COMPUTATIONAL LINGUISTICS

NAMED ENTITY RECOGNITION

FOR CODE-MIXED TEXT

Naman Singhal
Shrish Kadam
Yash More

Contents

1 Introduction

1.1 Context

In multilingual communities, *code-mixing*—the act of switching between two or more languages during a single sentence, phrase, or conversation—is a common and normal linguistic practice. Code-mixing is especially common on social media sites like *Twitter*, *Facebook*, *Instagram*, and *WhatsApp* in India, where speakers frequently switch between Hindi, English, Tamil, Telugu, Bengali, and other languages. Because of this, code-mixed language data can be studied in great detail on Indian social media.

1.2 Prospects for Research on Code-Mixing

- **Sociolinguistic insights:** Social dynamics, language preference, and cultural identity are all reflected in code-mixing. By examining it, researchers can better comprehend how bilingual or multilingual speakers use language in everyday interactions.
- **Language Innovation and Evolution:** Slang, neologisms, transliterations, and creative language use are frequently found in code-mixed texts. This offers a dynamic dataset for researching the interactions and evolution of languages.
- **Engagement and Localization:** By examining code-mixed data, media outlets and businesses can produce more relatable and interesting content for multilingual users. Improved user experiences can result from NLP tools that comprehend code-mixing.
- **Multilingual NLP with Rich Data:** Naturally, Indian code-mixed content provides a training ground for multilingual models, transfer learning techniques, and cross-lingual embeddings.

Named Entity Recognition (NER) is a key information retrieval task that forms the basis of many subsequent NLP applications. Conventional NER systems have trouble handling the complexity brought about by code-mixed content because they are usually made for monolingual text.

1.3 Problem Statement

In Natural Language Processing (NLP), Named Entity Recognition (NER) is a crucial task that entails recognizing and categorizing named entities in text, including people, organizations, and places. Even though NER for monolingual and grammatically structured texts has advanced significantly, standard NER systems still have problems when dealing with code-mixed data, especially in social media settings. This difficulty is particularly noticeable when dealing with code-mixed text in Hindi and English, which is common on Indian internet platforms.
The particular issues faced are as follows:

- **Language Identification Difficulties:** It can be challenging to identify the language of individual tokens in code-mixed text because of the frequent and unpredictable language switching. Although it is not simple in this context, accurate language identification is essential for downstream NER.
- **Non-standard Spellings and Transliterations:** People on social media frequently use informal spelling and write Hindi words in Roman script. As a result, the same word can have several variations (such as *dost*, *dosth*, and *dhost*), which makes it challenging for conventional token-based models to generalize.
- **Script Variations:** Because of keyboard or platform limitations, the same language may appear in different scripts, such as Devanagari for Hindi or Roman script. These modifications add more complexity to the preprocessing and normalization steps.
- **Culturally-Specific Entities:** Standard NER training corpora, which are usually Western-centric, either lack or have insufficient representation of many named entities in Indian contexts (such as local celebrities, locations, and festivals).
- **Absence of Annotated Corpora:** For supervised NER training, there aren't many sizable, openly accessible, annotated code-mixed corpora. Efforts to develop models and benchmark them are hampered by this lack of resources.

These difficulties necessitate the development of specialized NER systems that understand the social dynamics and linguistic quirks present in code-mixed communication.

1.4 Project Objectives

This project's main goal is to create a Named Entity Recognition (NER) system that is specifically suited for code-mixed Hindi-English text, like that which appears in Indian social media posts. The objective is to achieve strong entity recognition performance in a variety of linguistic contexts while managing the complexities of mixed-language usage.

The following are the project's main goals:

- **Establishing Guidelines for Annotation:** Create a thorough set of annotation guidelines tailored to code-mixed NER that address issues like entity boundary detection, language tagging, and managing terms that are unclear or culturally specific.
- **Corpus Development:** Create a top-notch dataset of social media sentences that are code-mixed in Hindi and English and manually annotated with named entities. This entails finding names of individuals, groups, and places in extremely informal and multilingual data.
- **Model Implementation with CRF:** Use Conditional Random Fields (CRF), a sequence labeling model that works well for NER and other structured prediction tasks. Character-level n-grams, contextual word windows, language tags, and POS tags are among the features.
- **Performance Evaluation:** Use common metrics (precision, recall, and F1-score) to compare the model's accuracy and robustness across various named entity types and code-mixing levels.
- **Linguistic Analysis:** Examine the effects of code-mixing frequency and pattern (intra-sentential, intra-word, etc.) on the model's accuracy in identifying and categorizing entities, offering suggestions for future improvements.

2 Project Scope

The project will encompass:

1. **Dataset Creation and Annotation**:
    - Collection of Hindi-English code-mixed social media text from platforms like Twitter, Facebook, and WhatsApp
    - Development of comprehensive annotation guidelines for named entities in code-mixed text
    - Creation of a gold-standard annotated corpus with at least 1000 sentences
2. **Entity Types**:
    - Person (PER)
    - Location (LOC)
    - Organization (ORG)
    - Cultural (CUL)
    - Relation (REL)
    - Kinship (KIN)
    - Particle (PAR)
3. **Model Development**:
    - Feature-based Conditional Random Field (CRF) model.
4. **Evaluation**:
    - Standard metrics: Precision, Recall, F1-score
    - Analysis across different entity types
    - Analysis based on code-mixing density and patterns

3 Project Overview

This project focuses on developing a Named Entity Recognition (NER) system for Hindi-English code-mixed text commonly found on social media platforms such as Twitter and Facebook. The code-mixed nature of the data presents unique linguistic challenges, making traditional monolingual NER models less effective.

