NER USING CRF

Ayush Kumar Gupta

Intro

Project Overview

As Mentioned in Previous Slide, I had to learn CRF, Feature Functions and had to make a Pipeline from Chunking,POS to NER all through CRF. I also had to do Annotations. I had to test on two different data andcheck their results.

Contents

Table Of Contents

Chapter 1

How I Proceeded?

My Goal was to test use CRF Feature Functions for English which includes many Lexical properties for NER, such as Capitalization, Position in a sentence, Articles etc. with a rich premade-gazeteer.

But after Presentation, My Topic changed to Hindi, so I researched about CRF, Feature Functions, How they work, Formulaes, Linear CRF and How to Implement on Hindi.

Summarise Learnings

- CRF is Conditional Random Field model, which works on its Template and Its Feature Functions, whereas Template provide information about which parts are dependent on which. Function provide probability the of current node in given nodes on which it is dependent.
- Functions are of form f(**X**,**Y**i-1,**Y**i,i), which have arguments current_val,prev_state,current_state,weight(acc. to length).
- P(**Y**/**X**)=1/**Z** * **e**^Sigma(F(**X**,**Y**i-1,**Y**i,i)), where F(**X**,**Y**i-1,**Y**i,i)= Sigma(**wj*fj**)
- **w** is calculated while training the model.
- The Above Formulaes are for Linear CRF++, in which Current State is only dependent on Previous adjacent Stage.

Problems

My Goal was to test use CRF Feature Functions for English which includes many Lexical properties for NER, such as Capitalization, Position in a sentence, Articles etc. with a rich premade-gazeteer,(Yup Ctrl+C,V).Whereas Hindi had no capitalization, It is Morphologically Rich, Having hard to find good gazeteer, No Particular Word Order, or Articles.

SO I DROPEED USING CRF FEATURE FUNCTIONS FOR NER

About

NER

Named Entity Recognition (NER) is a subtask of information extraction that seeks to locate and classify named entities mentioned in unstructured text into predefined categories such as the names of persons, organizations, locations.

CRF

Conditional Random Fields (CRF) is a type of probabilistic graphical model used for structured prediction, especially in sequence labeling tasks such as named entity recognition (NER), part-of-speech (POS) tagging, and chunking.

Chapter 2

Process

1

NER

Got NER Data from Hugging Face and Trained Model

2

POS

Collected the Data and trained POS Model. Also anotated some POS test.

3

PIPELINE

Connected Pipeline so, If Data is provided,it will return POS ,NER Annotated Data

4

TOKENIZER

To make It complete Text based, one System.

Chapter 3

NAME ENTITY RECOGNITION

STEPS:
IMPLEMENTED USING CRF++
COLLECTED TRAINING DATA WITH 150K LINES OF ANNOTATION

TRAINED MODEL ON PROVIDED DATA

TESTED ON PROVIDED COLLECTED TEST
TESTED ON NEWS DATA (MANUALLY)
TESTED ON STORY DATA (MANUALLY)

MADE PIPELINE TO PROVIDE TEXT->NER

PARTS OF SPEECH

STEPS:
IMPLEMENTED USING CRF++
COLLECTED TRAINING DATA WITH 100K LINES OF ANNOTATION

TRAINED MODEL ON PROVIDED DATA

TESTED ON ANNOTATED TEST (MANUALLY)
TESTED ON NEWS DATA (MANUALLY)
TESTED ON STORY DATA (MANUALLY)

MADE PIPELINE TO PROVIDE TEXT->NER

PIPELINE

STEPS:
WROTE JUPYTER SCRIPT TO TRAIN MODEL WITH JUST PROVIDING ANNOTATED DATA.

