WEST VISAYAS STATE UNIVERSITY

College of Information and Communications Technology

CCS 249 Final Project Documentation

HiliTag: Fine-Tuning XLM-RoBERTa for Named Entity Recognition in Low-Resource Hiligaynon Language

Prepared By:

Dallas Aquino

Frederick Jibril Buñag

Ethan Jed Carbonkell

Vincent Corpes Jr.

Submitted to:

Mr. John Christopher Mateo
CCS 249 Instructor

HiliTag: Fine-Tuning XLM-RoBERTa for Named Entity Recognition in Low-Resource Hiligaynon Language

Dallas Aquino, Frederick Jibril Buñag, Ethan Jed Carbonell, Vincent Corpes Jr. *(WVSU CICT)* | May 22, 2026

Abstract—The current AI digital divide disproportionately disenfranchises regional languages, leaving the 11 million Hiligaynon speakers underrepresented in modern NLP applications. HiliTag addresses this gap by developing a specialized sequence labeling engine for Named Entity Recognition (NER). By fine-tuning the XLM-RoBERTa (XLM-R) architecture, this project aims to process and preserve cultural history and historical manuscripts from digital obsolescence, providing a robust tool for historians, digital archivists, journalists, and local government officials.

Keywords—*Hiligaynon, Named Entity Recognition, XLM-RoBERTa, Low-Resource NLP, Sequence Labeling*

CONTENTS

1 Overview and Introduction 2
- 1.1 Application Identity . . . 2
- 1.2 Purpose & Problem Statement . . . 2
- 1.3 Technology Stack . . . 2

2 Functional Requirements & System Methodology 2
- 2.1 Data Collection and Preprocessing Pipeline . . . 2
- 2.2 Model Architecture and Training . . . 3
- 2.3 Core Application Features (Deployment) . . . 3

3 Evaluation, Metrics, and Error Analysis 4
- 3.1 Performance Metrics and Baseline Comparison . . . 4
- 3.2 Deep Error Diagnostics . . . 5

4 Future Enhancements and Roadmap 6
- 4.1 Current System Limitations . . . 6
- 4.2 Planned Features and Improvements . . . 6

5 Appendix / Project Submissions 6

1. OVERVIEW AND INTRODUCTION

1.1. Application Identity

The application is officially named as **HiliTag**.

1.2. Purpose & Problem Statement

The Problem: The rapid advancement of artificial intelligence and natural language processing (NLP) has inadvertently widened a global digital divide, disproportionately benefiting high-resource languages while leaving regional and low-resource languages heavily marginalized. While models trained on languages like English, Chinese, and Spanish benefit from massive datasets and significant algorithmic innovation, low-resource languages suffer from chronic data scarcity and architectural biases [2]. In the context of the Philippines, a linguistically diverse archipelago, this disparity is particularly acute. Recent surveys assessing global digital language support reveal that regional Philippine languages possess some of the lowest computational representations worldwide, creating a critical barrier to inclusive technology [1].

Among these critically underrepresented languages is Hiligaynon. As a major Austronesian language spoken by approximately 10 to 11 million people—predominantly across the Western Visayas, Negros Occidental, and Soccsksargen regions—Hiligaynon possesses a vibrant literary and cultural history [**teves2025hiligayner**]. However, it remains severely underdeveloped within the realm of computational linguistics. Until recently, there has been a near-total absence of publicly available, manually annotated corpora or baseline sequence-labeling models tailored for the language.

Consequently, contemporary AI architectures struggle to parse Hiligaynon's unique linguistic structure. Off-the-shelf, high-resource AI models frequently fail to comprehend its complex agglutinative morphology, intricate system of affixes (such as *nag-*, *gin-*, and *pag-*), and nuanced grammatical frameworks. Because these models cannot accurately tokenize or extract semantic meaning from Hiligaynon syntax, a severe threat of digital obsolescence looms over the region's recorded history.