Repository Structure

- `training_data/` — Annotated training dataset
- `testing_annotated_data/` — Gold-standard test data
- `testing_data.csv` — Raw testing data
- `model.py` — Contains the CRF-based NER model
- `run_model.py` — Script to train and test the model
- `requirements.txt` — Python dependencies
- `RESULTS.md` — Contains evaluation results and observations
- `cl_project_presentation.pdf` — Project presentation slides
- `guidelines.md` — Annotation guidelines
- `LexiCoders-Outline.pdf` — Team/project outline

Methodology

- **Data Collection**: Sourced Hindi-English tweets and Facebook posts with a variety of named entities (Person, Location, Organization, etc.).
- **Annotation**: Manually annotated using BIO tagging scheme.
- **Model**: Implemented a Conditional Random Field (CRF) model using features like:
  - Word shape and POS
  - Bilingual word embeddings
  - Language identification tags
- **Evaluation Metrics**: Precision, Recall, F1-score

How to Run

1. Clone the repository:

   ```
   git clone https://github.com/the-neemon/CL-1-Project-LexiCoders-
   cd CL-1-Project-LexiCoders
   ```

2. Install dependencies:

   ```
   pip install -r requirements.txt
   ```

3. Train and evaluate the model:

   ```
   python run_model.py
   ```

4 Web Interface for Code-Mixed Named Entity Recognition

To improve the accessibility and present the results of our Named Entity Recognition (NER) system on Hindi-English code-mixed social media data, a **web interface** has been developed and deployed at:

`https://the-neemon.github.io/CL-1-Project-LexiCoders-`

Purpose of the Web Interface

The purpose of this interface is to showcase the NER model's performance on real-world, code-mixed text data. Unlike a live demo system, this interface does not accept user input. Instead, it provides **visualizations of the model's predictions** on selected examples from our annotated training and testing datasets. This allows users to:

- Explore the model's handling of noisy, code-mixed input
- Understand the entity categories identified by the system
- View token-wise entity tagging in a clean, structured format

Features

- **Example-Based Output:** Displays pre-selected sentences from the dataset with corresponding model predictions.
- **Token-Level Annotation:** Each word is shown alongside its predicted NER tag (e.g., `PER`, `LOC`, `ORG`, `O`).
- **User-Friendly Design:** The interface is built to clearly present NER results to both technical and non-technical audiences.
- **Static Hosting:** The site is hosted on GitHub Pages using static HTML/CSS/-JavaScript, requiring no backend.

Technical Overview

- **Frontend:** HTML, CSS, JavaScript
- **Data Source:** NER model predictions on annotated Hindi-English code-mixed social media data
- **Model Backend:** Conditional Random Fields (CRF)-based model trained on custom-annotated code-mixed datasets

This interface serves as a bridge between model development and practical demonstration, helping stakeholders understand the capabilities and limitations of the system.

5 Literature Review

5.1 Code-Mixing in Indian Languages

Code-mixing refers to the blending of two or more languages within a single discourse, sentence, or even word. In multilingual societies such as India, this phenomenon is extremely common, especially in informal communication. Users often mix English with regional Indian languages such as Hindi, Tamil, Bengali, or Telugu in both spoken and written forms. This behavior is particularly prominent on social media platforms like Twitter, Facebook, and WhatsApp, where informal, user-generated content dominates.

Numerous studies have examined code-mixed text, pointing out its intricate linguistic structure that defies the presumptions of conventional NLP systems (e.g., Barman et al., 2014; Solorio et al., 2014). Tokenization, part-of-speech tagging, parsing, and entity recognition are all severely hampered by the inconsistent grammar, transliteration, and spelling. The necessity for context-aware models that can dynamically adjust to language switching at the word or subword level has also been highlighted by research.

5.2 Named Entity Recognition Approaches

Named Entity Recognition (NER) has traditionally been approached using three major methodologies:

Approaches to Named Entity Recognition

- **Rule-based Systems:** To identify entities, these systems use manually created rules, dictionaries, and regular expressions. Although they are quick and easy to understand, they frequently lack generalizability and perform poorly on noisy or informal texts.
- **Statistical Models:** Methods like Conditional Random Fields (CRFs) and Hidden Markov Models (HMMs) use probabilistic modeling to represent the word sequence and the associated entity tags. Because of their capacity to enforce sequential consistency and integrate rich linguistic features, CRFs in particular have seen extensive use.
- **Deep Learning Techniques:** Neural network-based techniques, like Bi-directional LSTM with CRF (BiLSTM-CRF), have demonstrated better results in sequence labeling tasks in recent years. By acquiring deep contextual embeddings from extensive corpora, transformer-based models such as BERT and its multilingual variants (mBERT, XLM-R) have further improved the state-of-the-art.

These models work well on formal, monolingual data, but because of noise, a lack of training data, and unpredictable linguistic variation, they become less effective when used on code-mixed social media text.

6 Limitations in Current Research

There are still a lot of unanswered questions in code-mixed natural language processing, especially when it comes to Named Entity Recognition for Hindi-English code-mixed content. The main restrictions on the current corpus of research are as follows:

- **Limited Resources:** Annotated corpora for Hindi-English code-mixed NER are scarce. The current datasets are not publicly accessible for research purposes, are too small, or are not diverse enough.
- **Insufficient Preprocessing Instruments:** Particularly for informal text written in Roman script, language identification, transliteration normalization, and part-of-speech tagging are still lacking in code-mixed contexts.
- **Model Generalization:** Formal, monolingual datasets are used to train a large number of cutting-edge NER models. Without domain adaptation or fine-tuning, these models have trouble generalizing to noisy, low-resource code-mixed data.
- **Contextual and Cultural Nuance:** It can be challenging to recognize named entities on Indian social media such as local celebrities, regional locations, or cultural allusions that are not mentioned in standard NER tagsets or gazetteers.
- **Absence of Benchmarking Efforts:** Since code-mixed NER lacks standardized benchmarks and evaluation protocols, it is challenging to compare and replicate findings across studies.