WROTE JUPYTER SCRIPT TO ANNOTATE DATA WITH JUST PROVIDING TEXT

IF DATA IS ALREADY ANNOTATED, IT TESTS THE PRECISION,RECALL,F1 SCORE OF MODEL ON ANNOTATED DATA

(PIPELINE CAN BE EXTRAPOLATED TO CHUNKING TOO,IF ANNOTATED DATA IS PROVIDED)

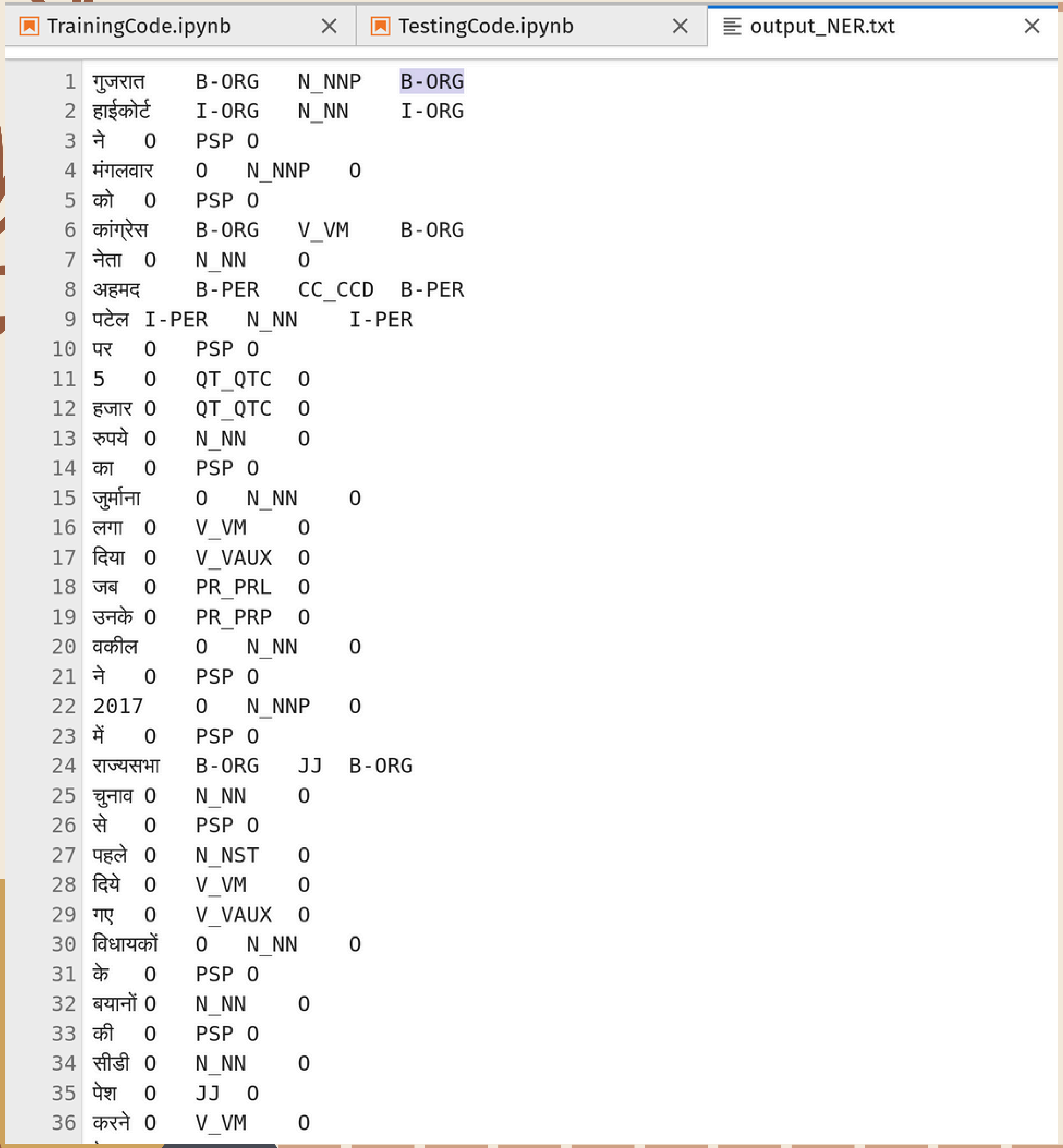
TOKENIZER

STEPS:
WROTE CODE TO CONVERT TEXT INTO SENTENCES
WROTE CODE TO CONVERT SENTENCES INTO WORD IN CONLL FORMAT FOR TESTING/ANNOTATING DATA

