IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.6, Issue.10, Oct. 2018 E-ISSN: 2347-2693

Developing & Deploying Algorithms for Information Extraction using Classification Measures for Named Entity Recognition

Rehan Khan[1*], A.J. Singh[2]

[1,2]Dept. of Computer Science, Himachal Pradesh University, Shimla, India

[*]*Corresponding Author: rehankhan94186@gmail.com*

Available online at: www.ijcseonline.org

Accepted: 15/Oct/2018, Published: 31/Oct/2018

***Abstract*—** The web is full of the content which is either in complete or semi unstructured form and retrieving the essential data out of this unstructured form is very difficult so the concept of the information extraction (IE) keeping in view the necessary parameters becomes highly essential. This paper presents a comparative study for how the problem of information extraction can be handled for a dataset by taking the first step towards IE of named entity recognition (NER) into consideration. Various classifiers/techniques and impact of pipeline on some of them is discussed in this paper for NER and based on the results with keeping the due response time into consideration the classifier/technique of conditional random fields for NER serves out to be the best with an average recall and precision of 0.97 each helping in predicting efficiently of whether a given word is a part of the named entity or not. The automation in the field of medical science for search of the patient for clinical trials from the clinical databases serves to be the most important area of concern at the present time & this paper provides an approach for choosing the technique according to parameters, also discussing the results of the novel algorithmic approach.

***Keywords*—**Information extraction, Natural language processing, Named entity recognition, Conditional random fields.

I. INTRODUCTION

Today the web is full of the data which is generally not available in a well structured form also the pertaining need for the evaluation of the electronic health records which is the freely available text for essential supply of the knowledge for recruitment of the trial of the patients in the medical science serves out to be the most essential concern at present. This data can only be processed and made available to the experts of the medical science on real time basis by the data scientists. [1] Here is the place where the automated engines facilitating information extraction with real time evaluation make the task easy and less cumbersome. Machine learning is the ultimate approach to this automated evaluation system building, as the focus of it lies in training algorithms in order to learn patterns and therefore make predictions from data. For highly agile and efficient system is required a proper information extraction system & in the information extraction domain there lie three important steps:

1. Named Entity Recognition.
2. Relationship Extraction.
3. Template Filing.

This paper dives into the first point of concern i.e., named entity recognition with algorithmic approach in order to explore which technique serves to best in terms of the parameters essentially needed like precision, recall, F-score, accuracy, response time and finally the evaluation with the confusion matrix over the chosen dataset. The entities which the named entity recognition extracts are defined categories like name of organization, person name, location, designation, financial record, time etc. [2] Apart from these simple pre defined entities there are also the context based entities or the user defined entities which need to be considered when dealing with the clinical trial or in general, extraction of some specific terms which represent the elements which have the unique context compared to the rest of the document's text. The named entities in a document could be of any nature from the three classified types:

1. Entity Names: It represents the name of the person, place or an organization of concern and or in general the identity of an element either living or non living.
2. Temporal Expression: The expressions which follow some sequence well defined in the time domain or time related elements like duration of the work, date, year of happening of the event etc. [3]
3. Numerical Expression: A sentence which constitutes of the mathematical symbols, expressions with operators and or only numbers. [4] It could depict the most tangible entities, expressions with financial records, or in general only mathematical expressions.

For a given dataset the constituting named entities are often not the singular simple units of words but chunks of text and therefore it's a need to have some chunking or parsing model which could predict whether a bundle of tokens in the dataset belong to the entity or not. In order to do the chunking or parsing of the text we need some tools and software means in order to achieve our goal of information extraction namely the named entity recognition. [5] The essential tools and related requirements for each are as follows:

1. GATE: General Architecture for Text Engineering is a software tool (rather an infrastructure) for developing and deploying of the essential components that could process human language; generally the document or the dataset fed to it is in the form of a plain text or an xml format file. Its architecture suggests that various elements of the software system which process the text or natural language could be broken down into the essential resources or various types of small components. These resources are:
   i) Language Resources: Generally represent the entities like the corpora, ontology or various lexicons of the text.
   ii) Processing Resources: Generally represent the entities of algorithmic form like n-gram modelers, generators etc.
   iii) Visual Resources: Generally represent the editing or the visualization components which participate in the GUIs.

   Out of these resources only language & processing resources components of the GATE have been used in this work of evaluation along with other essential functionalities of the architecture like ANNIE for building of the pipeline and development of the corpus from the downloaded dataset from ClinicalTrials.gov for the cancer in the region of Shimla. ANNIE (A Nearly-New Information Extraction System) is a processing resources functionality of the GATE which allows the annotations and the pipeline designing function for developing a corpus well suited for information extraction out of the given unstructured dataset. At the end of this paper a novel algorithmic approach [6] is used that functions similarly to the CRF.
2. Python: General purpose programming language, which means it can be used and provide commands to the computer in a variety of ways i.e., in all kinds of platforms and all kinds of applications. It also is a multi-purpose language meaning it can be used for scripting, web design, GUI, etc. Python code is extremely expressive and readable and is cross platform. [7] Python is an interpreted language meaning that whatever code is written gets directly interpreted without any compilation time loss which makes python the best choice for many artificial intelligence scientists. The essential python libraries used in this research work are as follows:
   i) NLTK: An open source python library that holds the modules, linguistic data & documentation necessary for research and the development of natural language processing systems with multi platform support.
   ii) Scikit-learn: This module of the python integrates machine learning algorithms with the world of scientific python packages and aims providing solutions that are simple and easy to the problems of the machine learning. It has many features, metrics and dataset models which make it the state of the art module for python programming and problem solving.
   iii) Numpy: Python library that supports large matrices and multidimensional arrays along with a high level of the support for the mathematical functions in order to operate these arrays and matrices.
   iv) Matplotlib: A python 2-D image plotting library with an interactive environment over multi platforms can be used to generate bar charts, histograms and other variable plots. In this research work it is used to plot confusion-matrix.
   v) Pandas: This library is exclusively used for manipulation of the data and its analysis. Manipulation of the time series and numerical table operations with access to data structures is the key feature of this library that's exclusively used in this research work.
   vi) Eli5: The python library works with sklearn or scikit-learn library of the python in order to show the weights and the predictions of the model, in this research work it is used for CRF model analysis. [2]
   vii) Itertools: This python module implements iterator algebra for a number of the iterator building blocks standardizing the core set of highly efficient and fast memory tools where the elements are treated as unique not on the basis of their values but the essential position they are at. This tool is the core heart for this research work as text is also evaluated with the pattern based estimation analysis and the confusion matrix is null without the efficient use of this tool.
   viii) IMP: This module is used to provide the python environment an essential interface and the mechanism used in order to implement the import statement of any constant or a function.

The organization of paper is as follows section 2 describes the related work, section 3 explains the classifiers and algorithms in use, section 4 explains about technique for comparison and classification measures used, section 5 shows the plots and graphs obtained after comparison and after implementation of Stochastic HeHiCl-DFID algorithm. Section 6 concludes the paper along with the future scope.

II. RELATED WORK

Huang et al. [1] in his paper describes how the word representations via a global context and multiple word prototypes can help to make a better word representation engine for the machine with a relative efficiency of the

machine learning algorithm in concern. The concept of corpora framework representation by means of prototypes is explored via a context resolution mechanism.

McCallum & Sutton [2] in their work gave a wide range of the parameters involved in the CRF algorithm and how this algorithm makes an exploration of the search space based on the parameter estimation so that the resulting model's distribution is set to the best level over the training examples. He gives a complete explanation about the working of the CRF model with due respect of the graphical modeling, feature engineering and others involved in this algorithm.

Bottou et al. [8] enlightened the concept of the natural language processing from the scratch by devising their own dataset and making the corpora out of that for training and testing of the dataset for information extraction and other related corpora activities from the predefined algorithms.
The machine learning work of [9] gives a deep insight on the topic of how to make a structured prediction model for the information extraction piece by piece with the annotation of the corpus to the data modeling in a pipeline order with due consideration of the accuracy and efficiency of the algorithm in concern and how the job of relationship extraction is carried out.

Rother et al. [10] in the work gave an insight on the shape and context modeling with due attention on the topic of the machine learning algorithms utilizing the pattern for prediction of the text and other numerical or temporal context. The multi class objects recognition and classification for segmenting text approach used gave a high level of accuracy compared to any other modeling method.

