Evaluating the Impact of Language Identification on NER: Spanish-English Code-Switched Tweets

Natural Language Processing and Deep Learning - BSNLPDL2KU

IT University of Copenhagen

Etele Kovács Julia Otto Laura Cabral Costa Ascensão Tiago Luca Costa Martins

etko@itu.dk juot@itu.dk lcab@itu.dk lucm@itu.dk

Abstract

Named Entity Recognition (NER) in code-switched text is challenging because models must identify entity boundaries while processing tokens from multiple languages within the same sentence. This project studies Spanish-English code-switched tweets from the LINCE benchmark and evaluates two multilingual transformer backbones, mBERT and XLM-RoBERTa. In addition to standard fine-tuning, we introduce a language-aware classification layer that uses token-level Language Identification (LID) annotations to adjust NER predictions according to the language of each token. Rather than changing the transformer encoder itself, the method adds a lightweight entity-type-specific bias to the output logits. Our results show that this explicit language signal improves overall performance, with XLM-RoBERTa + LID achieving the best F1-score of 0.6487. The gains are mainly driven by improved precision, suggesting that language-aware bias helps reduce false entity predictions in code-switched contexts. Code available at: https://github.com/Etele8/NLP-underachievers.

1 Introduction

Named Entity Recognition (NER) is a fundamental NLP task that identifies specific entities such as persons, locations, and organizations within a text. While transformer-based models have achieved high performance on monolingual data, they face challenges when processing code-switched text.

We utilize the pre-annotated LINCE benchmark to compare established multilingual models against a novel architectural variant designed to handle linguistic interference.

1.1 Research question

This study investigates the research question:

> "How do multilingual transformer models perform on NER for Spanish-English code-switched tweets, and can entity type-specific language bias improve their predictive accuracy?"

2 Related Work

2.1 Multilingual Transformers

To handle multilingual datasets, models such as mBERT and XLM-RoBERTa utilize shared subword tokenization to map multiple languages into a single vector space. However, because these models are primarily pre-trained on formal, monolingual corpora like Wikipedia, they might struggle to represent the mixed linguistic patterns commonly found in social media text (Bhat et al., 2021).

2.2 Previous Work on Code-Switched NER

Early approaches to code-switched NER favored robust character-level features to bypass complex linguistic rules (Aguilar et al., 2018). While the introduction of the LINCE benchmark (Aguilar et al., 2020) standardized evaluation, recent studies show that large multilingual models still suffer from "language interference" on code-switched data (Winata et al., 2021). Current state-of-the-art solutions often require computationally expensive secondary pre-training on translation pairs (Bhat et al., 2021).

Compared to previous approaches, our work introduces a lightweight architectural solution: entity type-specific language bias. We incorporate token-level language information into the classification layer, effectively reducing prediction ambiguity near code-switch points.

3 Data Exploration

This section describes the Spanish-English code-switched NER dataset from the LINCE benchmark (TheDevastator, 2021), which utilizes the IOB2 tagging scheme on Twitter-based text.

3.1 Dataset Statistics and Linguistic Composition

Table 1 summarizes the dataset size and sequence lengths. The partitions are consistent, with an average of 12 tokens per example, reflecting the shortness of social media posts.

Split	Examples	Avg Tokens	Max Tokens
Train	33,611	12.03	72
Validation	10,085	12.16	45
Test	23,527	11.97	72

Table 1: General dataset statistics across splits.

As shown in Table 2, the dataset is linguistically diverse. Spanish (Lang 2) is the majority language (≈50%), while English (Lang 1) accounts for ≈20%. Named Entity (NE) tokens are sparse (2.4%), creating a high-imbalance environment. This linguistic mix, where entities often appear at language boundaries, justifies our architectural focus on using language-specific bias to resolve prediction ambiguity.

Split	Lang 1 (%)	Lang 2 (%)	NE (%)
Train	20.30	49.78	2.37
Validation	23.63	45.18	2.37
Test	18.74	51.91	2.41

Table 2: Linguistic distribution (Lang 1: English, Lang 2: Spanish).

The entity distribution in Figure 1 in the Appendix shows that `PERSON` tags are most frequent, followed by `LOCATION`. While the `O` (non-entity) class is the dominant token-level label, it is omitted from the figure to emphasize entity-specific diversity.