(I HAVENT USED REGEX,IT WAS SIMPLE CODE, USING REGEX I CAN REMOVE PUNCTUATIONS OR ONE CAN DO AFTER TOKENIZATION,WITH JUST CTRL+F (ALT+ENTER))

Chapter 4

RESULTS



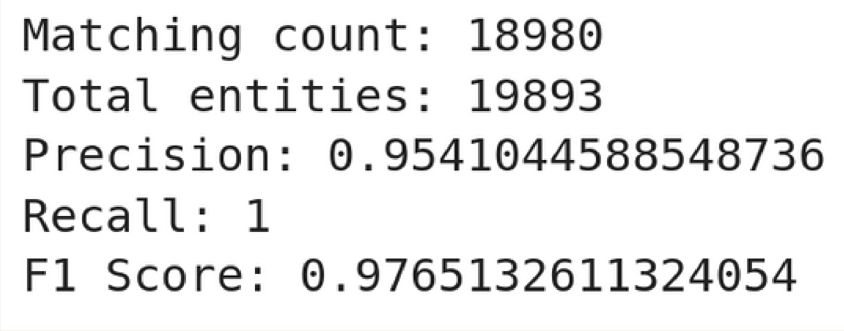
```
गुजरात	B-ORG	N_NNP	B-ORG
हाईकोर्ट	I-ORG	N_NN	I-ORG
ने	O	PSP	O
मंगलवार	O	N_NNP	O
को	O	PSP	O
कांग्रेस	B-ORG	V_VM	B-ORG
नेता	O	N_NN	O
अहमद	B-PER	CC_CCD	B-PER
पटेल	I-PER	N_NN	I-PER
पर	O	PSP	O
5	O	QT_QTC	O
हजार	O	QT_QTC	O
रुपये	O	N_NN	O
का	O	PSP	O
जुर्माना	O	N_NN	O
लगा	O	V_VM	O
दिया	O	V_VAUX	O
जब	O	PR_PRL	O
उनके	O	PR_PRP	O
वकील	O	N_NN	O
ने	O	PSP	O
2017	O	N_NNP	O
में	O	PSP	O
राज्यसभा	B-ORG	JJ	B-ORG
चुनाव	O	N_NN	O
से	O	PSP	O
पहले	O	N_NST	O
दिये	O	V_VM	O
गए	O	V_VAUX	O
विधायकों	O	N_NN	O
के	O	PSP	O
बयानों	O	N_NN	O
की	O	PSP	O
सीडी	O	N_NN	O
पेश	O	JJ	O
करने	O	V_VM	O
```

Given Test Data

NER:

F1 Score:0.97

Precision:0.95



```
Matching count: 18980
Total entities: 19893
Precision: 0.9541044588548736
Recall: 1
F1 Score: 0.9765132611324054
```