In [11] the dynamic model of the conditional random fields facilitating the labeling and segmenting of the data for the named entities in the corpora of the large dataset with the help of the factorized probabilistic modeling approach gave the highest accuracy over the dynamic range of operation. The work carried over the segmented data shows the accuracy level of the CRF algorithm over a range of datasets.

III. ALGORITHMS & CLASSIFIERS

A. Lexical Resources & Scikit-learn base classes

A very simple collection of the words and or the phrases along with the context or the associated information like a part of the speech or the simple definition is a lexical resource. Usually the lexical resources are created and therefore enriched by using the text which is secondary to it.
Here the concept of utilizing this simple idea of lexical entities or the most common entities by remembering them for every word and therefore predicting them, for any case left off which is not known just predicting it as 'O'. Scikit-learn (sklearn library in python) base classes & other inbuilt cross-validation tools are used to simply build a simple memorization engine for the work of classification and cross-validation. They are:

1. BaseEstimator:
   It serves to be the very base class for all the estimators that are found in the sklearn library.
   Method:
   get_params() ; set_params().
2. TransformerMixin:
   It serves as the Mixin class for all the available transformers which are found in the sklearn.
   Method:
   fit_transform(X, y={}).
3. Model Selection:
   As the name suggests is used for the selection of an appropriate model and models used so far in this research work are:
   model_select.train_test_split({arrays}) ⟶ It splits the arrays and or matrices into a random train & the request test subset.
   model_select.cross_val_predict() ⟶ For each input data point it generates a cross validation estimate.

B. Perceptron

A very simple classification algorithm that tends to be suitable for simple large scale learning and by default it:

- Do not require any deep down learning.
- Generally is not regularized or to say penalized.
- Is able to update its model only if it finds mistakes.

The resulting models in the perceptron are sparser with a little loss and the models are quite fast to train as the response time recorded by this model outcaste every other technique in this research work.

Perceptron model found in the sklearn is the artificial neuron i.e., an upgrade of the McCulloch-Pitts neuron model concept. Perceptron uses the concept of numerical weights that is a measure of importance for inputs, and a mechanism for learning those weights. [12] Inputs in the case of perceptron are not limited to boolean values and support the real input that makes it highly useful and generalized. A weighted sum of the inputs is taken and the output is set to one only when the actual sum tends to be more than the arbitrary threshold θ. Here instead of manually coding the threshold parameter it is added as one of the inputs with the average weight θ as shown in figure1 that makes it more learnable to the commands provided in the future.

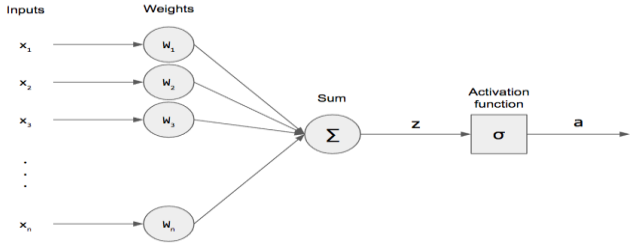

Figure1. Concept Model of the Perceptron

This concept of the perceptron model for predicting the future outcomes based on the inputs of the past is utilized for designing an NER model under the framework of the Sigmoid function the characteristic of which are pre defined in the sklearn library of the python.

C. Supervised Machine Learning Algorithms

Machine learning tends out to be the science and the art relative to the logic for giving computers the ability to learn to make decisions from the data provided to the machine for which the machine is not tentatively or explicitly programmed for. For example: the machine (mobile or immobile) could learn to predict whether an e-mail is a spam or no spam giving the content and the sender details. [13] This example leads to predict using some labels and hence the concept of the supervised learning is explored that uses labelled data. In this research work supervised machine learning approaches are discussed and widely used to sort out a comparative analysis and a bit of the glimpse of the deep learning is discussed for future use at the end of this paper.

D. Stochastic Gradient Descent (SGD Classifier)

It is a simple but still highly efficient approach for the linear models as it is particularly useful for the cases when the number of samples and the number of effective features is very large. The partial_fit method in the sklearn library of the python allows out-of-core learning with application on the web feature. [14] The classes in the sklearn library like SGDRegressor & SGDClassifier provide functionality of fitting the linear models for regression & the classification by using different penalties & convex (different) loss functions. For example by using loss="log", SGDClassifier is able to fit a logistic regression model on the other hand by using loss="hinge" it's able to fit a linear support vector machine (SVM). In the gradient descent the initial values for the coefficients for the function could be 0.0 or a small random value. [7] Cost of the coefficients is evaluated by plugging these coefficients into the desired function and thereof calculating the cost.

Cost = evaluate (f (coefficient))

The derivative, a concept from calculus & referred to as the slope of the function at a given point cost is then calculated. The knowledge of the slope is important so that the direction (sign) to move the coefficient values in order to get a lower cost on the next iteration is known.

Δ = derivative (cost)

As the derivative now gives the knowledge of which direction is downhill, an update to the coefficient values can thereof be made. Alpha (α) i.e., the learning rate parameter must be specified so that it can control the amount of change needed in the coefficients can change on each update.

Coefficient = coefficient – (α x Δ)

This process is repeated over the cycles til the cost of the coefficients comes out to be 0.0 or either close to zero. Gradient descent requires one to know the gradient of the cost function or the function that is to be optimized.

Randomising the order of the training datasets is the first step of the procedure, which is to mix up the order of the updates that are made to the coefficients. Mixing up the order for updates to the coefficients harnesses a kind of the random walk that avoids SGD from getting distracted or stuck and makes more exploration of the state space search.

Learning can be much faster by the use of stochastic gradient descent for training of the datasets and only a small number of passes are required through the dataset to get a good set of coefficients as is evident from the results of the model in this research work.

E. Passive Aggressive Algorithm (PA Classifier)

PA algorithm [15] is mainly used for large scale learning and like perceptron the algorithm does not require a learning rate, the algorithm is a type of the online learning algorithmic model. If a dataset is taken of the form

$$\begin{cases} X = \{\overline{x_0}, \overline{x_1}, \ldots\ldots, \overline{x_t}, \ldots.\} \text{where } \overline{x_i} \in R^n \\ Y = \{y_0, y_1, \ldots\ldots, y_t, \ldots..\} \text{where } y_i \in \{-1, +1\} \end{cases}$$

where the notation t stands for the temporal dimension in the dataset for which the samples could arrive continually for an infinite interval of time and can be drawn from same data generating distribution. The algorithm has a tendency of learning over and over without large parameter modifications, but if these dimensional attributes are drawn from a completely different distribution, the weights will slowly forget the previous one and learn the new distribution over time.

In the equation form for a weight vector w, the prediction function is

$$\tilde{y}_t = \sin(\overline{w}^T . \bar{x}_t)$$

This knowledge of the prediction is widely used in the sklearn library of the python for PA algorithm classifier which can be used to predict the named entity for the evaluation of the NER with effective response time and accuracy that is shown in this research work. [15]

F. Naïve Bayes Classifier

The classifiers which serve to be a collection of the classification algorithms based on the Bayes' Theorem are referred to as the Naive Bayes' Classifiers. NBC is not just a single algorithm but is a family of algorithms where all of them share a common principle for every pair of features being classified is independent of each other.

Bayes' Theorem can be used to find the probability of an event occurring, given the probability of another event that

has already occurred. Mathematically for two events A & B the theorem is defined as follows:

$$P(A|B) = \frac{P(B \mid A)\, P(A)}{P(B)}$$

In the algorithm the logic lies is to:

- Find the probability of an event A given that the event B is true and this event B is termed as evidence.
- Take P(A) as the priori of A that is the probability of an event before the evidence is seen where this evidence is the attribute value of an instance which is unknown.
- Have P(A|B) as a probability with great priority of event B which is termed as the probability of an event after the evidence is seen.

All the logics of the probabilistic model for an event or to say for this research work the probability of the lemmas and the corpus token values are explored to extract the essential dataset values and in order to achieve this the dataset is divided into two parts i.e., feature matrix and the response vector. [16] Feature matrix constitutes of all the vectors or only rows of the dataset in which each vector contains the value of the dependent features in the matrix. Similarly, the response vector holds the values of a class variable i.e., a sort of the prediction or the output for each row of this feature matrix with the NB's fundamental assumption that each feature is able to make an equal & independent contribution to the outcome.

