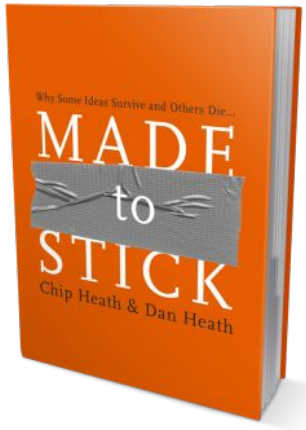
Wiki Based Named Entity Recognition

-By Ayushi Gupta (20162312)

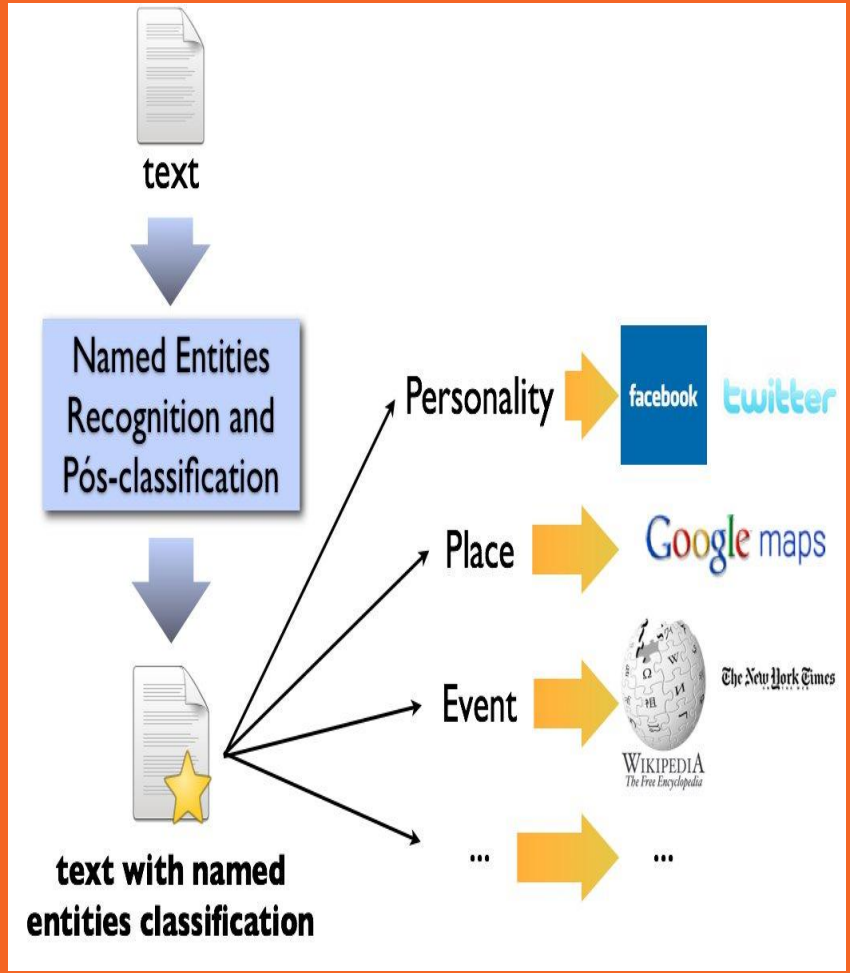
Named Entity

Named-entity recognition (NER) (also known as entity identification, entity chunking and entity extraction) is a subtask of information extraction that seeks to locate and classify elements in text into pre-defined categories such as the names of persons, organizations, locations, expressions of times, quantities, monetary values, percentages, etc.



INPUT: Plain Text, T

OUTPUT: List of named entities in given text.



Dataset:

The dataset used in this project is CoNLL 2003 dataset.

- **Dataset is of the form :-**

  word | Pos tag | Chunk Tag | NER tag.

- The chunk tags and the named entity tags have the format I-TYPE which means that the word is inside a phrase of type TYPE.

- Only if two phrases of the same type immediately follow each other, the first word of the second phrase will have tag B-TYPE to show that it starts a new phrase.

Implementation:

This project was implemented in python and the pseudo code is as follows.

PART A : Training

1. Parsed all the training data - tokenized and found all the bigrams
2. Computed transition and emission probabilities :

P(NEi|NEi-1), P(NEi|wi), P(ti|ti-1),P(wi|ti), P(chi|ti),....