Vital historical manuscripts, regional journalism, localized religious texts, and culturally significant archives are effectively rendered invisible to modern digital indexing and retrieval systems. Without foundational NLP tools—such as Named Entity Recognition (NER) to automate metadata tagging and construct knowledge graphs—these texts cannot be meaningfully processed, preserved, or translated by modern software. This not only risks the permanent loss of digital accessibility for centuries of cultural heritage but also structurally disenfranchises millions of native speakers from actively participating in the modern, AI-driven digital ecosystem.

The NLP Solution: HiliTag delivers a specialized sequence labeling pipeline engineered specifically for Named Entity Recognition (NER) within the Hiligaynon language. By automatically extracting crucial entities—such as People, Organizations, Locations, and Dates—HiliTag enables downstream systems to effectively index and semantically process Hiligaynon literature.

Target Audience: This tool is designed for historians, digital archivists, journalists, and local government units requiring automated metadata generation and knowledge-graph construction from regional texts.

1.3. Technology Stack

To ensure a robust, scalable, and highly responsive application, the project relies on the following decoupled architecture:

Architectural Layer	Technologies & Dependencies
Backend (Inference API)	Python 3.10+, Flask, Flask-CORS, PyTorch, Hugging Face transformers
Frontend (User Interface)	Next.js 14 (App Router), React 18, TypeScript, Tailwind CSS, PostCSS
Data & ML Utilities	BeautifulSoup4, spaCy, Label Studio, seqeval, sklearn-crfsuite, scikit-learn, tqdm, joblib

Table 1. HiliTag System Architecture and Dependencies.

2. FUNCTIONAL REQUIREMENTS & SYSTEM METHODOLOGY

2.1. Data Collection and Preprocessing Pipeline

- **Data Sourcing:** We constructed the corpus by web-scraping approximately 6,500 sentences using BeautifulSoup4. Sources included Panay News, Hiligaynon Wikipedia dumps, and localized community texts (e.g., JW.org), ensuring a comprehensive linguistic mix of formal journalism, encyclopedic data, and contemporary conversational phrasing.

Data Source	Target Size
Bombo Radyo Iloilo	4,000
JW.org (Hiligaynon)	1,500
Hiligaynon Wikipedia (Wikimedia Incubator)	1,000
PAGASA - DOST (Regional Forecast)	500

Table 2. HiliTag data collection sources and target sentence distribution.

- **Sanitization:** The raw text (`raw_sentences.txt`) was aggressively sanitized via our `clean_dataset.py` script. We applied regex and strict encoding normalizations to remove Unicode anomalies, broken HTML entities, whitespace errors, personally identifiable information (PII), and out-of-language noise, followed by strict deduplication.
- **Annotation (Silver to Gold):** Because manual annotation from scratch is resource-intensive, we applied the zero-shot multilingual GLiNER model to generate a baseline "Silver Standard" across 18 OntoNotes 5.0 categories. Following an 80/10/10 split, the Validation and Test sets were imported into Label Studio, where human annotators meticulously corrected AI biases to forge an immutable "Gold Standard" for evaluation.

Category	Description	Example
PERSON	People (incl. fictional).	Rizal, Maria
ORG	Companies, agencies, institutions.	DOH, Ayala
LOCATION	Countries, cities, mountains, waters.	Philippines
DATETIME	Absolute/relative dates and times.	Dec 25, 3 PM
MONEY	Monetary values and units.	500 Pesos
EVENT	Wars, sports, named festivals.	WWII, Sinulog

Table 3. Named Entity Categories derived from selected OntoNotes 5.0 standard.

- **Tokenization Strategy:** Standard subword tokenizers destroy the contextual syntactic structure of Hiligaynon, which relies heavily on affixes (e.g., nag-, gin-, pag-). We bypassed this by invoking `spacy.blank("xx")` to gracefully manage custom boundary splits prior to sub-word tokenization.
- **BIOES Formatting:** A custom conversion script (`json_to_bioes.py`) mathematically translated Label Studio's character-based offset boundaries into the strict BIOES (Begin, Inside, Outside, End, Single) sequence schema, discarding conflicting boundaries to output clean CoNLL-formatted data.