G. Random Forest Classifier & Pipeline

RFC is an ensemble algorithm meaning that it combines more than one algorithm with same or different kind for classifying objects. RFC can create a set of decision trees from randomly selected subset of training set following which it then aggregates the votes from different decision trees in order to decide the final class of the test object. In case if a training set is given of the form: [X_1, X_2, X_3, X_4] with the corresponding labels as [L_1, L_2, L_3, L_4]. RFC creates three decision trees taking the input of subset for example;

- [X_1, X_2, X_3]
- [X_1, X_2, X_4]
- [X_2, X_3, X_4]

RFC therefore predicts based on the majority number of votes from each of the decision trees made, that works quite well because a single decision tree is always prone to noise but by aggregating many decision trees reduces the effect of noise giving more accurate results. The subsets in different decision trees created by the PFC may have a tendency to overlap. Another way of random forest to problem solving is by applying the concept of weights for considering the impact of result from any decision tree. In this case the tree with higher error rate is given a low weight value and vice versa that could just increase the decision impact of trees with low error rate.

In the sklearn RF is a kind of a meta-estimator that fits to a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. The sub-sample size in the case of the RFC model is always the same as that of the original input sample size but the samples are drawn with replacement only if the value of the bootstrap is equal to true (i.e., the default condition). [17]

The dynamic nature of the RFC makes it as a state of the art for designing highly efficient information extraction systems. The work of [3] inspired to make a pipeline for the RFC in order to check the enhancement provided by the pipeline in terms of accuracy and precision but the pipelining and extraction on a single platform of NLTK costed for the average time response to increase drastically, which led to a more efficient concept of CRF.

H. Conditional Random Fields

In [6] the theoretical modelling of the algorithms for state space search were proposed and in the conclusion on comparison it was found that the random walk makes the more exploration of the state space, now taking every dataset with named entities as the state space the concept of CRF serves to be the most efficient [18] as is evident from the results with the logical part of it discussed exclusively here.

Denoting x = (x_1, x_2,……., x_m) in a dataset as the input sequence or the words of a sentence and s = (s_1, s_2,….., s_m) as the sequence of output states or the named entity tags, then the conditional random field models the conditional probability as

$$p(s_1, s_{2,\ldots\ldots,} s_m | x_1, x_2, \ldots\ldots, x_m)$$

This is obtained by defining a feature map as

$$\Phi\ (x_1, \ldots, x_m, s_1, \ldots s_m) \in R^d$$

and this maps an entire input sequence x which is paired with an entire state sequence s to a d-dimensional feature vector after this a modelling of the probability as a log-linear model is done with the parameter vector

$$w \in R^d$$

$$p(s|x; w) = \frac{\exp(w.\Phi(x, s))}{\sum_{s'} \exp(w.\Phi(x, s'))}$$

where, s' has a range over all possible output sequences. Taking a set of n labelled examples $\{(x^i, s^i)\}_{i=1}^n$ for the estimation of w, the log-regularized function L is therefore defined as

$$\sum_{i=1}^{n} \log p(s^i|x^i; w) - \frac{\lambda_2}{2}\|w\|_2^2 - \lambda_1\|w\|_1.$$

The operators $\frac{\lambda_2}{2}\|w\|_2^2$ and $\lambda_1\|w\|_1$ force in the respective norm the parameter vector to be small and this ultimately penalizes the model complexity and is known as

regularization. [13] Now this regularization can be enforced to be more or less depending upon the λ_1 & λ_2 parameters.
The new parameter vector w* is now estimated to be

$$w^* = arg.max_{w \in R^d} L(w)$$

Estimating this vector w* can find the most likely tag a sentence s* for a sentence x by the equation

$$s^* = \arg\max_s p(s|x;w^*)$$

This mechanism of vector optimization following the maximum rating over the given bound for the random walk derives the most positive response for the named entities in the search space for the given dataset.

IV. METHODS & TECHNIQUES

A. Making your own Corpus for Dataset evaluation

The making of the dataset is a time consuming system as the relative annotation, chunking and tokenizing of the data entries consumes a lot of time and requires a high end facilitating system. The dataset for evaluating the named entity using the algorithm [6] (only the part of random walk) was downloaded from ClinicalTrials.gov official website which serves as a repository of all the medical content uploaded online got from various institutions where the patients are undergoing the clinical trail under certain conditions with the current status of the patients trial. The NER here is facilitated by uploading the document in the form of an XML or simply a plain text file as GATE supports these two formats only.

GATE has a very powerful tool known as ANNIE through which the information extraction tool which is inbuilt could make a series of moves to get a well organized corpus for further evaluation. The file used in this research work was of XML format which gave the annotation sets as location, status, person, organization and query report. A bunch of the development sets like co-reference editor, and annotation sets helps to make a list of the co-reference items which could be used for developing a corpora for deploying the algorithm for evaluating the named entities. The corpora is then populated for the essential fields which are found the most for the annotation set like in this case the words like history of (); cancer treatment ongoing stats (); prior () for () found the most frequent use as seen by using the Quality Assurance functionality of the GATE & by using the annotation schema the text populated corpora was made. [19]

The dataset chosen was for the cancer of the people residing in the region of Shimla within the 300 miles parametric range that possessed details having 273 fields of information. The taggers and parsers of the CREOLE plugin of GATE gave the results by creating the datastore when using the Processing Resources application module for ANNIE's information extraction system. The pipeline module that is within the GATE developer made the pipeline for evaluating this chunked and parsed dataset with annotation fields. This well organised corpus is then uploaded to the Python interpreter by importing csv file reader from nltk.corpus. Now by using the Python commands this corpus is evaluated by giving an average precision and recall of 0.43 which is quite well as only a part of the algorithm is used for NER evaluation without having ontology sets defined and the net-algorithm evaluation be the future scope.

B. Preprocessed NER Dataset

A pre-processed dataset GMB (Groningen Meaning Bank) corpus from official website [20] having CSV format is taken that's having 4 columns i.e., sentence number, words, part of speech (POS) & tag for evaluating and comparing various algorithms for information extraction mainly focusing on the named entity recognition. The dataset having a number of random sentences with chunked and tokenized sets of words is processed under various algorithmic approaches mentioned in the part III of this paper, the results of which are mentioned in part V. The procedure and libraries of the Python imported necessarily for evaluation purpose in the Python interpreter are as follows:

import imp	#necessary for implementing python libraries for python 3.6 and later versions
import pandas as pd	#importing library for evaluation of data and its entries
import itertools	#importing library for pattern learning tools
import numpy as np	#importing library for implementation of large mathematical constructs for matrices as and when required
import matplotlib.pyplot as plt	#importing library for a 2-D plotting for confusion matrix
from sklearn import svm, datasets	#importing library for building an vector modulator for developing confusion matrix out of the dataset entries
from sklearn.base import BaseEstimator, TransformerMixin	#utilized here for making a simple memory element for recognizing of the essential patterns
from sklearn.feature_extraction import DictVectorizer	#for making a word vector map in order to apply fit_transform for training set under the sparse matrix
from sklearn.model_selection import cross_val_predict	#required to cross validate the trained and tested dataset entries from the scikit model.
from sklearn.metrics	#generating the report for

import classification_report	the tested set of the dataset for getting the precision & recall
from sklearn.model_selection import train_test_split	#used for developing a test and train selection model for the given dataset
from sklearn.metrics import confusion_matrix	#for giving the direction to print the evaluated normalized or non normalized confusion matrix
from sklearn.metrics import accuracy_score	#gives the accuracy score count printed on screen by evaluating the y_test and y_train from the library matrix

All the above mentioned libraries are exclusively used along with the algorithms mentioned in the part III for generating the results in the part V in the python interpreter.

