NLP APPLICATIONS

NER DETECTION IN CODE-MIXED DATA

Mentor: Prof. Ponnurangam Kumaraguru

Team Members:

- Akshat Maheshwari, 20161024
- Sourav Kumar, 20161072

Introduction

What Is Named-Entity Recognition?

- NER involves identification of proper names in texts, and classification into a set of predefined categories of interest.
- Three universally accepted categories:

Person :	Albert Einstein
Location :	Hyderabad
Organization :	Google

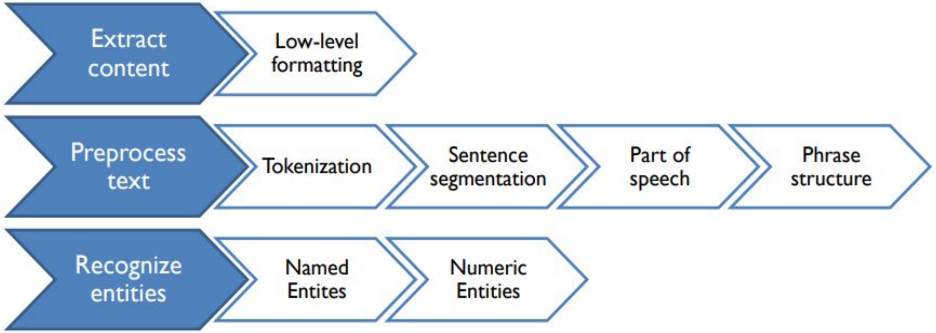
- Other common tasks: recognition of date/time expressions, measures (percent, money, weight etc), email addresses etc.
- Other domain-specific entities: names of Drugs, Genes, medical conditions, names of ships, bibliographic references etc.

Why do we need NER Detection for Code-Mixed Data?

- While growing code-mixed content on Online Social Networks (OSNs) provides a fertile ground for studying various aspects of code-mixing, the lack of automated text analysis tools render such studies challenging.
- To meet this challenge, a family of tools for analyzing code-mixed data such as language identifiers, parts-of-speech (POS) taggers, chunkers have been developed.
- The shortness of micro-blogs makes them hard to interpret. Consequently, ambiguity is a major problem since semantic annotation methods cannot easily make use of co-reference information.
- Micro-texts contain less amount of text with unorthodox capitalization that can turn out to be difficult for NER detection.

Current State of Art

General NER Detection For Monolingual Data



Current State of Art (Continued...)

NER Detection for Code-Mixed Data

- Paper - Link
- This approach uses explicit modelling of the given tweet into English Language Model (ELM) and Hindi Language Model (HLM)
- Features taken into account:
  - Token Based Features
  - Affixes
  - Character Based Features
  - Language Based Features
  - Syntactic Features
  - Tweet Capitalization Features

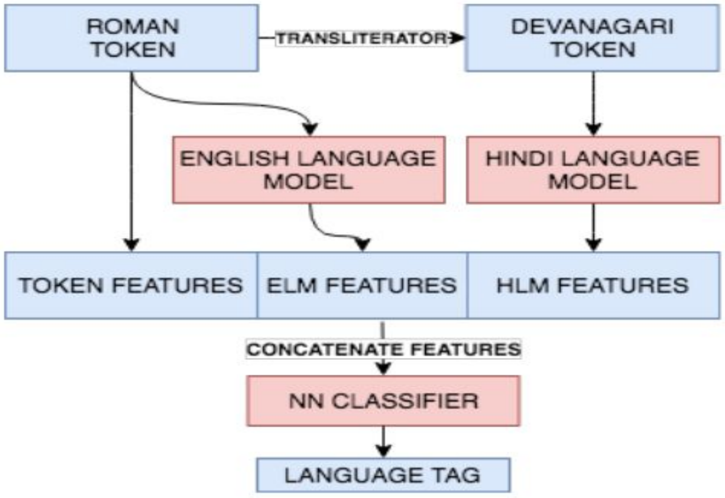
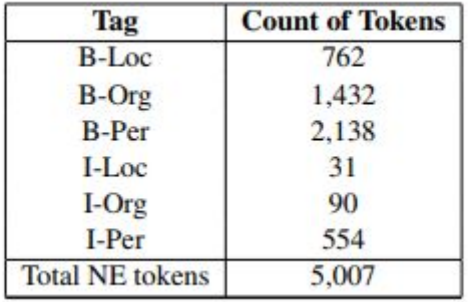


Figure 1: Different steps of our language identification pipeline. We extract features using language models trained on monolingual corpora, and train a classifier based on these features.