2.2. Model Architecture and Training

- **Primary Architecture:** `xlm-roberta-base` (XLM-R). We selected xlm-roberta-base as our core token classification engine. Unlike mBERT's WordPiece, XLM-R leverages the SentencePiece algorithm, trained on a massive 100-language corpus that includes Philippine languages. This provides a robust vocabulary highly capable of processing the Austronesian roots and Taglish/Spanish loanwords prevalent in Hiligaynon.
- **Hyperparameter Tuning & Training:** Training on Google Colab T4/A100 GPUs using the Hugging Face Trainer API.

Hyperparameter	Configured Value
Batch Size (`per_device_train/eval_batch_size`)	16
Learning Rate (`learning_rate`)	2e-5 (0.00002)
Epochs (`num_train_epochs`)	5
Weight Decay (`weight_decay`)	0.01
Warmup Ratio (`warmup_ratio`)	0.1 (10%)
LR Scheduler (`lr_scheduler_type`)	`'linear'`

Table 4. Core hyperparameters for XLM-RoBERTa fine-tuning.

- **Training Logs:** The model completed its training sequence over 8 epochs and 2,608 global steps, demonstrating rapid early convergence. By the third epoch, it achieved its lowest validation loss of roughly 0.1979 alongside a peak accuracy of nearly 94%. However, subsequent epochs revealed a classic divergence: training loss steadily dropped to a low of 0.0862 by the final epoch, while validation loss slowly crept up to 0.2191, signaling mild overfitting. Despite this increase in validation loss, the model's F1-score ultimately peaked at 0.754 in the eighth epoch, driven by a continuous improvement in recall.

Epoch	Training Loss	Validation Loss	Precision	Recall	F1	Accuracy
1	0.477320	0.341530	0.504651	0.552163	0.527339	0.908380
2	0.244653	0.220799	0.708478	0.694656	0.701499	0.932542
3	0.191055	0.197912	0.746282	0.723494	0.734711	0.939944
4	0.160268	0.200850	0.706923	0.779474	0.741428	0.937640
5	0.148681	0.207344	0.722581	0.759966	0.740802	0.938268
6	0.111225	0.215351	0.704906	0.792197	0.746006	0.935824
7	0.089867	0.215857	0.715283	0.782019	0.747164	0.937011
8	0.086209	0.219158	0.726060	0.784563	0.754179	0.938966

2.3. Core Application Features (Deployment)

- **Backend Server:** A decoupled Flask API exposing a `/api/predict` endpoint to process text strings and return entity JSON payloads.
- **Frontend Interface:** The Next.js client acts as a proxy to the ML backend. It programmatically translates the mathematical JSON bounding offsets into dynamic UI components, overlaying distinct visual highlights on detected entities directly within the user's original text.

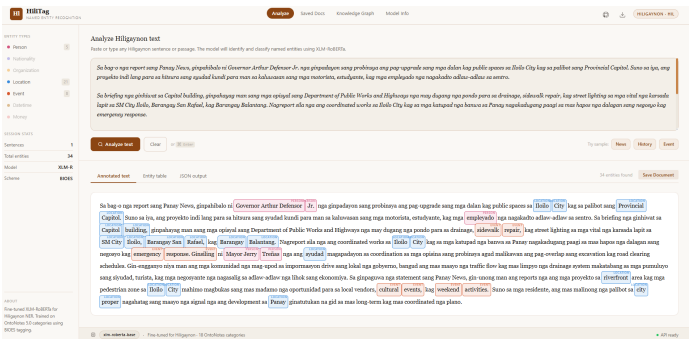


Figure 1. User interface of the HiliTag web application for Named Entity Recognition under the Analyze Tab.

