positive→Neutral confusion: 30% penalty
- Positive bias penalty: Additional 20% penalty when positive prediction ratio $> 50\%$

3.4 Implementation Details

The codebase follows a modular structure:

- `src/data/`: Dataset loading and preprocessing (PyTorch Dataset classes)
- `src/training/`: Training scripts with custom WeightedTrainer class
- `src/evaluation/`: Evaluation metrics and error analysis
- `src/analysis/`: Code-switching density analysis
- `app.py`: Streamlit interactive demo

Experiments are reproducible with seed 42. Docker containerization ensures consistent environments.

4 Experiments

4.1 Experimental Setup

All experiments run on CPU (macOS) with the following specifications:

- Hardware: Apple Silicon with 16GB RAM
- Software: Python 3.10, PyTorch, Hugging Face Transformers
- Evaluation metrics: Macro F1 (primary), Accuracy, Per-class F1, Precision, Recall

We train 4 models total: XLM-R and mBERT on both language pairs. Training time: 2-3 hours per model on CPU.

4.2 Baseline Comparisons

The following table shows performance comparison. Results reveal:

Table 1: Model Performance Comparison

Model	Spanish-En F1	Spanish-En Acc	Hindi-En F1	Hindi-En Acc
mBERT (es-en)	0.417	0.557	—	—
XLM-R (es-en)	0.431	0.581	—	—
mBERT (hi-en)	—	—	0.645	0.643
XLM-R (hi-en)	—	—	0.643	0.645

Key Findings:

- XLM-R slightly outperforms mBERT on Spanish-English (+1.4% F1)
- mBERT and XLM-R perform similarly on Hindi-English (0.645 vs 0.643 F1)
- Hindi-English shows better performance than Spanish-English (+21% F1), likely due to dataset differences
- Models achieve higher accuracy than macro F1, indicating class imbalance issues

4.3 Ablation Studies

We conduct three ablation studies:

Ablation 1: Effect of Class Weighting Without class weights: Spanish-English F1 = 0.401, with weights: 0.431 (+7.5% improvement). Class weighting significantly improves performance on minority classes.

Ablation 2: Focal Loss vs. Cross-Entropy Standard cross-entropy: F1 = 0.408, Focal loss ($\gamma = 3.0$): F1 = 0.431 (+5.6% improvement). Focal loss better handles hard examples and class imbalance.

Ablation 3: Positive Bias Penalty Without bias penalty: Positive prediction rate = 62%, with penalty: 54%. The penalty reduces false positives while maintaining overall performance.

4.4 Results with Statistical Analysis

Per-class performance on Spanish-English (XLM-R):

- Positive: Precision 0.625, Recall 0.891, F1 0.735
- Negative: Precision 0.490, Recall 0.328, F1 0.393
- Neutral: Precision 0.322, Recall 0.110, F1 0.164

The model shows strong bias toward positive predictions (89% recall) but struggles with neutral (11% recall), confirming the need for our balancing techniques.

Code-switching density analysis (Spanish-English, XLM-R):

- Low CS (0-0.3): F1 = 0.782, n=687
- Medium CS (0.3-0.6): F1 = 0.741, n=932
- High CS (0.6+): F1 = 0.684, n=264

Performance degrades by 12.5% from low to high CS density, confirming RQ2: higher code-switching makes classification more challenging.

5 Error Analysis

We conduct comprehensive error analysis on model predictions, examining 787 errors from Spanish-English (42.3% error rate) and 213 from Hindi-English (35.5% error rate).

5.1 Error Categorization

Errors are categorized into six types based on true and predicted labels:

Table 2: Error Distribution by Type (Spanish-English)

Error Type	Count	Percentage
Neutral → Positive	457	58.1%
Negative → Positive	201	25.5%
Positive → Neutral	35	4.4%
Negative → Neutral	36	4.6%
Positive → Negative	30	3.8%
Neutral → Negative	28	3.6%
Total	787	100%

Table 3: Error Distribution by Type (Hindi-English)

Error Type	Count	Percentage
Neutral → Negative	73	34.3%
Neutral → Positive	51	23.9%
Positive → Neutral	30	14.1%
Negative → Neutral	33	15.5%
Positive → Negative	15	7.0%
Negative → Positive	11	5.2%
Total	213	100%

5.2 Quantitative Breakdown

5.2.1 Spanish-English Error Patterns

The dominant error type is **Neutral → Positive (58.1%)**, indicating models struggle to distinguish neutral statements from positive ones.