These gaps highlight the need for focused research on developing reliable, flexible NER systems for Indian social media data that is code-mixed. By creating an annotated corpus, establishing annotation standards, and testing both conventional and machine learning-based NER models in code-mixed scenarios, this project advances this goal.

7 Methodology

7.1 Data Collection and Preparation

7.1.1 Data Sources

We utilized the following datasets for our project:

- We have taken dataset for training from the following source:
  Click here to visit the site.
- Replication Data for Automatic Language Identification in Code-Switched Hindi-English Social Media Text
- Code-Mixed Dataset for Hindi-English containing some manually scraped social media posts
- Additional sentences manually collected and annotated from Twitter, Facebook and WhatsApp

7.1.2 Preprocessing Steps

- Text normalization to handle non-standard spellings
- Filtering for sentences with substantial mixing (at least 20% of both languages)

7.2 Annotation Guidelines

We developed comprehensive annotation guidelines to ensure consistency in the labeling of named entities in code-mixed text. Our tagset included:

- **B-Per/I-Per**: Beginning and intermediate tokens of Person names
- **B-Org/I-Org**: Beginning and intermediate tokens of Organization names
- **B-Loc/I-Loc**: Beginning and intermediate tokens of Location names
- **B-Cul/I-Cul**: Beginning and intermediate tokens of Cultural terms
- **B-Kin/I-Kin**: Beginning and intermediate tokens of Kinship terms
- **B-Par/I-Par**: Beginning and intermediate tokens of Particles (e.g. “yaar”, “ji” etc.)
- **B-Rel/I-Rel**: Beginning and intermediate tokens of Religious terms
- **Other**: Non-entity tokens

Special attention was given to:

- Entity boundaries in mixed-language contexts
- Handling transliteration variations
- Culturally-specific entity classification

7.3 Feature Engineering for CRF

We implemented a Conditional Random Field (CRF) model with the following features:

7.3.1 Lexical Features

- Word identity, prefix/suffix n-grams
- Word shape (capitalization, digits, etc.)

7.3.2 Contextual Features

- Previous/next words and their language tags
- Word n-grams

7.4 Model Architecture

We used the `sklearn-crfsuite` implementation of CRF for our NER system. The model was trained on our annotated corpus and evaluated using standard NER metrics.

8 Corpus Creation

8.1 Introduction

The process of creating a corpus for Named Entity Recognition (NER) in code-mixed languages, specifically from platforms like Twitter, Facebook and WhatsApp involves several key steps: data collection, cleaning, preprocessing, annotation and ensuring diversity in language use.
Social media platforms are especially rich in informal, conversational and code-mixed text, making them ideal for training NER models aimed at real-world scenarios where language mixing (e.g. Hindi-English, Spanglish) is common.

8.2 Dataset Statistics

We created a substantial corpus of Hindi-English code-mixed text with named entity annotations: The model was trained on 8 different datasets:

1. `train_datav1.csv` (100 sentences, 1,336 tokens)
2. `shrish_data.csv` (179 sentences, 1,624 tokens)
3. `naman_data_2.csv` (525 sentences, 7,144 tokens)
4. `naman_data_3.csv` (338 sentences, 3,209 tokens)
5. `train_datav2.csv` (100 sentences, 1,346 tokens)
6. `annotatedData.csv` (3,085 sentences, 68,506 tokens)
7. `naman_data_1.csv` (281 sentences, 4,976 tokens)
8. `yash_data.csv` (241 sentences, 1,187 tokens)

Total Training Data: 4,849 sentences with 89,328 tokens.
Testing Data: 565 sentences with 11,662 tokens

8.3 Annotation Process

1. Development of detailed annotation guidelines
2. Training annotators on the guidelines
3. Manual annotation of the corpus
4. Quality control
5. Adjudication of disagreements

8.4 Data Format

- Training: CSV with columns `Sent`, `Word`, `Tag`
- Testing: Space-separated format with word and tag per line
- Sentences separated by blank lines

8.5 Dataset Sample

Sample entries from the annotated corpus:

- **Sentence:** *main kal mumbai ja raha hoon*

main	Other
kal	Other
mumbai	B-Loc
ja	Other
raha	Other
hoon	Other

- **Sentence:** *meri favorite actress deepika padukone hai*

meri	Other
favorite	Other
actress	Other
deepika	B-Per
padukone	I-Per
hai	Other

- **Sentence:** *diwali ke liye new clothes khareedne hain*

diwali	B-Cul
ke	Other
liye	Other
new	Other
clothes	Other
khareedne	Other
hain	Other