HiliTag: Fine-Tuning XLM-RoBERTa for Named Entity Recognition in Low-Resource Hiligaynon Language

Aquino, Buñag, Carbonell – Corpes Jr. *(WVSU CICT)* | May 22, 2026

The Analyze Tab is the main workspace where you enter Hiligaynon text and run named entity recognition. It sends your input to the backend model, then returns highlighted entities like person, organization, location, datetime, and more. You can review results in multiple formats, including annotated text, entity table, and JSON output. It is best for fast inference, quick iteration, and checking model behavior on new or edited passages.

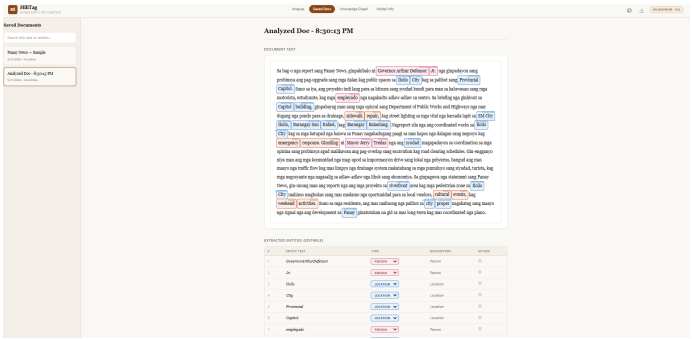


Figure 2. User interface of the HiliTag web application for Named Entity Recognition under the Saved Docs Tab.

Saved Docs is your document library for storing analyzed texts together with their extracted entities. You can search, reopen, review, and edit previously saved entries without re-running inference every time. It also supports refining entity labels directly, so your curated outputs improve over time. This tab is useful for building a reusable dataset of validated examples and tracking annotation progress.

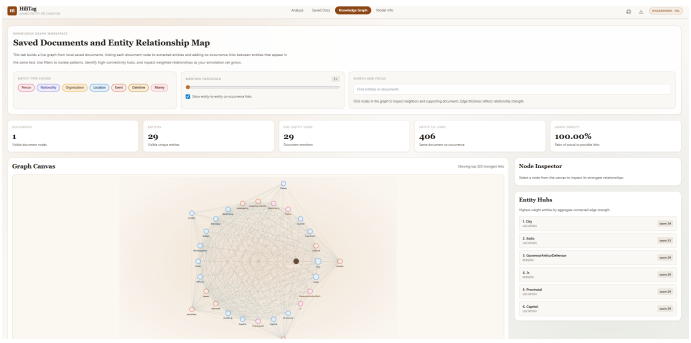


Figure 3. User interface of the HiliTag web application for Named Entity Recognition under the Knowledge Graph.

Knowledge Graph turns your saved documents and entities into a relationship map. It connects documents to entities and also shows co-occurrence links between entities so you can spot patterns, hubs, and frequent associations. You can filter by entity type, adjust mention thresholds, search nodes, and inspect weighted edges to explore how concepts relate across your corpus. This tab is ideal for error analysis, trend discovery, and presenting insights from accumulated NER results.

3. EVALUATION, METRICS, AND ERROR ANALYSIS

3.1. Performance Metrics and Baseline Comparison

We evaluate the model against the manual Gold Standard test set using the `seqeval` library to calculate strict entity-level precision, recall, and F1-scores.

Entity	Precision	Recall	F1-Score	Support
DATETIME	0.76	0.76	0.76	107
EVENT	0.63	0.70	0.66	181
LOCATION	0.78	0.78	0.78	352
MONEY	0.70	0.49	0.58	43
ORG	0.38	0.27	0.32	44
PERSON	0.77	0.76	0.76	467
micro avg	0.74	0.73	0.73	1196
macro avg	0.57	0.54	0.55	1196
weighted avg	0.73	0.73	0.73	1196

Table 5. XLM-RoBERTa Performance classification report.