Dataset

- We have used the Twitter Dataset for the project
- Used **twitterscraper API** for scraping tweets, but these tweets had to be annotated manually
- So we used the already annotated dataset from this paper : Link



Tag	Count of Tokens
B-Loc	762
B-Org	1,432
B-Per	2,138
I-Loc	31
I-Org	90
I-Per	554
Total NE tokens	5,007

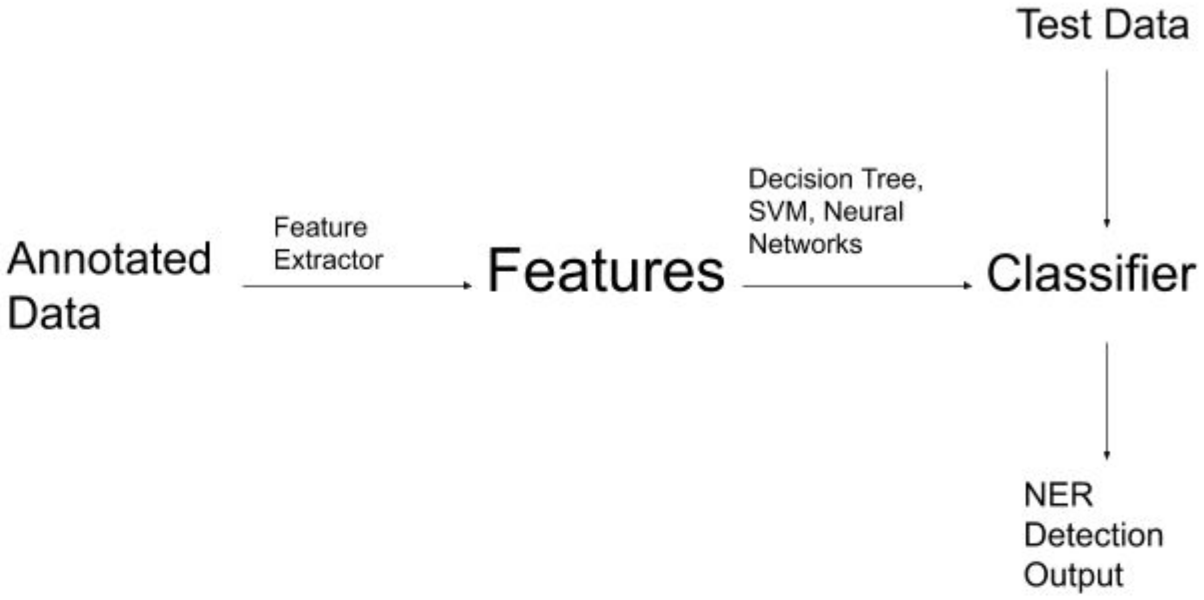
APPROACH

- We follow similar procedure to the State of the art paper, but the main difference is that we have not done explicit Language modelling, rather we have extracted features (as is described in the paper we have used for the dataset).

- We then extract more number of features, which themselves are somewhat responsible for some kind of language modelling (or we can say Named Entity detection).

- After that, we used the different non-neural and neural network models for classification tasks.

Model Pipeline



Features Extracted

- Character N-Grams
- Word N-Grams
- Capitalization
- Mentions and Hashtags
- Numbers in Strings
- Previous Word Tags
- Common Symbols

Experimental Settings

Different Approaches Used

Statistical Approaches

- Decision Tree Classifier
- Naive Bayes Classifier
- SVM Classifier
- Random Forest Classifier
- CRF (Context Random Field) Classifier

Neural Network Approaches

- LSTM Classifier
- Bidirectional Classifier
- GRU Classifier
- Bidirectional GRU Classifier

Results

Statistical Approaches

Classifier	Precision	Recall	F1-Score
Decision Tree Classifier	0.94	0.94	0.94
Naive Bayes Classifier	0.91	0.75	0.82
SVM Classifier	0.91	0.75	0.82
Random Forest Classifier	0.93	0.94	0.92
CRF Classifier	0.85	0.48	0.60

Neural Network Approaches

Model	Train Accuracy	Train Loss	Validation Accuracy	Validation Loss
LSTM	0.9224	0.3018	0.9394	0.2719
Bidirectional LSTM	0.9226	0.3092	0.9392	0.2768
GRU	0.9246	0.2941	0.9324	0.2845
Bidirectional GRU	0.9241	0.2958	0.9153	0.2825

NOTE: We have used Adam’s Optimizer with Categorical Cross-Entropy Loss

Why Neural Network Approach is preferred over Statistical Approach?

Statistical approaches generally use N-gram models, whereas Neural Network models are models that take into account the previous context of the sentence.

For example,

{Aditya Birla} {started the} {Aditya Birla Group}.

In this, the First Phrase is the name of a person (Aditya Birla) whereas the last phrase is the name of a company (Aditya Birla Group). So here is the case where our general statistical approach can fail, hence shifting to neural models would be helpful.

Thank You!