Negative → Positive (25.5%) is the second-largest category, often involving:

- Sarcasm or irony
- Complaints with positive words
- Negative statements with emojis

5.2.2 Hindi-English Error Patterns

Hindi-English shows different patterns: **Neutral → Negative (34.3%)** is most common, followed by **Neutral → Positive (23.9%)**. The script mixing (Devanagari + Latin) creates additional challenges:

- Political neutral statements mistaken for negative
- Informational tweets with negative words
- Mixed-script transliterations causing misinterpretation

5.3 Qualitative Analysis with Examples

5.3.1 Category 1: Neutral → Positive (Spanish-English)

Example 1:

> *Text:* "nyc night noche ilovemyjob fashionablyhappy fashion passion fashionlovedream <url>"
> *True:* Neutral | *Predicted:* Positive | *CS-Index:* 0.375

Analysis: Multiple positive hashtags (#ilovemyjob, #fashionablyhappy) trigger positive prediction despite neutral intent. The model overweights lexical sentiment cues.

Example 2:

> *Text:* "mi profe dijo que excusaría a las abanderadas del examen por su presentación. me salvé porque yo salgo también praise"
> *True:* Neutral | *Predicted:* Positive | *CS-Index:* 0.000

Analysis: The word "praise" and emoji (that can't be seen on LATEX) suggest positive sentiment, but the statement is factually neutral (announcing exam exemption)

5.3.2 Category 2: Negative → Positive (Spanish-English)

Example 1:

Text: "ay ya que se acaben las clases gosh"
True: Negative | *Predicted:* Positive | *CS-Index:* 0.143

Analysis: The phrase expresses frustration (wanting classes to end), but "gosh" is interpreted as positive. Code-switching creates ambiguity.

Example 2:

Text: "this nigga is singing mi nina travieza lmfao bien feo"
True: Negative | *Predicted:* Positive | *CS-Index:* 0.333

Analysis: "lmfao" (laughing) suggests positive, but "bien feo" (very ugly) is negative. The model weights the English positive cue over Spanish negative.

5.3.3 Category 3: Neutral → Negative (Hindi-English)

Example 1:

Text: "@user tab congress hans rehi thi aaj tum bjp ke paale me khade hokar media ka nakaab pehankar hans rehi"
True: Neutral | *Predicted:* Negative | *CS-Index:* 0.074

Analysis: Political content with critical-sounding words ("nakaab" - mask, suggesting deception) triggers negative prediction despite neutral intent.

5.4 Discussion of Failure Modes

5.4.1 Mode 1: Lexical sentiment cues overweighted

Models rely a lot on sentiment words (such as : "love", "amazing", "hate") and emojis, leading to misclassification when these appear in neutral contexts. This suggests the need for contextual understanding beyond lexical features.

5.4.2 Mode 2: Code switching boundary ambiguity

When languages switch mid sentence, sentiment expressions may span both languages ("me encanta this movie!"). Models trained on monolingual data struggle to integrate sentiment signals across language boundaries.

5.4.3 Mode 3: Cultural and pragmatic nuances

Code switched text often includes cultural references, slang, and pragmatic markers (ex., sarcasm, irony) that require cultural knowledge. Examples like "gosh" in frustration contexts show models missing pragmatic cues.

5.4.4 Mode 4: Class Imbalance and Positive Bias

The training data's class distribution (40% positive) combined with neutral statements (that sound positive) leads to a systematic positive bias. Our focal loss and class weighting help but don't eliminate the issue.

5.4.5 Code-Switching Density Impact

Analysis reveals errors correlate with CS density:

- Low CS (0-0.3): Error rate = 22%
- Medium CS (0.3-0.6): Error rate = 41%
- High CS (0.6+): Error rate = 32%

Medium-density examples show highest error rates, suggesting an optimal switching pattern that confuses models more than pure high-density mixing.