3.2 Data Cleaning and Masking

The LINCE dataset stores tweets and their NER tags as stringified lists within CSV cells (`['token1', 'token2']`). To handle this, we implemented a custom parser to safely extract the tokens. The pipeline then performs strict IOB2 validation on every sequence, identifying and flagging invalid tag transitions (e.g., an `I-` tag appearing without a corresponding `B-` tag) to ensure the models train on stable data.

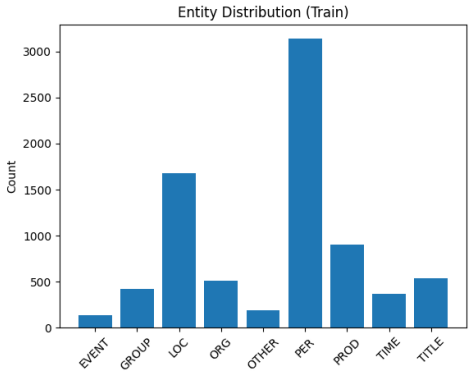

Figure 1: Entity type frequency in the training set.

To prepare the data for training, we implement two levels of masking:

- **Target Masking:** The test set contains blank labels used for evaluation-only inference. Our data loader dynamically detects the `test` split and masks these targets, bypassing label validation so the system can generate predictions without throwing errors.

- **Sub-word Masking:** Because models like mBERT and XLM-RoBERTa split words into multiple sub-tokens, we align the original ground-truth label strictly to the *first* sub-token of a word. All subsequent sub-tokens and special padding tokens are assigned a masking index of `-100`. This ensures the PyTorch Cross-Entropy loss function correctly ignores these fragments during backpropagation.

3.3 Target Variable

The target variable in this project is the Named Entity Recognition (NER) label assigned to each token in a sentence. These labels follow the IOB2 tagging scheme, where tags such as `B-PER`, `I-PER`, `B-LOC`, and `O` indicate whether a token belongs to a named entity and its position within that entity. During preprocessing, all entity labels are automatically extracted from the dataset.

As is common in NER tasks, the dataset is highly imbalanced because the majority of tokens are labeled as `O` (outside any entity).

4 Methodology

4.1 Baseline Models

We utilize two multilingual models: **mBERT** (*bert-base-multilingual-cased*) and **XLM-RoBERTa** (*xlm-roberta-base*). While both models use shared sub-word tokenization across multiple languages, they differ significantly in their pre-training data and scale. mBERT is pre-trained using masked language modeling (MLM) on Wikipedia text across 104 languages, meaning its representations are shaped by formal, encyclopedia-style writing. XLM-RoBERTa, by contrast, is pre-trained on CC-100, a significantly larger and more diverse web-crawled corpus spanning 100 languages, and applies a more robust training regime with larger batch sizes and longer training. This makes XLM-R generally stronger on informal and out-of-domain text.

4.2 Entity Type-Specific Language Bias

Code-switched NER is difficult because the same surface form or context can have different interpretations depending on the language being used. For example, a token near a Spanish-English switch may be incorrectly included in an entity span if the model relies only on multilingual contextual embeddings. To address this, we add token-level language information directly to the classification decision.

The LINCE dataset provides a Language Identification (LID) gold label for each token. These labels indicate whether a token belongs to Spanish, English, or another language-related category. During preprocessing, LID labels are aligned with the same subword positions as the NER labels.

Our language-aware model keeps the transformer encoder unchanged. First, the encoder produces a contextual representation for each token, and the standard classifier produces NER logits. Then, the model computes an additional language-dependent adjustment. Each LID label is represented with a small trainable language embedding. A gating network uses the token representation and its language embedding to decide how strongly language information should affect each entity type. The model also learns a bias table that captures associations between entity types and languages. The final NER logits are obtained by adding this gated language bias to the standard classifier logits.

This design is lightweight because it does not require additional pre-training or a second transformer model. It only modifies the classification layer, allowing the model to learn patterns such as whether certain entity types are more likely to appear in Spanish, English, or mixed-language contexts.

4.3 Training Objective

The models are trained as sequence labeling systems using cross-entropy loss over BIO NER tags. The loss is computed only for the first subword of each token, while special tokens, padding, and continuation subwords are ignored using the masking value `-100`. This ensures that each original word contributes only once to training, even when split into multiple subwords.

