**EMPLOYING CRF ON NER TASKS:**

import nltk

import sklearn\_crfsuite

from sklearn\_crfsuite import metrics

# Download dataset

nltk.download("conll2002")

# Load Spanish dataset (you can also try Dutch)

from nltk.corpus import conll2002

train\_sents = list(conll2002.iob\_sents("esp.train"))

test\_sents = list(conll2002.iob\_sents("esp.testb"))

# ---------------------------

# Feature extraction function

# ---------------------------

def word2features(sent, i):

word = sent[i][0]

postag = sent[i][1]

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(),

"postag": postag,

"postag[:2]": postag[:2],

}

if i > 0:

word1 = sent[i-1][0]

postag1 = sent[i-1][1]

features.update({

"-1:word.lower()": word1.lower(),

"-1:postag": postag1,

})

else:

features["BOS"] = True # Beginning of sentence

if i < len(sent)-1:

word1 = sent[i+1][0]

postag1 = sent[i+1][1]

features.update({

"+1:word.lower()": word1.lower(),

"+1:postag": postag1,

})

else:

features["EOS"] = True # End of sentence

return features

def sent2features(sent):

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

def sent2labels(sent):

return [label for token, postag, label in sent]

def sent2tokens(sent):

return [token for token, postag, label in sent]

# ---------------------------

# Prepare training/test data

# ---------------------------

X\_train = [sent2features(s) for s in train\_sents]

y\_train = [sent2labels(s) for s in train\_sents]

X\_test = [sent2features(s) for s in test\_sents]

y\_test = [sent2labels(s) for s in test\_sents]

# ---------------------------

# Train CRF model

# ---------------------------

crf = sklearn\_crfsuite.CRF(

algorithm="lbfgs",

c1=0.1,

c2=0.1,

max\_iterations=100,

all\_possible\_transitions=True

)

crf.fit(X\_train, y\_train)

# ---------------------------

# Evaluate model

# ---------------------------

y\_pred = crf.predict(X\_test)

print("=== F1 Score (NER) ===")

print(metrics.flat\_f1\_score(y\_test, y\_pred, average="weighted"))

# Example: show predictions for one sentence

print("\n=== Example Prediction ===")

example\_sent = test\_sents[0]

print(list(zip(sent2tokens(example\_sent), y\_pred[0])))