The transformer model is the stronger overall system, with a weighted F1 of 0.73 and a macro F1 of 0.55, which means it performs well on the common classes but still struggles on rare or ambiguous ones. It is especially solid on PERSON and LOCATION, both at 0.76–0.78 F1, and it also does reasonably well on DATETIME and EVENT. The main weakness is recall on MONEY and ORG. Overall, this suggests XLM-R is better at capturing broader contextual patterns and entity boundaries, but it still needs more balanced training data and better handling of low-frequency classes.

Entity	Precision	Recall	F1-Score	Support
DATETIME	0.63	0.53	0.58	107
EVENT	0.50	0.35	0.41	181
LOCATION	0.57	0.50	0.53	352
MONEY	0.63	0.28	0.39	43
ORG	0.46	0.34	0.39	44
PERSON	0.58	0.46	0.51	467
micro avg	0.57	0.45	0.50	1196
macro avg	0.48	0.35	0.40	1196
weighted avg	0.57	0.45	0.50	1196

Table 6. CRF Baseline Performance classification report.

The CRF baseline is more conservative and ends up with a lower overall weighted F1 of 0.57 and macro F1 of 0.48, so it is less robust than XLM-R across the full label set. It shows relatively strong precision on DATETIME and MONEY, but recall is noticeably weaker on EVENT, LOCATION, MONEY, and ORG, which means it tends to miss entities more often than it falsely predicts them. PERSON is the most stable class for the CRF, but even there it trails the transformer in recall and overall balance. In practical terms, the CRF still captures some clean lexical patterns well, but it lacks the contextual flexibility needed for harder Hiligaynon cases, especially when entities are longer, less canonical, or mixed with Taglish-like surface forms.

The key takeaway is that XLM-RoBERTa is not just slightly better than CRF, it is the model your system needs for reliable Hiligaynon NER in real usage. It delivers higher overall quality (stronger weighted and macro F1), better balance across major entity types, and clearly stronger contextual understanding, which matters for low-resource and code-mixed text where rule-like features break down. While CRF can still be useful as a lightweight baseline and sanity-check model, its weaker recall and poorer generalization on harder classes make it risky as the primary engine. In short, CRF is good for benchmarking, but XLM-RoBERTa is the necessary production backbone if the goal is robust entity extraction, better adaptability, and future scalability as your dataset grows.

HiliTag: Fine-Tuning XLM-RoBERTa for Named Entity Recognition in Low-Resource Hiligaynon Language

Aquino, Buñag, Carbonell – Corpes Jr. *(WVSU CICT)* | May 22, 2026

3.2. Deep Error Diagnostics

We provide an insightful breakdown of the model's limitations through a confusion matrix.

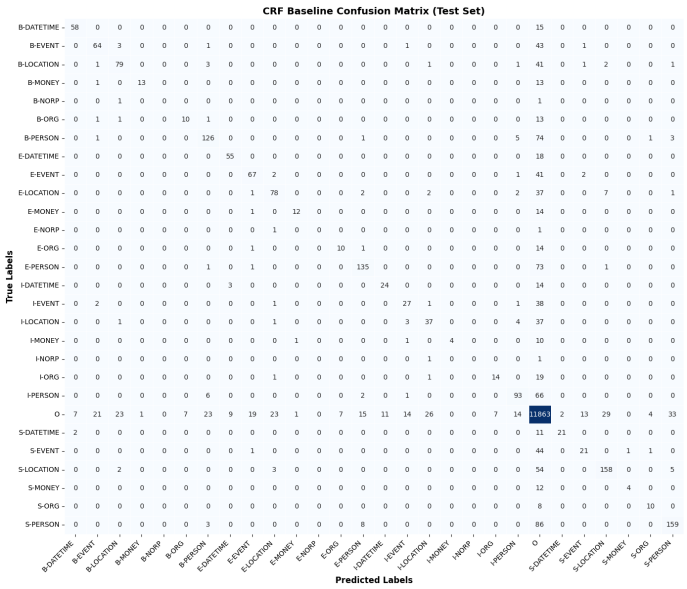


Figure 4. CRF Baseline Confusion Matrix.