9 Results and Analysis

Overall Performance

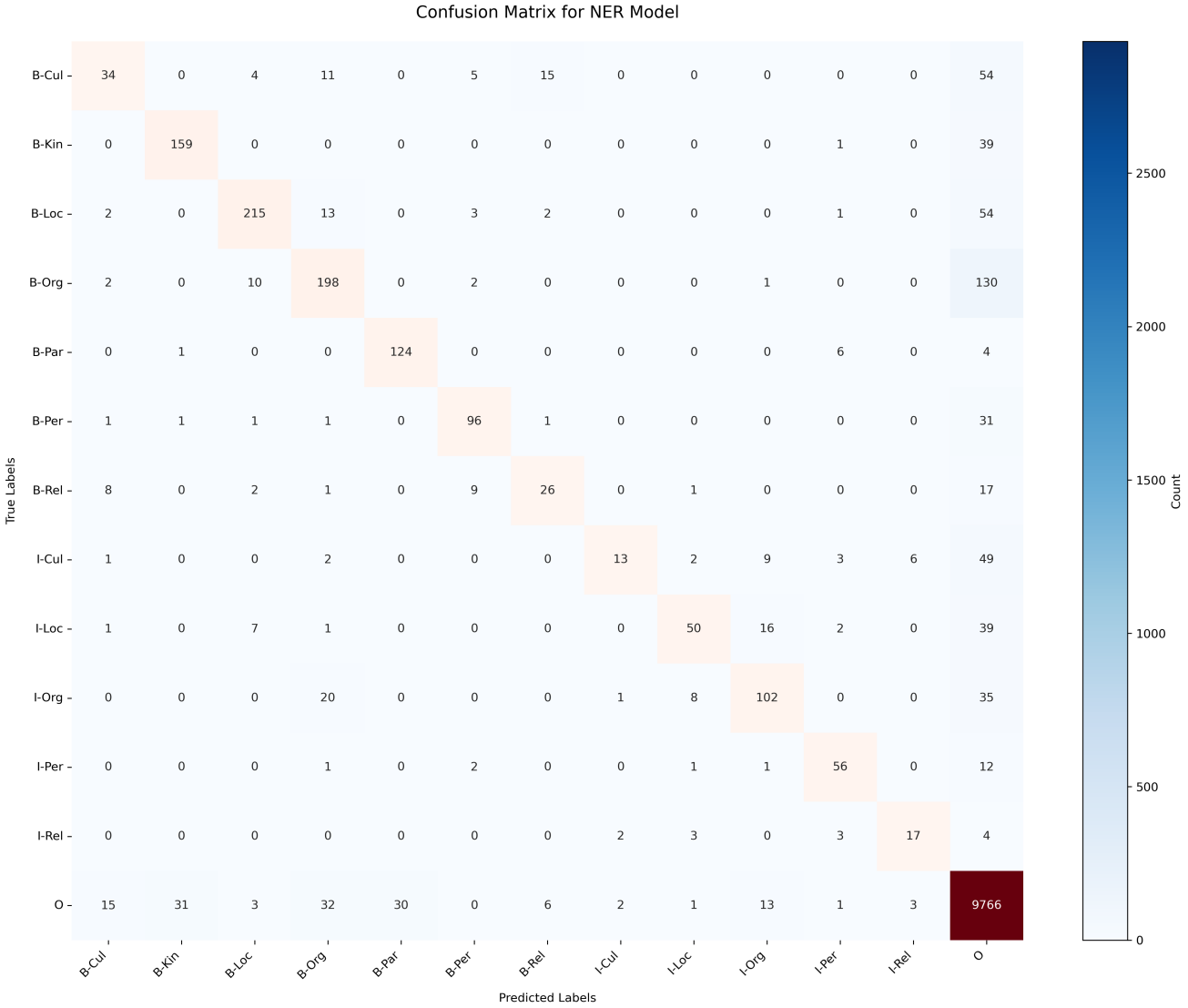
- **Token Accuracy:** 93.1% (10,859 correct out of 11,662 tokens)

Entity-wise Performance

Entity Type	Precision	Recall	F1-Score	Support
B-Cul	0.531	0.276	0.364	123
B-Kin	0.828	0.799	0.813	199
B-Loc	0.888	0.741	0.808	290
B-Org	0.707	0.577	0.636	343
B-Par	0.805	0.892	0.846	139
B-Per	0.821	0.727	0.771	132
B-Rel	0.520	0.406	0.456	64
I-Cul	0.722	0.153	0.252	85
I-Kin	0.000	0.000	0.000	0
I-Loc	0.758	0.431	0.549	116
I-Org	0.718	0.614	0.662	166
I-Par	0.000	0.000	0.000	0
I-Per	0.767	0.767	0.767	73
I-Rel	0.654	0.586	0.618	29
Other	0.954	0.986	0.970	9903

Sample Predictions: Sentence 1

```
aaj:        True=Other        Pred=Other
main:       True=Other        Pred=Other
andheri:    True=B-Loc        Pred=Other
station:    True=I-Loc        Pred=Other
pe:         True=Other        Pred=Other
late:       True=Other        Pred=Other
ho:         True=Other        Pred=Other
gaya:       True=Other        Pred=Other
yaar:       True=B-Par        Pred=B-Par
nan:        True=Other        Pred=Other
mumbai:     True=B-Loc        Pred=B-Loc
```


Confusion Matrix for NER Model
True Labels
Predicted Labels
Count
B-Cul
B-Kin
B-Loc
B-Org
B-Par
B-Per
B-Rel
I-Cul
I-Loc
I-Org
I-Per
I-Rel
O
0
500
1000
1500
2000
2500

9.1 Code-Mixing Pattern Analysis

- **Pure Hindi/English Entities Are More Accurately Recognized**
  The model shows higher accuracy for tokens that are unambiguously in either standard Hindi or English. Entities like `mumbai` (B-Loc) and `yaar` (B-Par) are recognized correctly, indicating strong performance for commonly occurring monolingual tokens.

- **English Spellings in Roman Script Enhance Recognition**
  Tokens written in Romanized English (e.g., `station`, `late`, `main`) are generally well recognized, particularly for non-entity (O) tokens. This suggests that the model benefits from familiar English lexical forms even when embedded in Hindi sentences.