C. Performance Measures

Machine learning involves teaching systems to teach themselves to solve problems and in order to evaluate how well a computer teaches itself the performance measures is used. Depending upon the type of problem an appropriate measure is used. For a classification problem the performance measures are:

1. Simple Accuracy.
2. Precision.
3. Recall.
4. F-measure.

The graphical methods are used for determining the performance of classification problems such as:

1. Receiving operating characteristic curve.

Also in the case of regression the performance measures are:

1. Sum of squares error.
2. Root mean square error.
3. Mean absolute error.

Classification involves determining the category of the sample while regression involves predicting the value of the sample. For this research work the classification is taken as the base and the measures involved with it are used. The classification measures used in this research work involves the use of essential entities in the corpus which are taken up as tags for evaluating the word to the respective part of speech tag like:

- per ⟶ person
- org ⟶ organization
- geo ⟶ geographical entity
- gpe ⟶ geopolitical entity
- tim ⟶ time indicator
- eve ⟶ event
- art ⟶ artefact
- nat ⟶ natural phenomenon

The above abbreviation criteria is used with inside-outside beginning (IOB) tagging for tagging tokens with:

- I ⟶ Prefix before tag represents that the given tag is inside a chunk
- B ⟶ Prefix before tag represents that the given tag is at the beginning of a chunk
- O ⟶ Tag indicates that a token belongs to no chunk that is it is outside of the chunk

Based on the above criteria in aggregate a confusion matrix is developed for evaluating the classification measures as:

It is required to construct a system that determines if a chunked sentence with a word corresponding to a POS is either of Tag categorised as above or not, this is a classification problem where words and POS are classified into two pre-defined categories either being a tag corresponding to the POS or not. [21]

Table1. Confusion Matrix explanation table

	P` (predicted)	**n` (predicted)**
P (actual)	True positive (TP)	False negative (FN)
n (actual)	False positive (FP)	True negative (TN)

In the table P represents that the tag is actually for the POS, n represents that the tag is actually not the POS.

- P` ⟶ represents that the system has predicted that the tag is the POS.
- n` ⟶ represents that the system has predicted that the tag is not the POS.

From these two columns there are four possible outcomes which are shown in the table2 below.

The resulting confusion matrix [22] plots using the logic from the table 1 & 2 for the algorithms in part III are given in part V using the matplotlib library of python. [21]

Table2. Outcomes using the P` and n`

True positive	For example predict a tag I-per as I-per	This implies that a system is correct
False positive	For example predict another tag as I-per	This implies that the system is not correct
False negative	For example predict a tag I-per as non I-per	
True negative	For example predict a non I-per as non I-per	This implies that the system is correct

Based on table 1 & 2 the classification measures are:

1. Precision:
   Of the tags classified as X, how many are actually X. (here X is the corresponding tag entry for the POS like I-per or else.)

Number of tags classified X = TP + FP.
Number of tags actually X (when classified X)= TP.
Therefore, precision = $\frac{TP}{TP + FP}$

2. Recall:
Of the tags that are actually X, how many are classified as X.
Number of tags actually X = TP + FN.
Number of tags classified X (when actually X)= TP.
Therefore, recall = $\frac{TP}{TP+FN}$ [23]

3. Accuracy:
Accuracy = $\frac{\text{Number of tags predicted correctly}}{\text{Total number of tags}}$
Therefore, accuracy = $\frac{TP+TN}{TP+TN+FP+FN}$

4. F-beta measure:
Considering the output from two distinct systems;

System 1	System 2
- precision : x%	- precision : a%
- recall: y%	- recall : b%

In order to know which of these two is actually performing better the F-beta measure can be computed as:

$$F_\beta = \frac{1}{\beta \times \frac{1}{Precision} + (1 - \beta) \times \frac{1}{Recall}}$$

F_β is the rated harmonic mean of precision and recall where β is the factor that determines the importance of precision over recall. [24]

Greater β, greater importance to precision. This value is set depending upon the type of classification problem. In certain problems a higher precision is required and this is the case while determining if a patient has cancer or not and hence higher β value. In the case of $\beta = 0.5$ this classification measure is called as F-Measure.
The function used to evaluate the classification measures explained above is defined in the python interpreter as:

```
def next_get(self):
    try:
        s = self.data[self.data["Sentence #"] == "Sentence: {}".format(self.n_sent)]
        self.n_sent += 1
        return s["Word"].values.tolist(), s["POS"].values.tolist(), s["Tag"].values.tolist()
    except:
        self.empty = True
        return None, None, None
```

V. COMPARISION & ANALYSIS

The Confusion matrix plots for the algorithms mentioned in part III except for CRF (as for CRF the eli5 library for seeing the contents is used) keeping the non-normalized and normalized concept into consideration are plotted using the matplotlib library and using the svm from the sklearn library of the python as below. [25] The confusion matrix without normalization plots the amount of the difference between the actual and the achieved precision over the number of the tags the algorithm made the search for whereas the confusion matrix with normalization does the plotting for fraction between predict to actual.