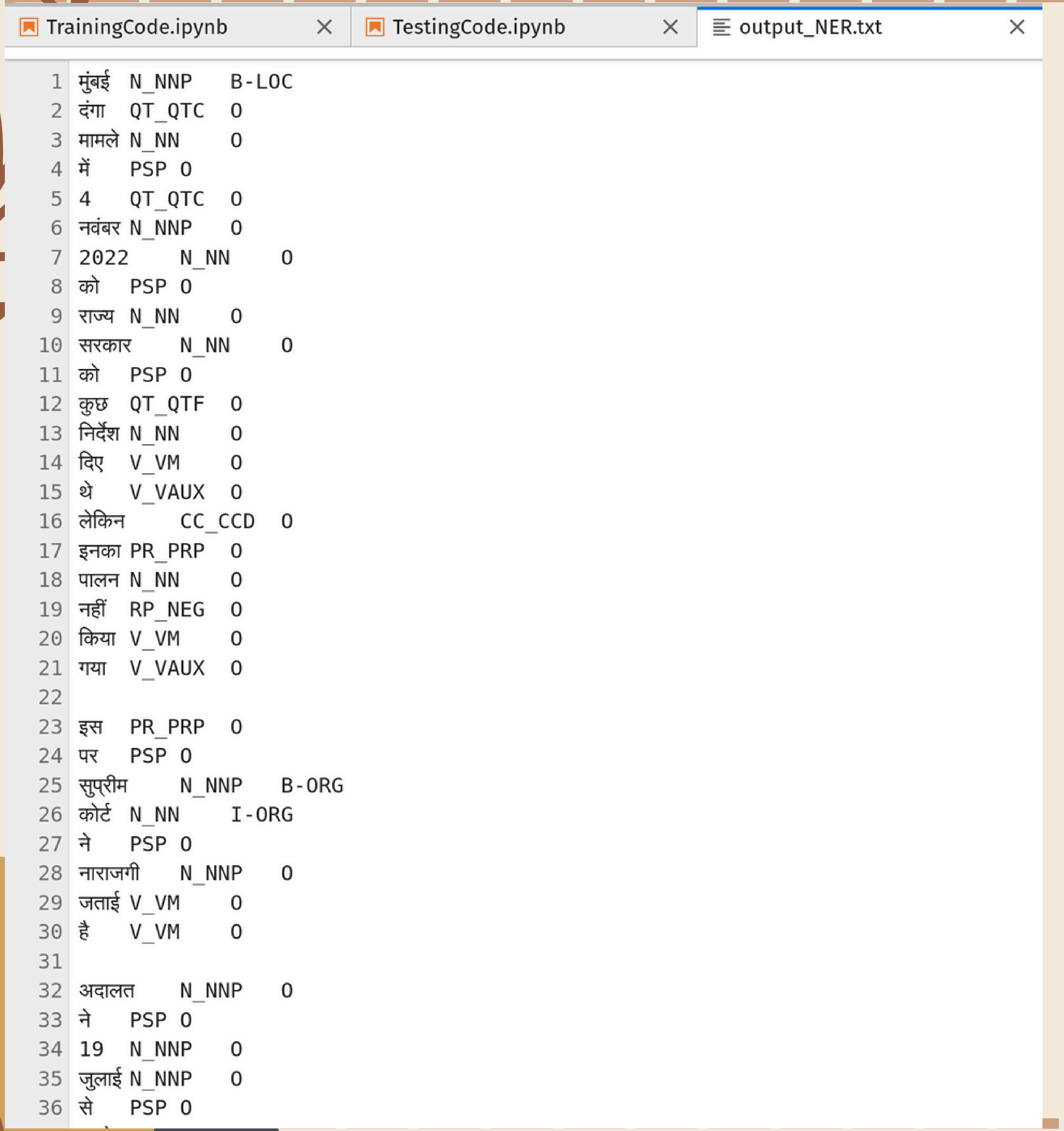
I wonder O are so many so should we consider them in precision or not

POS:

F1 Score:0.93

Precision:0.88

News Test Data



```
मुंबई	N_NNP	B-LOC
दंगा	QT_QTC	O
मामले	N_NN	O
में	PSP	O
4	QT_QTC	O
नवंबर	N_NNP	O
2022	N_NN	O
को	PSP	O
राज्य	N_NN	O
सरकार	N_NN	O
को	PSP	O
कुछ	QT_QTF	O
निर्देश	N_NN	O
दिए	V_VM	O
थे	V_VAUX	O
लेकिन	CC_CCD	O
इनका	PR_PRP	O
पालन	N_NN	O
नहीं	RP_NEG	O
किया	V_VM	O
गया	V_VAUX	O

इस	PR_PRP	O
पर	PSP	O
सुप्रीम	N_NNP	B-ORG
कोर्ट	N_NN	I-ORG
ने	PSP	O
नाराजगी	N_NNP	O
जताई	V_VM	O
है	V_VM	O

अदालत	N_NNP	O
ने	PSP	O
19	N_NNP	O
जुलाई	N_NNP	O
से	PSP	O
```

NER:

F1 Score:0.97

Precision:0.95

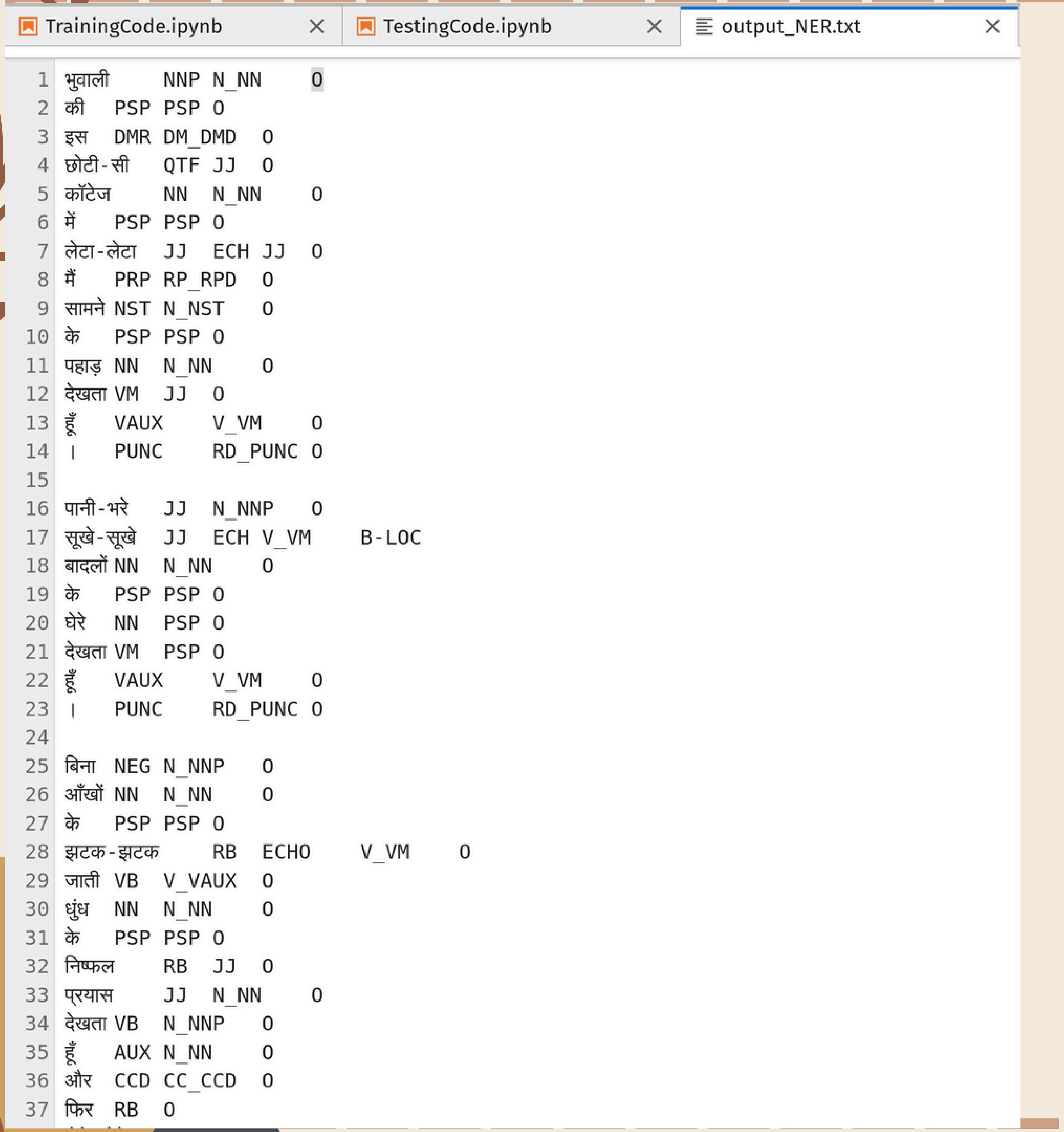
This One Feels True having true Precision