- **Common Entities Recognized Better Than Domain-Specific Ones**
  Entities such as locations (B-Loc, e.g., `mumbai`) and persons (B-Per) have relatively higher F1-scores (0.808 and 0.0.771 respectively). In contrast, culturally or domain-specific entities like B-Cul (F1-score: 0.364) and I-Cul (F1-score: 0.) are poorly predicted, possibly due to their lower frequency and ambiguity in code-mixed contexts.

- **Poor Recall on Long and Nested Entities**
  The transition from B- to I- tags (e.g., B-Loc to I-Loc) often fails, as seen with `station` being mislabeled. This pattern reflects the model's difficulty in maintaining entity continuity across multi-token Roman-script entities.

- **Informal/Slang Phrasing Challenges the Model**
  Informal Hindi expressions like `yaar`, though correctly tagged here, can contribute to noise due to high lexical variance and contextual ambiguity in code-mixed dialogue.

9.2 Impact On NER

The analysis shows that the complexity of code-mixed data has a major effect on NER performance. The model does well with Romanised English and clear, monolingual entities but it has trouble with nested entities, informal slang and terms that are specific to a particular domain or culture. This implies that future research in code-mixed NER ought to concentrate on:

- Improving recognition of multi-token entities, especially those that switch between languages (e.g. B-Loc to I-Loc).
- Expanding the model's understanding of informal and slang terms.
- Enhancing the model's ability to handle domain-specific and culturally-specific entities, potentially by increasing training data diversity.
- Further optimizing the model for handling romanized text and code-switching, particularly in informal contexts.

These adjustments will help increase NER accuracy in real-world, code-mixed scenarios.

10 System Implementation

10.1 Running the Model

Prerequisites

- Python 3.x
- Required Python packages:
  - `sklearn_crfsuite`
  - `pandas`
  - `numpy`

Steps to Run

1. **Data Preparation**
   - Ensure training data is placed in the `training_data` directory.
   - Place testing data as `testing_annotated_data` in the root directory.
2. **Train and Test the Model**

   ```
   python run_model.py
   ```

   This script will:
   - Load all training data
   - Convert and load testing data
   - Train the model
   - Evaluate performance
   - Display results

11 Annotation Guidelines

These guidelines ensure consistency in annotating named entities in Hindi-English code-mixed text, addressing the specific challenges of mixed language text. The annotation scheme uses a **BIO (Beginning-Inside-Outside)** tagging format to mark entity boundaries.

11.1 Entity Types and Definitions

Person (B-Per/I-Per)

Definition: Names of individuals
Examples:

- `Narendra Modi` → [B-Per, I-Per]
- `Amitabh Bachchan` → [B-Per, I-Per]
- `@narendramodi` → [B-Per]

Organization (B-Org/I-Org)

Definition: Companies, institutions, agencies, government bodies
Examples:

- `Indian Railways` → [B-Org, I-Org]
- `State Bank of India` → [B-Org, I-Org, I-Org, I-Org]
- `Delhi University` → [B-Org, I-Org]

Location (B-Loc/I-Loc)

Definition: Geographical locations, buildings, landmarks
Examples:

- `Mumbai` → [B-Loc]
- `Taj Mahal` → [B-Loc, I-Loc]
- `Dilli` / `Delhi` → [B-Loc]

Date/Time (B-Date/I-Date)

Definition: Temporal expressions
Examples:

- `5 January 2023` → [B-Date, I-Date, I-Date]
- `agle mahine` → [B-Date, I-Date]

Culturally-Specific Entities (B-Cul/I-Cul)

Definition: Cultural events, festivals, rituals, traditional practices
Examples:

- `Diwali` → [B-Cul]
- `Durga Puja` → [B-Cul, I-Cul]

Kinship Terms (B-Kin/I-Kin)

Definition: Family relationship terms
Examples:

- `Chacha` → [B-Kin]
- `Dadi` → [B-Kin]

Products/Brands (B-Prod/I-Prod)

Definition: Commercial products, brands, services
Examples:

- `iPhone` → [B-Prod]
- `Maruti Swift` → [B-Prod, I-Prod]

Particles (B-Par/I-Par)

Definition: Discourse particles specific to Indian languages
Examples:

- `yaar` → [B-Par]
- `ji` → [B-Par]

Religious Terms (B-Rel/I-Rel)

Definition: Religion-related terms, places, deities
Examples:

- `Mandir`, `Church` → [B-Rel]
- `Shiva` → [B-Rel]

Other

All tokens not falling into any of the above categories.

12 Entity Boundaries in Mixed-Language Contexts

Basic Principles

- Annotate the entire span of a named entity as a single entity regardless of language shifts.
- Include articles and function words only if they are part of the official name.

Mixed-Language Named Entities

- **Example:** *"State Bank of India ke branch"* → Only `State Bank of India` is **ORG**
- **Example:** *"delhi university mai"* → Only `delhi university` is **ORG**

*Nested Entities

- **Example:** *"University of Delhi ke History Department"* → `University of Delhi` as **ORG**, `History Department` as **ORG**

Handling Transliteration Variations

Standardization Principle

- Annotate the same entity consistently across all transliteration or spelling variants.