A. *Simple memory tagging algorithm*

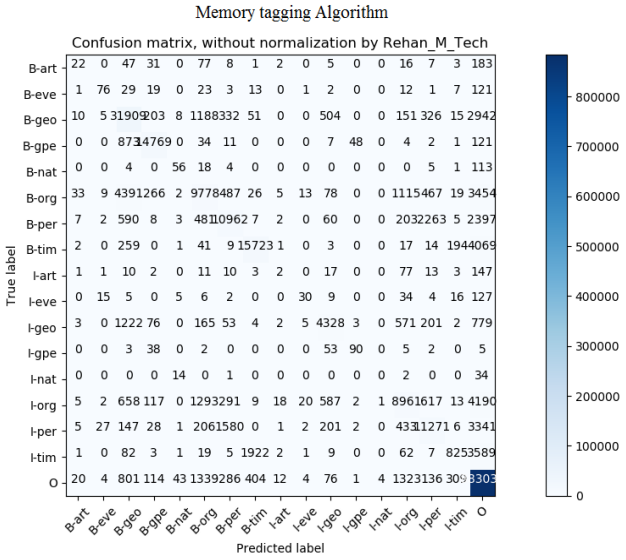

Figure2. CM without normalization for simple memory tagging algorithm

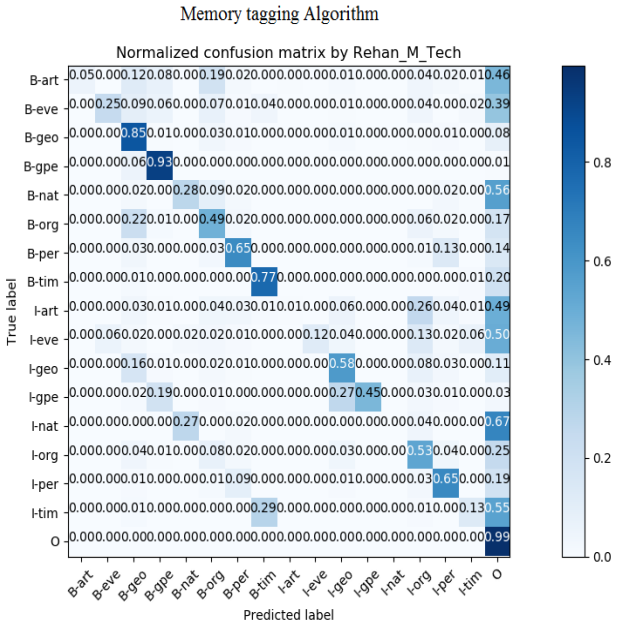

Figure3. CM with normalization for simple memory tagging algorithm

B. Perceptron

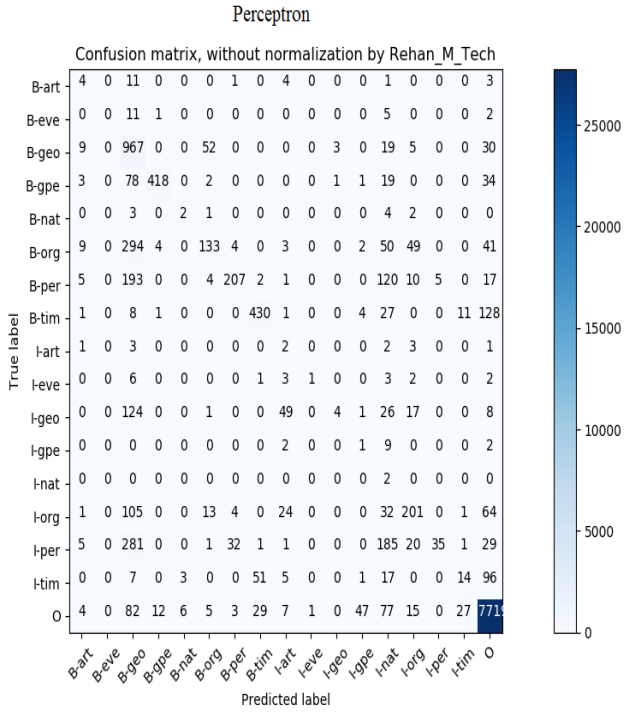

Figure4. Perceptron confusion matrix without normalization

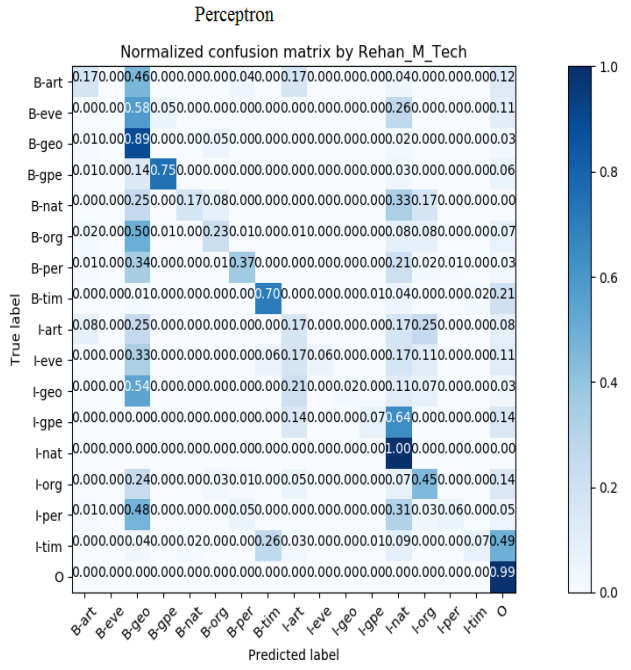

Figure5.Perceptron confusion matrix with normalization

C. Passive Aggressive Classifier

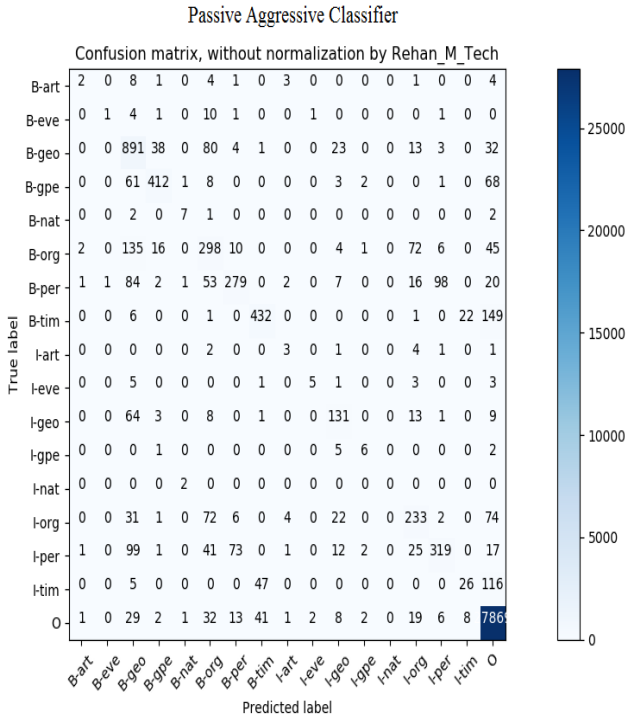

Figure6. Passive Aggressive Classifier confusion matrix without normalization

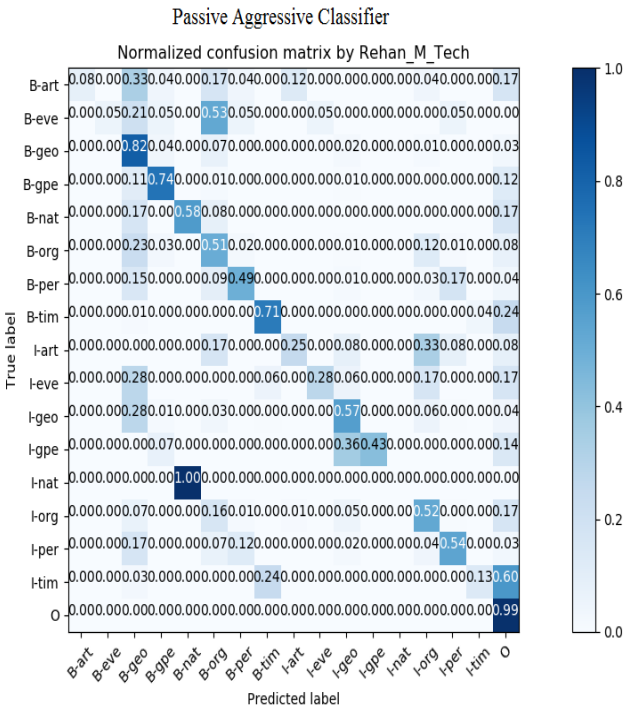

Figure7.Passive Aggressive Classifier confusion matrix with normalization

D. Stochastic Gradient Descent Classifier

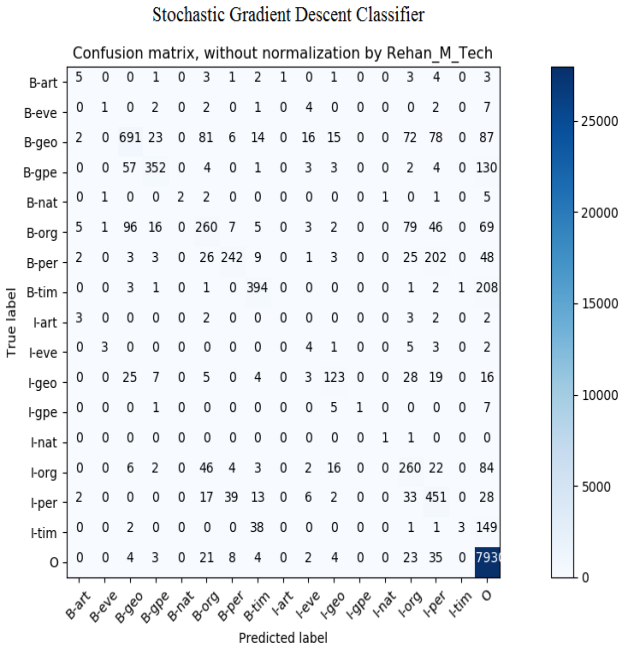

Figure8. Stochastic Gradient Descent Classifier confusion matrix without normalization

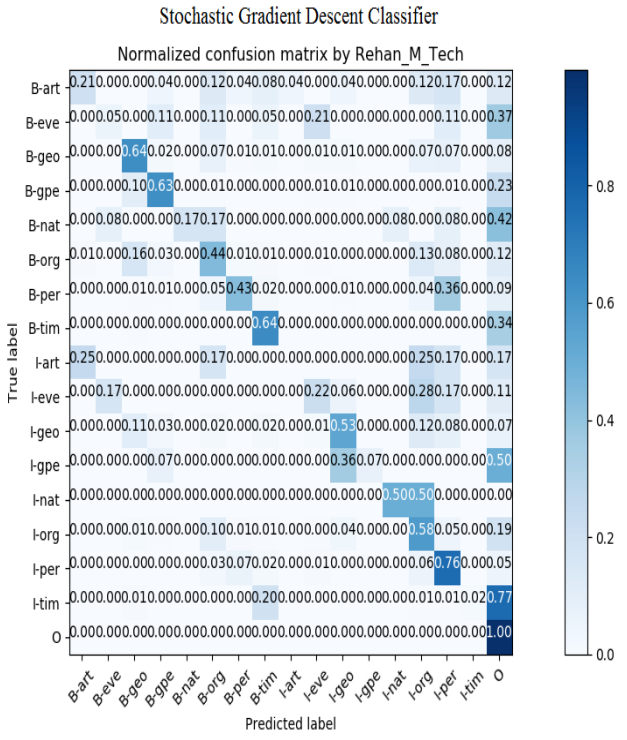

Figure8. Stochastic Gradient Descent Classifier confusion matrix with normalization