The CRF matrix shows a heavy concentration on the O class, with many true entity tags collapsing into O, which indicates recall weakness more than label confusion. You still get a visible diagonal for major classes like PERSON, LOCATION, and DATETIME, but ORG and MONEY boundary tags are comparatively weak on the diagonal and frequently missed. In practical terms, CRF is catching obvious lexical patterns but dropping many valid entities when context is subtle, long-span, or code-mixed. This is consistent with feature-based models that are precise on simple cues but brittle on harder contextual cases.

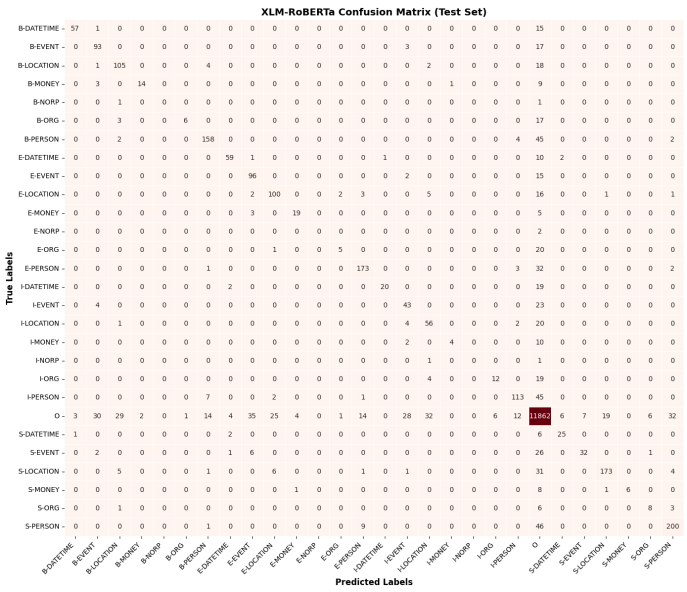


Figure 5. XML RoBERTa Confusion Matrix.

The XLM-R matrix has a stronger and cleaner diagonal than CRF for most high-support tags, especially across PERSON, LOCATION, EVENT, and DATETIME BIOES variants. It still has substantial leakage into O, but less severe than CRF for many classes, which means better recall and more stable span recognition overall. The main weak zone is still ORG (especially B-ORG/I-ORG) and rare labels, where misses remain common. Overall, this plot supports that XLM-R is meaningfully better at contextual token classification, but not yet fully robust on low-frequency and boundary-sensitive entities.

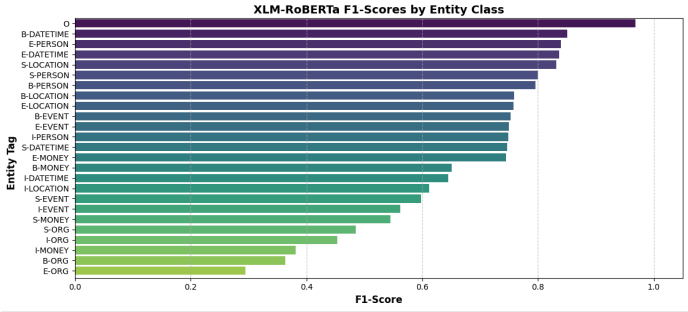


Figure 6. XML RoBERTa F1 Scores by Entity Class.

The F1 chart confirms strong performance on common classes and weaker performance on sparse/ambiguous ones: upper-tier scores are dominated by PERSON, DATETIME, and parts of LOCATION. A key caution is that O appears near the top and inflates perceived overall stability, so model quality should be judged mainly from non-O entity tags. The spread between strong boundary tags and weak ones also suggests boundary consistency is still uneven (some B/E/S tags perform better than their I counterparts). In short, the model is good where data is abundant and context is clearer, but still data-limited on rare classes.