Common Variations

- `Shahrukh` / `Shah Rukh` / `Sharukh` → **PER**
- `Dilli` / `Delhi` → **LOC**

Phonetic Variations

- `Varanasi` / `Waranasi`
- `Ashok` / `Asok`
- `Calcutta` / `Kolkata`

Script Mixing Within Entities

- **Example:** *" college"* → Entire phrase is **Org** if referring to `Ram College`

Culturally-Specific Entities Classification

Festival and Cultural Events (Cul)

- `Diwali` / `Eid` / `Durga Puja` / → **Cul**

Religious and Spiritual References

- `Shiva` / `Krishna` / → **Cul** (If referring to deity; if human, annotate as **Per**)

Traditional Practices and Rituals

- `Kathak` /(dance form), `Mehndi` / (ritual) → **Cul**

Regional Cuisine and Food Items

- `Biryani` / `Chaat` / → **Cul**
- **Exception:** `Delhi Biryani House` → **ORG**

13 Special Cases and Ambiguities

Ambiguous Entities

- Use context to assign the most specific and relevant entity type.

Metonymic Uses

- **Example:** *"White House ne kaha"* → `White House` is **ORG**, not **LOC**

Social Media Handles and Usernames

- `@narendramodi` → **PER**
- `@BJP4India` → **ORG**

Annotation Process Guidelines

1. Read the entire text for context.
2. Identify potential named entities.
3. Resolve any ambiguities using linguistic or contextual clues.
4. Mark entity boundaries precisely and consistently.
5. Assign the correct entity type (PER, ORG, LOC, CUL, PROD, etc.).
6. Document cases where annotation is uncertain for review.
7. Ensure consistency in tagging across all instances.

Example Annotations

- *"main kal mumbai airport gaya tha"* → `mumbai airport` is **LOC**
- *"dilli university mein admission liya"* → `dilli university` is **ORG**
- *"kal holi celebrate karenge"* → `holi` is **CUL**
- *"State Bank of India ke Delhi branch"* → `State Bank of India` is **ORG**, `Delhi` is **LOC**

14 Code Snippet

Listing 1: Feature Extraction for CRF Model

```
def word2features(sent, i):
    """Extract features for word at position i in sentence sent"""
    word = sent[i][0]

    features = {
        'bias': 1.0,
        'word.lower()': word.lower(),
        'word[-3:]': word[-3:],
        'word[-2:]': word[-2:],
        'word.isupper()': word.isupper(),
        'word.istitle()': word.istitle(),
        'word.isdigit()': word.isdigit()
    }

    # Features for previous word
    if i > 0:
        word1 = sent[i-1][0]
        features.update({
            '-1:word.lower()': word1.lower(),
            '-1:word.istitle()': word1.istitle(),
            '-1:word.isupper()': word1.isupper()
        })
    else:
        features['BOS'] = True

    # Features for next word
    if i < len(sent)-1:
        word1 = sent[i+1][0]
        features.update({
            '+1:word.lower()': word1.lower(),
            '+1:word.istitle()': word1.istitle(),
            '+1:word.isupper()': word1.isupper()
        })
    else:
        features['EOS'] = True

    return features

def sent2features(sent):
    return [word2features(sent, i) for i in range(len(sent))]

def sent2labels(sent):
    return [label for token, label in sent]
```

15 Challenges and Limitations

15.1 Data Challenges

15.1.1 Imbalanced Entity Distribution

Our project encountered significant challenges due to the imbalanced distribution of entity types in the dataset. As evident from the results table, there was substantial variation in support counts across different entity types:

- High-frequency entities (B-Org: 343 instances, B-Loc: 290 instances) received adequate representation.
- Medium-frequency entities (B-Kin: 199 instances, B-Par: 139 instances) had reasonable representation.
- Low-frequency entities (B-Rel: 64 instances, I-Rel: 29 instances) were underrepresented.
- Some entity types (I-Kin, I-Par) had zero occurrences in the test set.

15.1.2 Transliteration Inconsistencies

A major challenge specific to code-mixed NER was the wide variation in transliteration practices. Indian language speakers demonstrate substantial inconsistency when writing Hindi words in Roman script:

- Multiple valid transliterations exist for the same word (e.g. *Delhi*, *Dilli*, *Dillee*).
- Inconsistent vowel representations (e.g. *ji*, *jee*, *g*).
- Variable handling of aspirated consonants (e.g. *Bharat*, *Bhaarat*, *Bhārat*).
- Spelling simplifications based on pronunciation (e.g. *kya*, *kia*, *kyaa*).

These inconsistencies created challenges for feature extraction and entity recognition, particularly for culturally-specific entities. Our model performed best on standardized English entity names and struggled with transliterated Hindi terms.

Furthermore, the same named entity might appear in:

- Pure Devanagari script
- Pure Roman script (with transliteration variations)

Each representation requires different handling strategies, significantly complicating the NER task compared to monolingual settings.

15.2 Model Limitations

15.2.1 Reliance on Lexical Features

The CRF model implemented in this project relies heavily on lexical features, which presents several limitations:

- **Out-of-vocabulary problem:** The model struggles with entities not seen during training, especially transliterated Hindi terms.
- **Feature sparsity:** Many lexical features occur rarely, weakening parameter estimation, especially in code-mixed settings.
- **Script sensitivity:** Lexical features are script-sensitive, resulting in poor generalization across scripts.

More sophisticated approaches like deep learning models with contextual embeddings (e.g. BERT, MuRIL) could address these limitations by learning semantically meaningful representations across languages and scripts.

15.2.2 Weak for Long-range Dependencies

CRF models, while effective for sequence labeling tasks have inherent limitations in capturing long-range dependencies:

- **Limited context window:** Feature extraction focused mainly on neighboring tokens, limiting broader contextual understanding.
- **Entity boundary detection challenges:** Multi-token entities, particularly those crossing language boundaries were often misclassified.
- **Discourse-level coherence:** The model lacks the ability to maintain entity consistency across sentences.
- **Coreference inability:** Pronouns or partial references could not be resolved back to antecedent entities.