E. Naïve Bayes Classifier

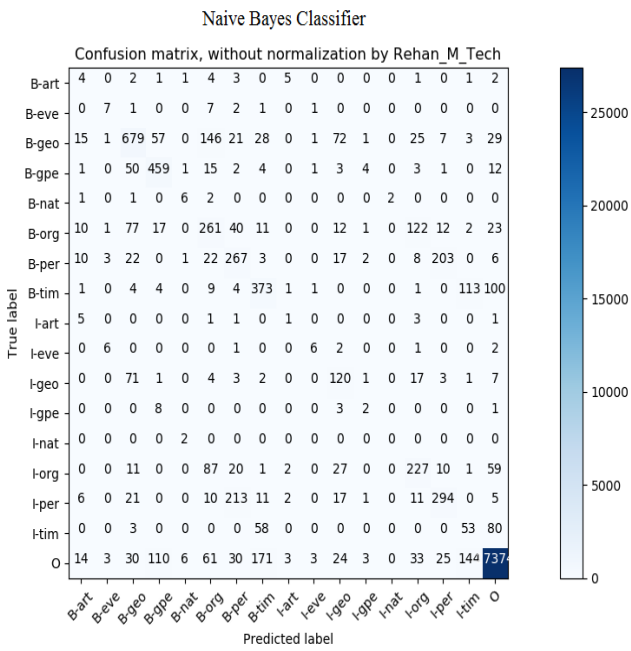

Figure10. Naïve Bayes Classifier confusion matrix without normalization

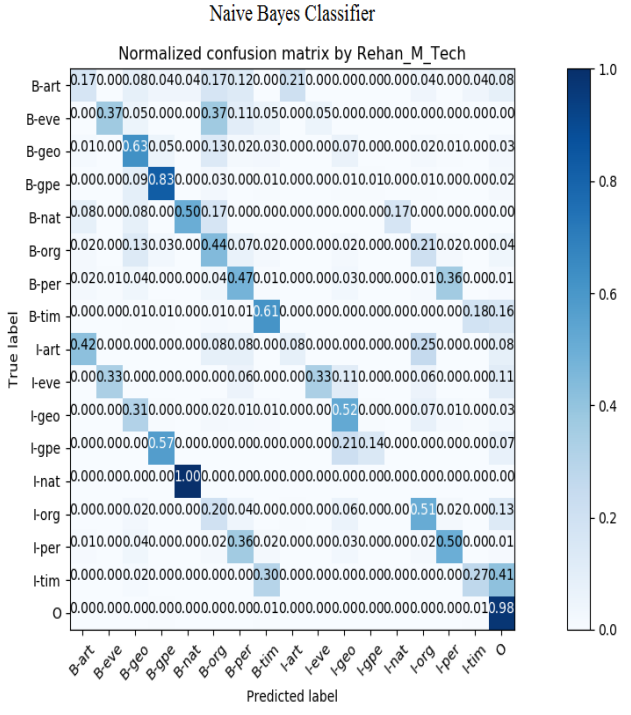

Figure10. Naïve Bayes Classifier confusion matrix with normalization

F. *Random Forest Classifier*

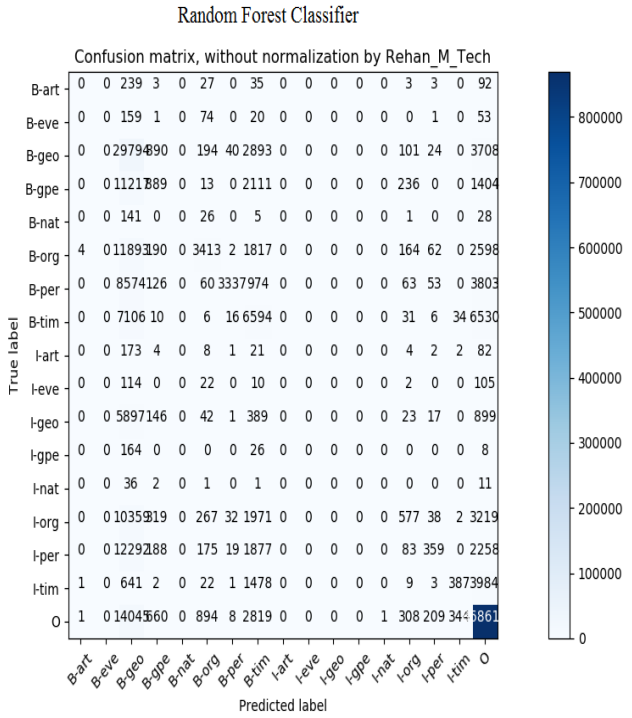

Figure11. Random Forest Classifier confusion matrix without normalization

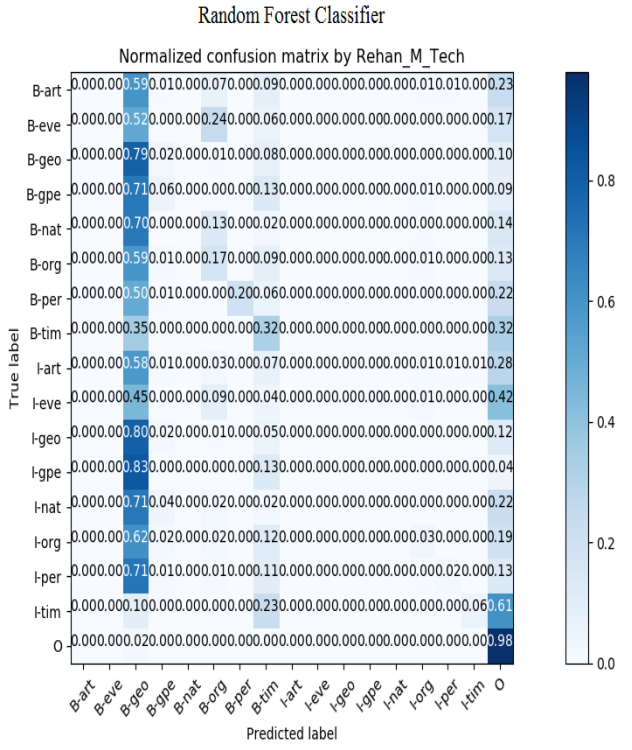

Figure13. Random Forest Classifier confusion matrix with normalization

G. *Random Forest Classifier with Pipeline*

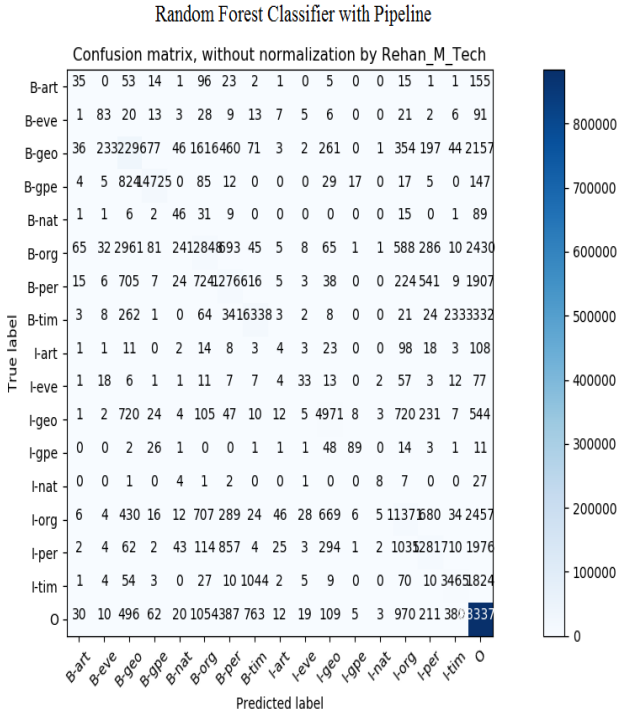

Figure14. Random Forest Classifier with pipeline confusion matrix without normalization

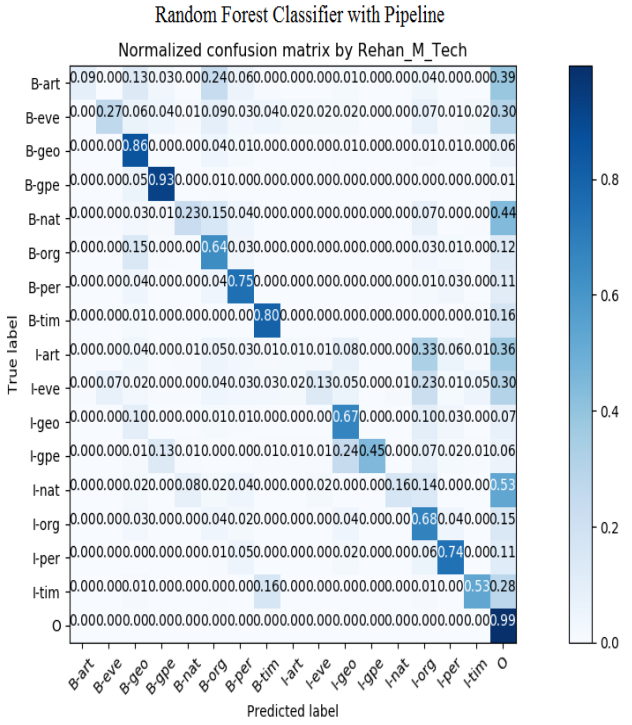

Figure15. Random Forest Classifier with pipeline confusion matrix with normalization