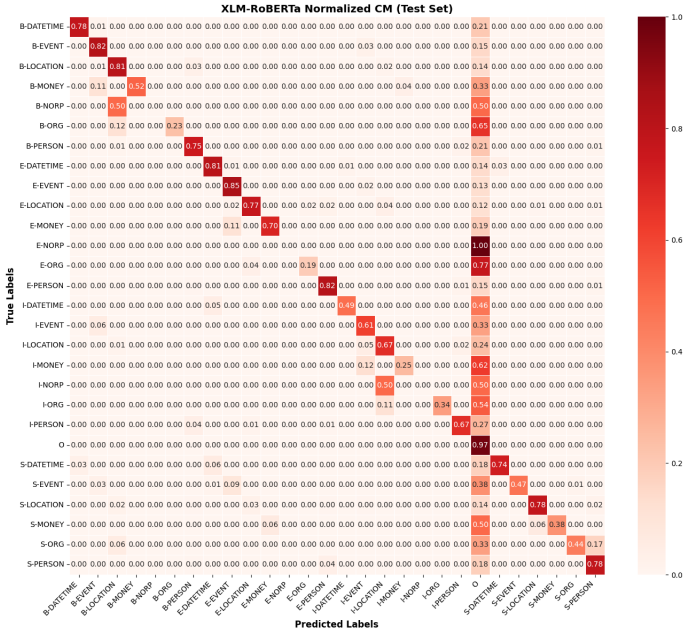


Figure 7. XML RoBERTa Normalized CM.

This is the most diagnostic plot because each row is a recall distribution: it clearly shows where each true label is going when missed. The biggest pattern is systematic drift from true entity tags into O (not just into the wrong entity type), especially for weaker classes like ORG, MONEY, and very rare tags. Strong rows (higher diagonal fractions) appear for many PERSON/DATETIME/LOCATION tags, while weak

rows expose exactly which labels need targeted augmentation. This tells you the bottleneck is not only semantic confusion; it is also entity retention and boundary robustness under sparse evidence.

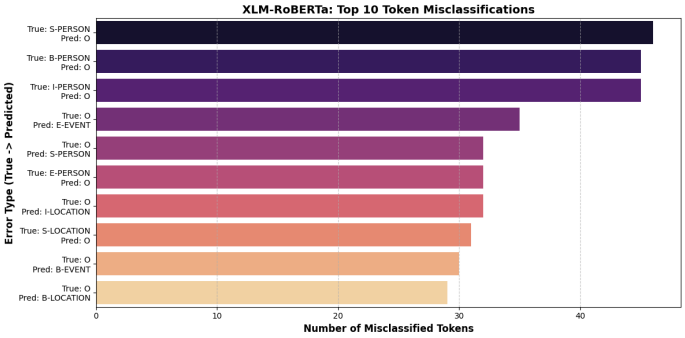


Figure 8. XML RoBERTa: Top 10 Token Misclassifications

The top errors are dominated by true entity tags predicted as O (notably S-PERSON, B-PERSON, I-PERSON), which confirms false negatives are a primary failure mode. There has also notable O -> entity errors (for example toward event/person/location tags), meaning the model sometimes overfires entity labels in non-entity context. Together, this shows a precision-recall tension: the model both misses real entities and occasionally hallucinates spans, especially around boundary tags. The strongest next improvement lever is class/boundary-focused data curation and calibration aimed specifically at reducing entity -> O collapse while controlling O -> entity overprediction.

The overall takeaway from the confusion-matrix analysis is that your XLM-RoBERTa pipeline is clearly the better primary model, with stronger diagonal concentration and better recovery of major entity classes than the CRF baseline, but both models still struggle with minority and boundary-sensitive tags where predictions collapse to O or drift across similar labels. The normalized and top-misclassification views confirm that the central failure mode is false negatives on less frequent entities, while some false positives still appear in ambiguous contexts; this aligns with your low-resource setting and class imbalance. In conclusion, your current system is already strong enough to justify XLM-RoBERTa as the production backbone, while CRF remains useful as a baseline diagnostic reference, and the highest-impact next improvements are targeted data balancing, boundary-focused annotation cleanup, and continued error-driven refinement using the exact confusion/misclassification loops you built in this notebook.