For the language-bias models, the same NER loss is used. LID labels are not predicted as a separate task; instead, they are provided as input features that condition the NER predictions.

5 Experimental Setup

5.1 Hyperparameters

All models were fine-tuned using the **AdamW** optimizer (PyTorch Contributors, 2024) together with a linear learning rate scheduler. To ensure consistency across experiments, we used the same baseline training configuration for both models.

The baseline configuration used a learning rate of 3×10^{-5}, a batch size of 8, a weight decay of 0.01, and 5 training epochs. We also used a maximum sequence length of 128 tokens, which covered the majority of tweets in the dataset without excessive padding.

To evaluate how different training settings affected performance, we varied one hyperparameter at a time while keeping all remaining parameters fixed at their baseline values. The explored values are summarized in Table 4 in the Appendix.

This setup allowed us to observe how individual training parameters influenced model performance on code-switched NER.

5.2 Evaluation Metrics

Following standard practices for NER, we evaluated our model's performance using Precision, Recall, and the F1-score. Given the sensitivity of entity boundaries in code-switched text, we utilized strict matching criteria (using the `seqeval` framework). This requires that both the entity type and the exact token boundaries (start and end)

match the ground truth for a prediction to be considered correct.

6 Results

This section presents the empirical results of our experiments, comparing the baseline multilingual models against our language-bias-aware architecture.

6.1 Overall Performance

The baseline performance for standard full fine-tuning is summarized in Table 3. XLM-RoBERTa (XLM-R) outperformed mBERT in the baseline configuration, achieving a higher F1-score (0.6358 vs 0.6198) and significantly better recall. The introduction of the Entity Type-Specific Language Bias module resulted in an uplift across both backbones. The **XLM-R + Bias** model achieved the best overall performance with a Micro F1 of 0.6487. We report Micro F1, which computes the F1-score globally across all entity tokens rather than averaging per entity type, making it more representative of overall performance on imbalanced datasets where some entity classes appear far more frequently than others.

Model	Precision	Recall	F1-Score	Accuracy
mBERT (Baseline)	0.6261	0.6136	0.6198	0.9844
mBERT + Bias	**0.6574**	0.6102	0.6329	0.9851
XLM-R (Baseline)	0.6352	0.6365	0.6358	0.9843
XLM-R + Bias	0.6541	**0.6433**	**0.6487**	**0.9853**

Table 3: Overall performance comparison between baseline and language bias models.

6.2 Per-Entity Results

Table 5 in the Appendix provides a breakdown of F1-scores across all nine entity categories. High-support categories like `PER` and `LOC` show the most stability, whereas low-support categories like `EVENT` and `TIME` exhibit higher variance across models.

7 Analysis

In this section, we interpret our results by looking at the model's training behavior.

7.1 LID Integration Impact

We analyzed the effect of adding the Language Identification (LID) module to our backbones. As shown in Table 3, this addition primarily improved the model's **Precision**.

As visualized in the Impact Chart (Figure 2), the LID module provided a positive performance boost for the majority of entity types. By giving the model explicit information about which language is being used, it becomes better at reducing incorrect predictions. It allows the model to adjust entity predictions using explicit language information, leading to a significant reduction in false positives.

7.2 LID Integration Impact

Adding token-level LID information improved both mBERT and XLM-R, with the strongest result obtained by XLM-R + LID. The improvement is most visible in precision: the language-aware models make fewer incorrect entity predictions. This suggests that LID information helps the classifier decide when a token should not be included in an entity span.

The effect is consistent with the design of the bias module. Since the language signal is added at the classification stage, it does not replace the contextual knowledge learned by the transformer. Instead, it acts as an additional decision signal that helps the model calibrate entity predictions according to the token's language identity. This is especially useful in code-switched text, where entity boundaries often occur near language changes.

7.3 Quantitative Analysis: What the model confuses

To understand the model's mistakes using data, we look at the **Confusion Matrix** in Figure 4. This counts how often the model mistakes one label for another.

Mistaking People for Organizations: There is a common pattern where the model confuses `PER` (Person) and `ORG` (Organization). This happens because many names are used for both (for example, a band or a company named after a person).