H. Condotional Random Fields

CRF eli5 output for Rehan_M_Tech/ner_dataset

From \ To	O	B-art	I-art	B-eve	I-eve	B-geo	I-geo	B-gpe	I-gpe	B-nat	I-nat	B-org	I-org	B-per	I-per	B-tim	I-tim
O	3.465	0.477	-2.326	0.973	-2.034	0.919	-4.235	0.506	-1.857	0.049	-1.256	0.794	-4.544	2.058	-3.123	1.417	-4.153
B-art	-0.876	-0.023	4.951	-0.003	-0.101	-0.373	-0.232	-0.373	-0.251	-0.008	-0.08	0.606	-0.601	-0.816	-0.784	-0.669	-0.324
I-art	-0.986	-0.279	4.665	-0.014	-0.086	0.336	-0.262	-0.272	-0.089	-0.008	-0.066	-0.44	-0.52	-0.747	-0.563	0.093	-0.399
B-eve	-0.533	-0.006	-0.077	-0.022	4.847	-0.234	-0.219	-0.328	-0.177	0.0	-0.04	-0.479	-0.504	-0.844	-0.409	-0.656	-0.515
I-eve	-0.333	0.0	-0.034	-0.653	3.968	-0.257	-0.193	-0.105	-0.059	-0.01	-0.009	-0.233	-0.272	-0.351	-0.387	-0.384	-0.177
B-geo	0.216	1.413	-1.024	-0.136	-0.695	-1.541	6.008	1.1	-1.881	-0.05	-0.502	-1.03	-1.947	-0.966	-1.813	1.688	-1.373
I-geo	-0.034	-0.048	-0.417	-0.029	-0.256	-1.011	5.296	-0.468	-0.719	-0.009	-0.147	-0.786	-1.018	-0.791	-0.642	1.238	-0.928
B-gpe	0.62	-0.255	-0.858	-0.278	-0.661	-0.184	-2.152	-3.169	3.92	-0.049	-0.296	0.951	-1.848	0.572	-1.357	-0.347	-0.987
I-gpe	-0.656	-0.163	-0.082	-0.01	-0.031	-0.007	-0.61	-0.624	3.97	0.0	-0.024	-0.377	-0.622	-0.619	-0.441	-0.684	-0.247
B-nat	-0.405	-0.001	-0.055	0.0	-0.042	-0.254	-0.109	-0.182	-0.068	-0.005	3.208	-0.255	-0.334	-0.55	-0.394	-0.231	-0.078
I-nat	-0.835	-0.002	-0.037	0.0	-0.007	-0.18	-0.053	-0.093	-0.026	-0.066	1.92	-0.133	-0.227	-0.364	-0.231	-0.182	-0.04
B-org	0.046	2.002	-1.136	-0.195	-0.816	-0.611	-1.839	-0.26	-1.572	-0.129	-0.703	-2.258	4.456	-0.771	-2.332	-0.652	-1.306
I-org	0.042	-0.319	-0.961	-0.174	-0.68	-1.657	-1.318	-0.708	-0.912	-0.434	-0.591	-2.177	4.321	-0.133	-2.456	0.119	-1.327
B-per	0.016	-0.302	-0.773	-0.174	-0.758	0.028	-1.0	0.617	-1.042	-0.095	-0.668	0.918	-1.698	-4.279	4.712	-0.386	-0.846
I-per	-0.223	-0.169	-0.683	-0.278	-0.747	-1.268	-1.189	-0.71	-0.974	-0.078	-0.593	-1.132	-1.89	-3.095	4.04	0.177	-1.16
B-tim	0.311	-0.451	-0.4	-0.059	-0.557	-0.759	-0.991	-1.165	-0.447	0.611	-0.252	-0.629	-1.229	-1.309	-0.897	-2.448	4.575
I-tim	0.145	-0.144	-0.142	-0.356	-0.15	0.62	-0.291	0.037	-0.067	-0.064	-0.017	-0.812	-0.74	-0.006	-0.224	-1.571	4.789

Figure16. eli5 library output for the CRF.

I. Comparision plot for Precision, Recall, F-score, Accuracy & Response Time

The python line commands for dataset with csv extension in this research work were built in notepad and saved in the same directory where the dataset was kept as on running the python file saved with py extension it searches for the file with the name specified under the panda and read_csv library for further processing and making predictions out of the designed commands for the classifier in use. The output for python commands made in notepad and then saved in py extension for the GMB dataset by keeping in mind the annotation set and the entity tags defined for the corpus for CRF (Conditional Random Fields) classifier/technique is shown in figure17. The comparison graphs for all the classifiers/techniques are shown in figures 18 to 23 by comparing classification measures output obtained in a similar way like figure17 for all other techniques.