PART B : Testing

1. Parsed the given test data - tokenized
2. Applied the model build during training (Viterbi algorithm)

Experimental Results

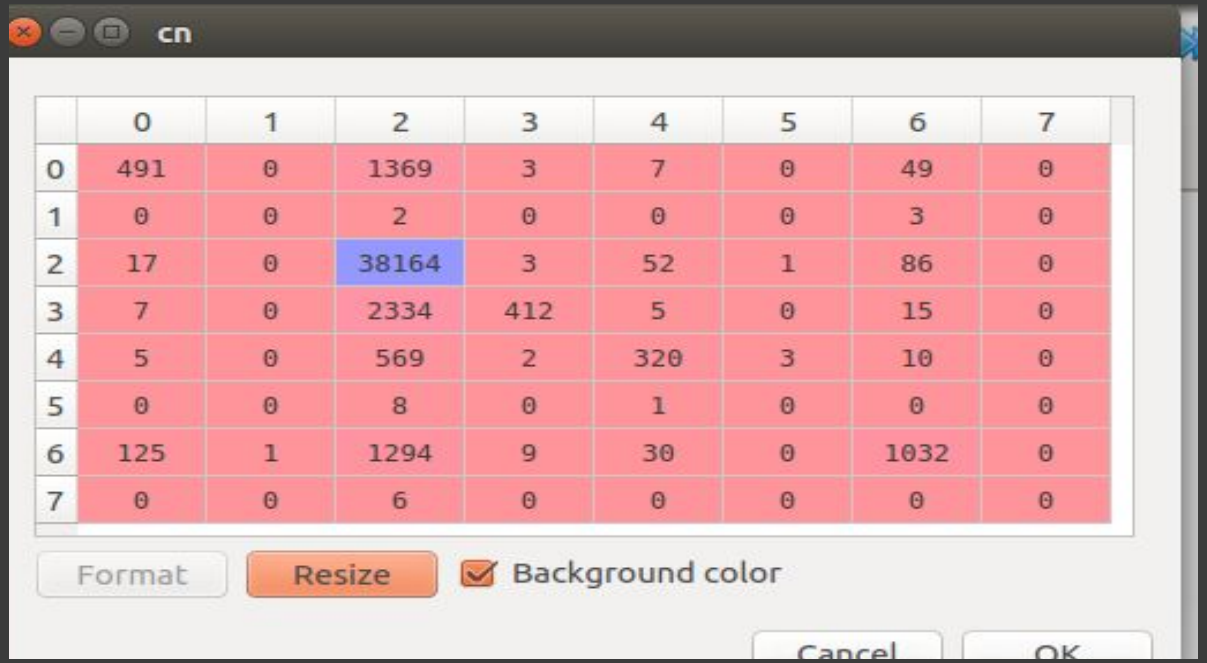
S. No	Inputs	Accuracy
1.	Generalized Method: Emission Probability :$P(W_i\|NE_i)$ Transition Probability : $P(NE_i\|NE_{i-1})$ States : Named-Entities Observations : Words	87.0442554108 %
2.	Emission Probability :$P(W_i,POS_i\|NE_i)$ Transition Probability : $P(NE_i\|NE_{i-1})$ States : Named-Entities Observations : (Word, tag) tuple.	85.7908904921 %
3.	Emission Probability :$P(POS_i\|NE_i)$ Transition Probability : $P(NE_i\|NE_{i-1})$ States : Named-Entities Observations : Words	85.8059653279 %

Various models were build to check the efficiency on CoNLL 2003 dataset.

Accuracy on an average obtained was around 85%

Confusion Matrix:

[u'I-LOC', u'B-ORG', u'O', u'I-PER', u'I-MISC', u'B-MISC', u'I-ORG', u'B-LOC']



	0	1	2	3	4	5	6	7
0	491	0	1369	3	7	0	49	0
1	0	0	2	0	0	0	3	0
2	17	0	38164	3	52	1	86	0
3	7	0	2334	412	5	0	15	0
4	5	0	569	2	320	3	10	0
5	0	0	8	0	1	0	0	0
6	125	1	1294	9	30	0	1032	0
7	0	0	6	0	0	0	0	0

Experimental Results

CoNELL 2003 corpus was selected to compute the efficiency of viterbi algorithm.

Total Number of training sentences : 14041

Total Number of testing sentences : 3453

Efficiency : 87.04%

DEMO GUI