Predicting "Nothing": Many errors occur when the model labels an entity as `O` (Outside/Nothing). This is most common in categories like `EVENT`, which the model rarely saw during training. This shows the model tends to favor the majority class for infrequent entities.

7.4 Learning Curves: Training Stability

We use **Learning Curves** to see how the model learned with each epoch (Figure 3) . By compar-

ing the baseline to the version with the LID module, we see two clear trends:

- **Faster Learning:** The models with the LID module reach their best performance faster (usually by the 3rd epoch) compared to the baselines.
- **Smoother Progress:** The loss curve is much smoother with the LID module. This leads to more stable optimization during training.

7.5 Qualitative Analysis: Looking at Actual Tweets

To understand the reasoning behind the errors, we manually read specific tweets where the model failed.

Switch-Point Confusion: Errors often happen exactly when the language switches from Spanish to English. For example, in *"en el Center Park"*, the model may incorrectly include surrounding function words as part of an entity span near language switch-points.

Slang and Formatting: Tweets often lack capital letters or use slang (like *"nocheee"*). Without standard grammar rules, the model sometimes struggles to tell the difference between a product name (`PROD`) and a regular word.

7.6 Support vs. Performance

Finally, we compare the model's score to how much data it was given (Support). As shown in Figure 5, there is a direct link: categories with a lot of examples (`PER`, `LOC`) have high scores, while rare categories (`EVENT`) have low scores. This confirms that while our LID module helps, the model still needs more examples of rare entities to improve performance on low-resource entity categories.

8 Limitations and Conclusion

8.1 Limitations

Despite the performance gains, several limitations remain. First, the model's accuracy is heavily dependent on the quality of the **Language Identification (LID) labels**. If the LID prediction is noisy or incorrect, the bias head may provide a misleading signal, potentially degrading NER accuracy. Second, **class imbalance** continues to be a challenge; rare entity types such as `EVENT` and `TIME` showed lower improvement because the model had fewer examples to learn their specific linguistic biases. Finally, this approach was tested exclusively on **Twitter data**; social media's informal nature (lack of capitalization, heavy slang) means these results might not generalize to formal code-switched documents without further fine-tuning.

8.2 Conclusion

This project demonstrates that multilingual transformer models can be modestly improved for code-switched NER by introducing a lightweight **entity type-specific language bias**. Our experiments show that adding language-aware bias signals improved performance near linguistic switch-points at linguistic switch-points where standard models typically fail. Our best-performing model, **XLM-RoBERTa + LID**, achieved a peak F1-score of **0.6487**, driven largely by a boost in Precision. This suggests that explicit language information can be useful for named entity recognition in code-switched text. Future work could involve joint multi-task learning, where the model learns to predict Language ID and NER tags simultaneously to reduce dependency on pre-annotated LID labels.

References

[Aguilar et al.2018] Gustavo Aguilar, Thamar Solorio, and Mona Diab. 2018. Simple features for strong performance on named entity recognition in code-switched twitter data. https://aclanthology.org/W18-3213/. Proceedings of the Workshop on Computational Approaches to Linguistic Code-Switching.

[Aguilar et al.2020] Gustavo Aguilar, Sudipta Kar, and Thamar Solorio. 2020. LINCE: A centralized benchmark for linguistic code-switching evaluation. https://aclanthology.org/2020.lrec-1.565/. Proceedings of the 12th Language Resources and Evaluation Conference (LREC 2020).

[Bhat et al.2021] Riyaz Bhat, Genta Indra Winata, and Pascale Fung. 2021. Improved multilingual language model pretraining for social media text via translation pair prediction. https://aclanthology.org/2021.wnut-1.42.pdf. Proceedings of WNUT 2021.

[PyTorch Contributors2024] PyTorch Contributors. 2024. AdamW — PyTorch 2.12 documentation. https://docs.pytorch.org/docs/2.12/generated/torch.optim.AdamW.html. Accessed: 2024.

[TheDevastator2021] TheDevastator. 2021. Unlock universal language with the LINCE dataset. http

s://www.kaggle.com/datasets/thedevastator/unlock-universal-language-with-the-lince-dataset. Kaggle dataset.