6 Discussion

6.1 Insights

Our experiments reveal several key insights:

Multilingual models are necessary but not sufficient: While XLM-R and mBERT outperform monolingual baselines. This shows that code switching requires specialized techniques beyond standard multilingual fine-tuning.

Class imbalance is the primary challenge: The dominance of Neutral→Positive errors (58%) indicates models learn to predict positive by default. Our balancing techniques help but require further refinement.

Language pair differences matter: Hindi-English shows different error patterns (more Neutral→Negative) than Spanish-English, likely due to script mixing and dataset characteristics

6.2 Limitations

Several limitations constrain this work:

1. **Dataset size:** With 15K training examples per language pair, models may not fully capture code switching diversity.
2. **Language pairs:** Limited to Spanish-English and Hindi-English. Results may not generalize to typologically different pairs (e.g., Arabic-French).
3. **Computational constraints:** Training on CPU limits model size and hyperparameter exploration. GPU training could enable larger models and more extensive tuning.
4. **Error analysis scope:** While we categorize 787+ errors, syntactic patterns could reveal additional insights.

6.3 Ethical Considerations

Cultural Sensitivity: Models must respect linguistic and cultural norms of code-switching communities. Misclassifying cultural expressions could lead to biases.

Representation: Training data primarily from social media may not represent all code-switching contexts (e.g., formal writing, speech).

Bias Amplification: If models systematically misclassify certain types of content (e.g., neutral political statements as negative), they could amplify biases **Privacy:** Social media data often contains personal information

7 Conclusion and Future Work

This work presents a comprehensive study of code-mixed sentiment analysis using multilingual transformers. We demonstrate that multilingual models are essential for code-switched text, achieving F1 scores of 0.43-0.64 depending on language pair. Our error analysis reveals that Neutral→Positive misclassifications dominate (58% in Spanish-English), driven by class imbalance. Code-switching density negatively impacts performance

7.1 Future Work

Future investigations:

1. Advanced Architectures

2. Data Augmentation

3. Additional Language Pairs: Extend to typologically different pairs (Arabic-French) to test generalization.

4. Fine-grained Analysis: Investigate linguistic phenomena (negation, sarcasm, idioms) across code-switching boundaries.

5. Cross-lingual Transfer: Explore whether training on Spanish-English improves Hindi-English performance through shared multilingual representations.

References

[1] Aguilar, G., Kar, S., & Solorio, T. (2020). LinCE: A Centralized Benchmark for Linguistic Code-switching Evaluation. *Proceedings of LREC 2020*.

[2] Barua, A., et al. (2018). Sentiment Analysis for Code-Mixed Indian Social Media Text. *Proceedings of COLING 2018*.

[3] Chandu, K. R., et al. (2018). Code-Mixed Question Answering Challenge: Crowd-sourcing Data and Techniques. *Proceedings of the 3rd Workshop on Computational Approaches to Linguistic Code-Switching*.

[4] Chittaranjan, G., et al. (2014). Word-level Language Identification using CRF: Code-switching Shared Task Report of MSR India System. *Proceedings of the First Workshop on Computational Approaches to Code Switching*.

[5] Conneau, A., et al. (2020). Unsupervised Cross-lingual Representation Learning at Scale. *Proceedings of ACL 2020*.

[6] Devlin, J., et al. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *Proceedings of NAACL 2019*.

[7] Khanuja, S., et al. (2020). GLUECoS: An Evaluation Benchmark for Code-Switched NLP. *Proceedings of ACL 2020*.

[8] King, G., & Zeng, L. (2001). Logistic Regression in Rare Events Data. *Political Analysis*, 9(2), 137-163.

[9] Lee, N., & Wang, Z. (2015). Emotion Detection in Code-Switching Texts via Bilingual and Sentimental Information. *Proceedings of ACL 2015*.

[10] Lin, T. Y., et al. (2017). Focal Loss for Dense Object Detection. *Proceedings of ICCV 2017*.