POS:

F1 Score:0.85

Precision:0.91

Story Test Data



```
भुवाली      NNP N_NN      O
की    PSP PSP O
इस    DMR DM_DMD   O
छोटी-सी     QTF JJ   O
कॉटेज       NN   N_NN      O
में    PSP PSP O
लेटा-लेटा   JJ   ECH JJ   O
मैं    PRP RP_RPD   O
सामने NST N_NST    O
के    PSP PSP O
पहाड़ NN   N_NN      O
देखता VM   JJ   O
हूँ    VAUX      V_VM      O
।     PUNC      RD_PUNC O

पानी-भरे    JJ   N_NNP    O
सूखे-सूखे   JJ   ECH V_VM      B-LOC
बादलों NN   N_NN      O
के    PSP PSP O
घेरे   NN   PSP O
देखता VM   PSP O
हूँ    VAUX      V_VM      O
।     PUNC      RD_PUNC O

बिना  NEG N_NNP    O
आँखों NN   N_NN      O
के    PSP PSP O
झटक-झटक         RB   ECHO      V_VM      O
जाती  VB   V_VAUX   O
धुंध   NN   N_NN      O
के    PSP PSP O
निष्फल       RB   JJ   O
प्रयास       JJ   N_NN      O
देखता VB   N_NNP    O
हूँ    AUX N_NN      O
और   CCD CC_CCD   O
फिर  RB   O
```

NER:

F1 Score:0.98

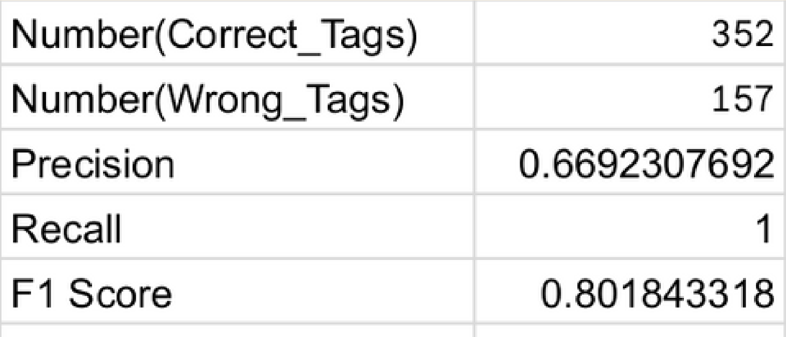
Precision:0.97

But the truth is there are only o tags, and most NER tags are wrong or missed. so Precision is around 0.1

POS:

F1 Score:0.80

Precision:0.67



Number(Correct_Tags)	352
Number(Wrong_Tags)	157
Precision	0.6692307692
Recall	1
F1 Score	0.801843318

What else?

With Chunking Training Data, One can easily ass Chunk Annotator in Pipeline.
With Regex One make Tokenizer better

BIO Tagging is with NER Tagging so no need to do another time

With Some more Time and Effort,
One can make complete Pipeline which takes text and annotate it

The Problem is Precision, It depends on whether Training Data include such type or not
can be fixed by increasing amount of Training Data

SOURCES

Github Repo: https://github.com/AKGIIITH/NER_CRF_Hindi [Read README.md]

THANK YOU