[Winata et al.2021] Genta Indra Winata, Samuel Cahyawijaya, Zhaojiang Liu, Zihan Lin, Andrea Madotto, and Pascale Fung. 2021. Are multilingual models effective in code-switching? https://aclanthology.org/2021.calcs-1.20.pdf. Proceedings of the Fifth Workshop on Computational Approaches to Linguistic Code-Switching (CALCS 2021).

A Group Contributions

Our group met and worked together on all elements of the project. All decisions were made together and we all share equal responsibility.

B Generative AI Usage

To comply with Generative AI policy we list the reasons we used AI in this project:

- help in Latex syntax;
- generating Python code for creating plots.

C Additional Figures

Hyperparameter	Values Explored
Learning Rate	$1 \times 10^{-5},\ 3 \times 10^{-5},\ 5 \times 10^{-5}$
Batch Size	4, 8, 16
Training Epochs	3, 5, 10
Weight Decay	0.0, 0.01, 0.1
Max Sequence Length	64, 128, 256

Table 4: Hyperparameter search space.

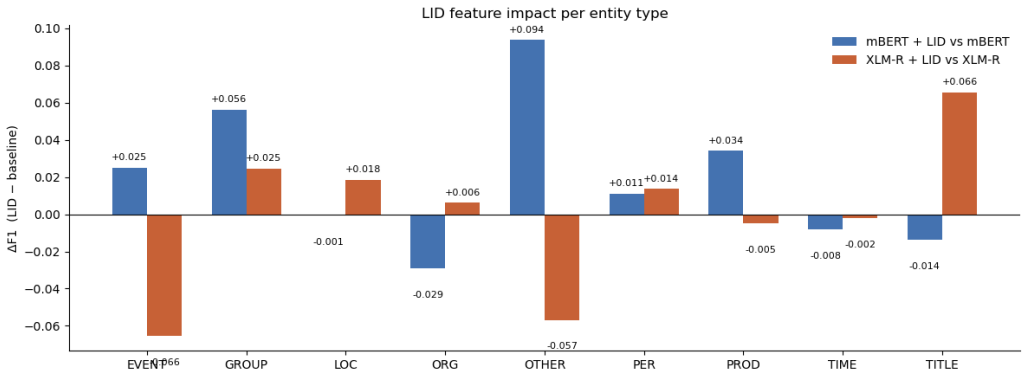

Figure 2: The impact of the LID module (Δ F1).

Entity	Support	mBERT	mB+LID	XLM-R	XR+LID
PER	940	0.744	0.755	0.772	**0.786**
LOC	511	0.719	0.718	0.716	**0.734**
PROD	256	0.482	**0.516**	0.514	0.509
TITLE	157	0.433	0.420	0.360	**0.426**
ORG	154	0.392	0.362	0.413	**0.419**
GROUP	118	0.477	**0.533**	0.493	0.518
TIME	108	0.415	0.407	**0.426**	0.423
OTHER	65	0.296	0.389	**0.419**	0.362
EVENT	46	0.268	0.293	**0.371**	0.306

Table 5: Token-level F1-score comparison per entity type.

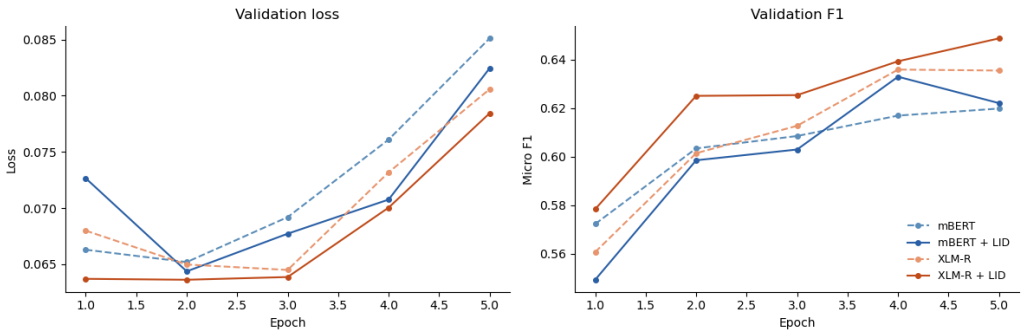

Figure 3: Learning curves for all models.