Neural architectures like BiLSTM-CRF, Transformers, or attention-based models could better capture long-range and contextual dependencies.

15.3 Evaluation Limitations

15.3.1 Metrics May Not Capture Code-mixing Challenges

Standard NER evaluation metrics (precision, recall, F1-score) do not capture several unique challenges in code-mixed NER:

- **Transliteration sensitivity:** Different spellings of the same entity are treated as distinct, penalizing models twice.

- **Cultural knowledge dependency:** Metrics do not account for the cultural familiarity required to identify some entities.

More nuanced evaluation frameworks specific to code-mixed text - e.g. based on script, transliteration clusters or code-mixing indices could offer better insights.

15.3.2 Limited Test Examples for Rare Entities

The evaluation of our model was hindered by sparse representation of certain entity types in the test set:

- **Zero-shot evaluation:** Some entity types (e.g. I-Kin, I-Par) had no occurrences in the test set.
- **Statistical unreliability:** Categories with very few examples (e.g. I-Rel: 29, I-Per: 73) yield unstable metrics.
- **Inability to assess generalization:** It's hard to determine model generalization on rare types due to insufficient data.
- **Semantic coverage gaps:** Certain semantic subtypes may dominate a category, biasing performance scores.

Future work should prioritize balanced test sets, stratified sampling and robust evaluation methods such as cross-validation.

16 Error Analysis of Code-Mixed NER Model

16.1 Overview

This section contains a detailed analysis of the errors and performance metrics of our Named Entity Recognition (NER) model for Hindi-English code-mixed text. The analysis is based on the testing results from 565 sentences containing 11,662 tokens.

16.2 Overall Performance

- Token Accuracy: 93.1% (10,859 correct out of 11,662 tokens)
- Macro Average F1-Score: 0.568
- Weighted Average F1-Score: 0.925

16.3 Entity-wise Error Analysis

16.3.1 High-Performing Entities

1. **Other (Non-entity tokens)**
    - Precision: 0.954
    - Recall: 0.986
    - F1-Score: 0.970
    - Support: 9,903 tokens
    - Analysis: Excellent performance on non-entity tokens, which is crucial for overall accuracy
2. **B-Loc (Location Beginnings)**
    - Precision: 0.888
    - Recall: 0.741
    - F1-Score: 0.808
    - Support: 290 tokens
    - Analysis: Strong performance in identifying location names with high precision
3. **B-Par (Particle Beginnings)**
    - Precision: 0.805
    - Recall: 0.892
    - F1-Score: 0.846
    - Support: 139 tokens
    - Analysis: Good balance between precision and recall for particles

16.3.2 Underperforming Entities

1. **B-Cul (Cultural Beginnings)**
    - Precision: 0.531
    - Recall: 0.276
    - F1-Score: 0.364
    - Support: 123 tokens
    - Analysis: Poor performance especially in recall, indicating some missed cultural terms
2. **I-Cul (Cultural Continuations)**
    - Precision: 0.722
    - Recall: 0.153
    - F1-Score: 0.252
    - Support: 85 tokens
    - Analysis: Low recall suggests difficulty in identifying multi-token cultural terms
3. **B-Rel (Religious Beginnings)**
    - Precision: 0.520
    - Recall: 0.406
    - F1-Score: 0.456
    - Support: 64 tokens
    - Analysis: Moderate performance with room for improvement

16.3.3 Zero-Performance Entities

1. **I-Kin (Kinship Continuations)**
    - Precision: 0.000
    - Recall: 0.000
    - F1-Score: 0.000
    - Support: 0 tokens
    - Analysis: No instances in test data. This is expected as:
        - Kinship terms rarely appear as multi-token entities
        - Most kinship terms are single tokens (e.g., “papa”, “mummy”)
        - The scraped test dataset’s limited size (11,662 tokens) didn’t contain any tokens with this tag

2. **I-Par (Particle Continuations)**
    - Precision: 0.000
    - Recall: 0.000
    - F1-Score: 0.000
    - Support: 0 tokens
    - Analysis: No instances in test data. This is expected because:
        - Particles in Hindi-English code-mixed text are typically single tokens
        - Multi-token particles are extremely rare in natural language
        - The scraped test dataset's limited size didn't contain any tokens with this tag

16.4 Common Error Patterns

1. **Cultural Terms**
    - False negatives (missed cultural terms)
    - Difficulty in identifying cultural concepts that span multiple tokens
    - Confusion between cultural and religious terms
2. **Location Names**
    - Good precision but moderate recall
    - Some confusion between locations and organizations
    - Difficulty with compound location names
3. **Organization Names**
    - Moderate performance across all metrics
    - Challenges with identifying organization boundaries
    - Confusion with location names in some cases

16.5 Recommendations for Improvement

1. **Data Collection**
    - Increase the number of examples for cultural and religious terms
    - Add more multi-token entity examples
    - Include more diverse code-mixed patterns
2. **Feature Engineering**
    - Add language-specific features for better entity boundary detection

- Incorporate contextual features for cultural and religious terms
- Consider adding word embeddings for better semantic understanding

3. **Model Architecture**
    - Experiment with different CRF parameters
    - Consider adding a language identification component
    - Explore hybrid approaches combining CRF with neural networks

16.6 Inference

The model shows strong performance on common entities and non-entity tokens but struggles with cultural and religious terms. The zero performance on I-Kin and I-Par is due to their absence in the test data, which is expected given their rarity in natural language. Future improvements should focus on enhancing performance for cultural and religious terms while maintaining the current strong performance on other entity types.