4. FUTURE ENHANCEMENTS AND ROADMAP

4.1. Current System Limitations

The system is already functional end-to-end, but it currently faces several structural limits:

1. **Tokenizer & Backend Compatibility:** The backend and evaluation paths depend on a checkpoint tokenizer that is not fully compatible with the installed tokenizers runtime. Both `app/backend/app.py` and `evaluation/eval_script.py` must fall back to `xlm-roberta-base` instead of cleanly loading the saved tokenizer, meaning inference quality is not guaranteed to match the trained export.
2. **Local-Only Document Storage:** The saved-documents flow is local-only through browser storage in `app/frontend/src/lib/storage.ts`. There is no cross-device sync, no multi-user collaboration, and no server-side persistence or audit trail beyond the backend JSONL log.
3. **Manual UI Interaction Model:** While saved entities can be edited, there is no true document versioning, no undo history, and no robust conflict resolution if the same document is changed in multiple tabs.
4. **Diagnostic Constraints:** On the diagnostics side, `evaluation/confusion_matrix.ipynb` gives useful error analysis, but its word-to-subword alignment is still approximate. The confusion matrix can be dominated by the 'O' class, so minority-entity errors remain visually under-emphasized even when the model is wrong.
5. **Heuristic Knowledge Graph:** The knowledge graph in `app/frontend/src/components/KnowledgeGraphView.tsx` is informative but relies on a heuristic visualization built from co-occurrence and document links. It shows association patterns rather than a true semantic graph with typed, factual relation extraction.

4.2. Planned Features and Improvements

To address these limitations, the following steps and long-term goals are planned:

1. **Tokenizer Standardization:** The clearest next step is to remove tokenizer fragility at the source by regenerating and re-saving the model artifacts with a runtime-matched `transformers/tokenizers` stack. Standardizing this export pipeline will ensure the backend, evaluator, and notebook all load the same checkpoint without fallback logic.
2. **Robust Backend Persistence:** The frontend will evolve from browser-only document storage toward a real persistence layer—ideally a backend API for saving documents, editing entities, and keeping revision history. This makes the *Saved Docs* and *Knowledge Graph* tabs durable across sessions and devices.
3. **Enhanced Knowledge Graph:** The graph view will be upgraded from a visualizer into a real analysis workspace by adding:
   - Better layout control and clustering by entity type.
   - Edge filtering by relation strength and search-with-focus capabilities.
   - Export functions (to image or JSON).
   - A side panel explaining why a specific node or edge appears.
4. **Automated Diagnostics Reporting:** `evaluation/confusion_matrix.ipynb` will be improved by centralizing tokenizer loading, adding normalized matrices, and generating automated ORG-vs-LOCATION and Taglish error summaries. This will turn the notebook into a repeatable diagnostics report instead of a one-off exploratory tool.
5. **Tighter Editor-State Flow:** The analyzer (`app/frontend/src/components/NERAnalyzer.tsx`) will benefit from passing analyzed text and edited entities through a single source of truth, preserving save states after entity edits, and exposing validation when entity labels are changed.
6. **Long-Term Architectural Maturity:** The system will incorporate model/version metadata, richer backend health reporting, better logging for failed inferences, and a clearer release workflow so all components stay aligned as the project evolves.

5. APPENDIX / PROJECT SUBMISSIONS

GitHub Repository Details

The source code, including frontend, backend, and Jupyter Notebooks for training, can be found in our organization repository.

https://github.com/dallqq/HiliTag/

REFERENCES

[1] G. F. Simons, C. D. Fennig, et al., *A survey of digital language support*, 2022.

[2] S. Ranathunga, E. A. Lee, I. Skadiņa, et al., "Neural machine translation for low-resource languages: A survey", 2023.