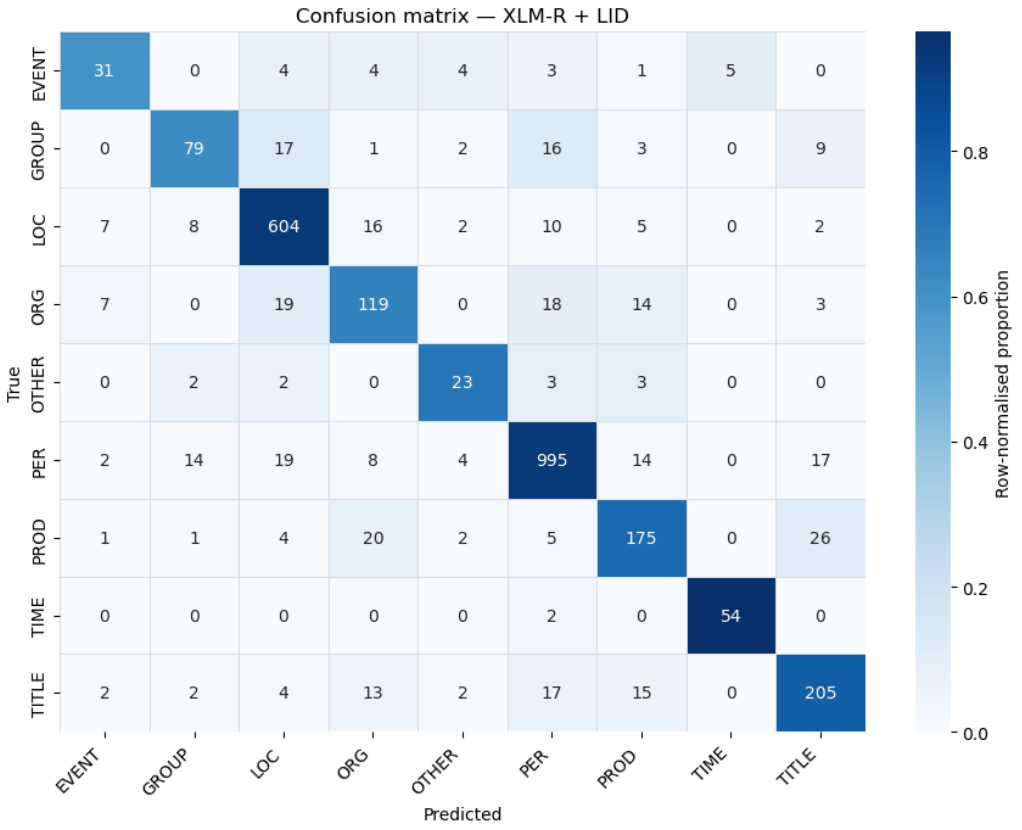

Figure 4: Confusion matrix of the best performing model.

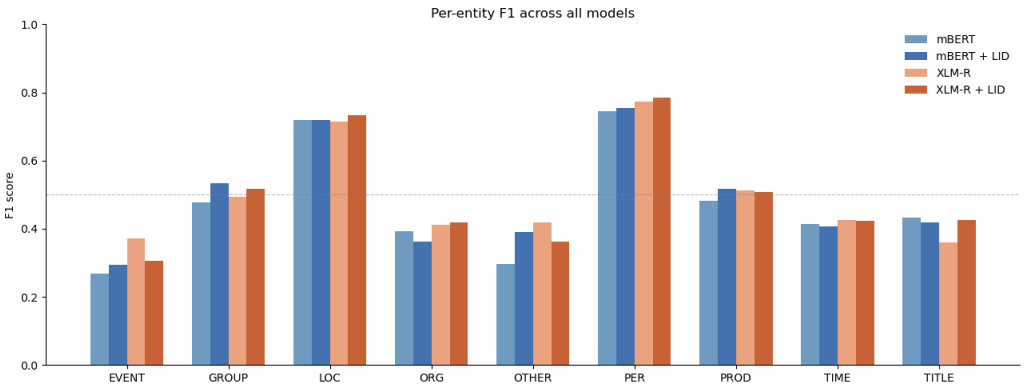

Figure 5: F1-score comparison per entity type across all models.