17 Future Directions and Applications

17.1 Future Directions

There are several avenues for extending the work presented in this project:

- **Deep Learning Models:** While our CRF-based model offers a strong baseline, future work can explore deep learning approaches such as BiLSTM-CRF or transformer-based models like mBERT and XLM-R. These models have shown superior performance in multilingual and low-resource settings.
- **Joint Modeling:** Integrating tasks such as language identification, part-of-speech tagging, and entity normalization with NER through multi-task learning could improve entity recognition in code-mixed scenarios.
- **Corpus Expansion:** The current annotated corpus can be expanded to include data from various domains such as customer reviews, technical support chats, and informal social media conversations to increase coverage and robustness.
- **Active Learning and Semi-Supervised Learning:** Employing active learning could reduce manual annotation effort, while semi-supervised learning techniques can help utilize large amounts of unannotated data effectively.
- **Error-specific Model Refinement:** Focused improvements based on error analysis, especially for multi-token entities and cultural/religious terms, could lead to better performance for these challenging categories.

17.2 Applications

The tools and resources developed in this project have broad applicability across various real-world scenarios:

- **Social Media Monitoring:** Accurate entity recognition in code-mixed social media posts can aid sentiment analysis, trend detection, and misinformation tracking.
- **Customer Support Automation:** Many customer service conversations involve code-mixing. Improved NER systems can enhance chatbot understanding and automate issue classification.
- **Voice Assistants and Multilingual Interfaces:** NER in code-mixed language is crucial for building effective voice assistants and user-friendly interfaces in multilingual regions.
- **Cross-lingual Information Retrieval:** Better recognition of entities in code-mixed queries can improve the performance of search engines and recommendation systems in multilingual environments.
- **Language Technology for Low-resource Languages:** The methodologies developed can be adapted to other language pairs, contributing to inclusive and equitable NLP research.

18 Conclusion

This project addressed the challenging task of Named Entity Recognition (NER) in Hindi-English code-mixed text. We developed a comprehensive annotation scheme, created a substantial annotated corpus, and implemented a CRF-based NER system that achieved reasonable performance on most entity types.

Our analysis revealed the specific challenges of NER in code-mixed text, particularly for cultural and religious entities and for multi-token entities. The performance varied significantly across entity types, with the best results for particles and locations and the worst for cultural and religious terms.

The resources created in this project, including the annotation guidelines and the annotated corpus, will be valuable for future research on code-mixed Natural Language Processing (NLP).

While our CRF-based approach provides a strong baseline, future work could explore more sophisticated models, particularly deep learning approaches that can better capture the complexities of code-mixed text. Pretrained multilingual transformers, such as mBERT or XLM-R, offer promising directions due to their ability to model contextual and cross-lingual semantics. Incorporating language identification as an auxiliary task or leveraging joint models for entity recognition and normalization may further improve performance.

Additionally, future efforts could benefit from extending the corpus to cover more domains, including social media, customer support, and informal conversations, to improve generalizability. Active learning strategies could also be employed to enhance annotation efficiency and focus on ambiguous or underrepresented cases.

In conclusion, this work lays a strong foundation for NER in Hindi-English code-mixed settings and highlights the critical need for tailored linguistic resources and robust models to tackle the nuances of multilingual and informal language use. By addressing these challenges, future research can advance the development of more inclusive and effective NLP tools for diverse language users.

References

[1] Gustavo Aguilar et al. "Named Entity Recognition on Code-Switched Data: Overview of the CALCS 2018 Shared Task". In: *Proceedings of the Third Workshop on Computational Approaches to Linguistic Code-Switching*. 2018, pp. 138–147. DOI: 10.18653/v1/W18-3219.

[2] Irshad Ahmad Bhat et al. "Universal Dependency parsing for Hindi-English code-switching". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2018, pp. 987–998. DOI: 10.18653/v1/N18-1090.

[3] Khyathi Raghavi Chandu et al. "Language informed modeling of code-switched text". In: *Proceedings of the Third Workshop on Computational Approaches to Linguistic Code-Switching*. 2018, pp. 92–97. DOI: 10.18653/v1/W18-3211.

[4] Amitava Dey and Sivaji Bandyopadhyay. "WEINLP at CALCS 2018: Named Entity Recognition over Code-switched Data". In: *Proceedings of the Third Workshop on Computational Approaches to Linguistic Code-Switching*. 2019, pp. 92–97. DOI: 10.18653/v1/W18-3211.

[5] Prathyusha Geetha, Khyathi Chandu, and Alan W. Black. "Tackling Code-Switched NER: Participation of CMU". In: *Proceedings of the Fifth Workshop on Computational Approaches to Linguistic Code-Switching*. 2021, pp. 66–70. DOI: 10.18653/v1/2021.calcs-1.9.

[6] Simran Khanuja et al. "MuRIL: Multilingual Representations for Indian Languages". In: *arXiv preprint arXiv:2103.10730* (2021).

[7] Radhakrishna Murthy, Mitesh M. Khapra, and Pushpak Bhattacharyya. "A Dataset for Named Entity Recognition in Indian Languages". In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC)*. 2016, pp. 2801–2806.

[8] Barun Ghosh Patra, Dipankar Das, and Amitava Das. "Sentiment Analysis of Code-Mixed Indian Social Media Text". In: *Proceedings of the 11th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*. 2018, pp. 116–122. DOI: 10.18653/v1/W18-6220.

[9] Abhijit Pratapa, Monojit Choudhury, and Sunayana Sitaram. "Word embeddings for code-mixed language processing". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 2018, pp. 3067–3072. DOI: 10.18653/v1/D18-1344.


THANK
YOU