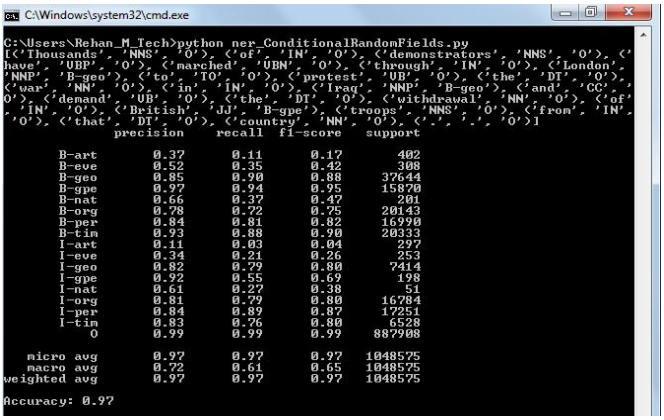

Figure17. Command prompt output for CRF

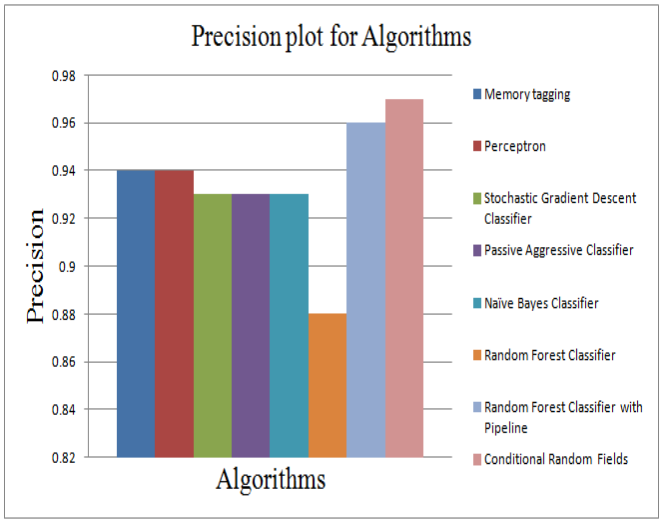

Figure18. Precision plot

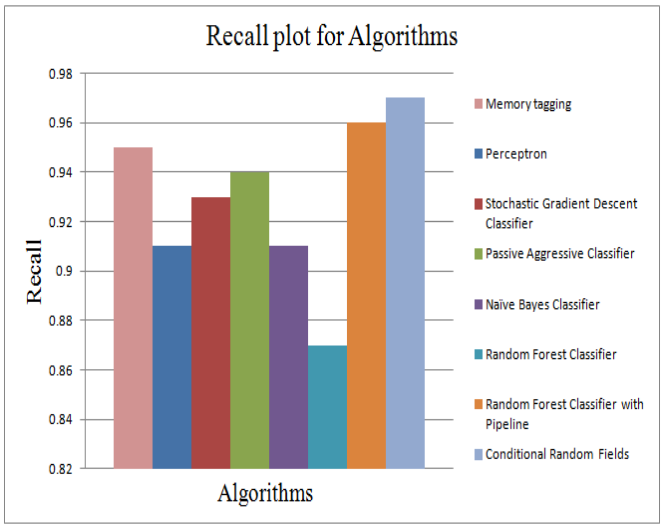

Figure19. Recall plot

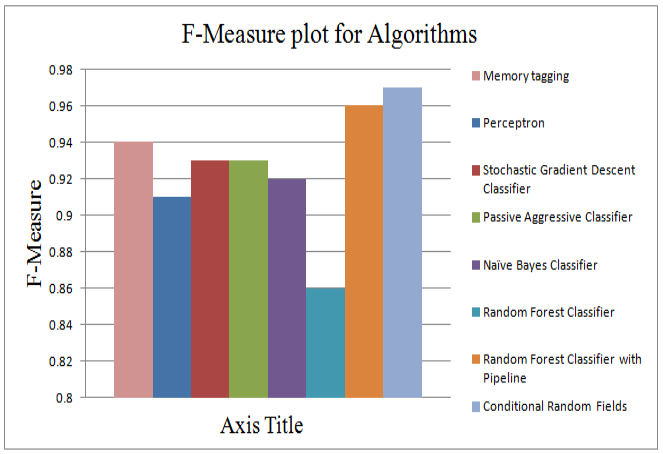

Figure20. F-Measure plot

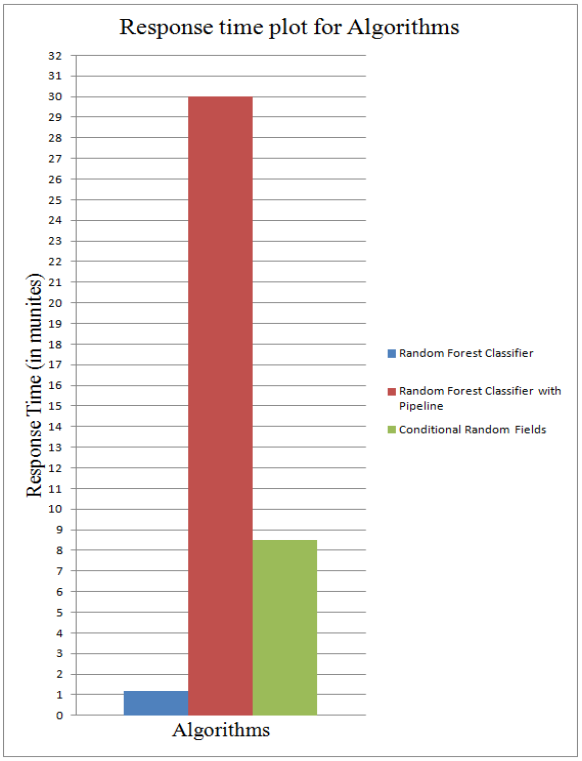

Figure21. Response time (in minutes) plot

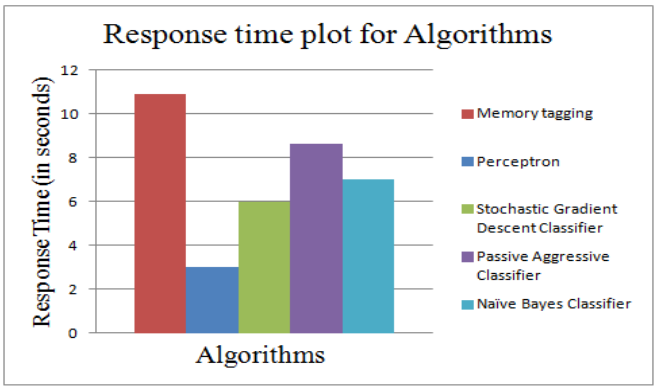

Figure22. Response time (in seconds) plot

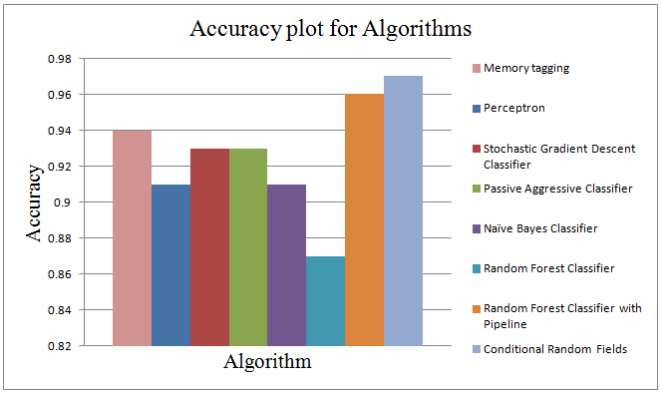

Figure23. Accuracy plot

Analysis of Results by classifiers using GMB dataset

1. Precision in the case of the conditional random field is the highest in comparison to any other technique.
2. Recall for the conditional random field is the highest.
3. F-Measure for the CRF also turns out to be best.
4. The accuracy shown by the conditional random field tends to be the highest amongst all.
5. Response time of the perceptron is unmatched and hence the best in comparison to any other technique used.

Analysis of Results by Stochastic HeHiCl-DFID using clinical dataset

Precision = 0.35; recall = 0.43; F-score = 0.32 & Accuracy = 0.43 for clinical dataset downloaded and annotated using GATE. The algorithm in comparison to conditional random field (CRF) makes no stand but the response time of 1.8 seconds that puts to shade all other techniques so far used as it uses the concept of the perceptron along with the random walk logic of CRF.

VI. Conclusion & Future Scope

In this paper a comparative analysis of the classifiers/techniques is made on the basis of the classification measures. The training & testing for classifiers is made by using the predefined GMB dataset and for Stochastic HeHiCl-DFID using dataset from the ClinicalTrials.gov for cancer in and around Shimla. The development of Stochastic HeHiCl-DFID using ontology sets and utilization of the concept of conditional random field can make a high facilitating supervised technique as being the future scope. The concept using the tensorflow i.e., the unsupervised form of machine learning that can make a fully automated information extraction system can make a major breakthrough as a good amount of accuracy and precision can be achieved using the concept of cascaded classifiers giving the efficiency of self and where limited utilizing other, which can ultimately help in making the clinical trial of the patients undergoing certain treatment fast and less cumbersome.

References

[1] E. H. Huang, R. Socher, C. D. Manning, and A. Y. Ng. "Improving Word Representations via Global Context and Multiple Word Prototypes." In ACL, 2012.

[2] Charles Sutton and Andrew McCallum, "An Introduction to Conditional Random Fields", Foundations and Trends in Machine Learning, Vol. 4, No.4, 267-373, 2012.

[3] Paul Anderson, Aspen Olmsted, Gayathri Parthasarathy, "NLP Pipeline for Temporal Information Extraction & Classification from free text Eligibility Criteria", International Conference on Information Society, IEEE, 2016.

[4] Dekai Wu, Grace Ngai, Marine Carpuat, Jeppe Larsen, and Yongsheng Yang. "Boosting for named entity recognition." In Dan

Roth and Antal van den Bosch, editors, Proc. 6th Conf. on Computational Natural Language Learning (CoNLL), 2002.
[5] Yefeng Wang and Jon Patrick. "Cascading classifiers for named entity recognition in clinical notes."In Proc. Workshop on Biomedical Information Extraction (WBIE), pages 42-49, 2009.
[6] Rehan Khan and A.J. Singh, "NLP: A Comparative Study with Algorithmic Approach for Information Extraction", International Journal of Emerging Technologies and Innovative Research, Vol.5, Issue 9, page no.970-979, September-2018.
[7] D. Koller and N. Friedman, "Probabilistic Graphical Models: Principles and Techniques", MIT Press, 2009.
[8] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. "Natural Language Processing (Almost) from Scratch." Journal of Machine Learning Research, 12:2493,2537, 2011.
[9] C. Sutton and A. McCallum, "Piecewise training for structured prediction," Machine Learning, vol. 77, no. 2–3, no. 2–3, pp. 165–194, 2009.
[10] J. Shotton, J. Winn, C. Rother, and A. Criminisi, "Textonboost: Joint appearance, shape and context modeling for mulit-class object recognition and segmentation," in European Conference on Computer Vision (ECCV), 2006.
[11] C. Sutton, K. Rohanimanesh, and A. McCallum, "Dynamic conditional random fields: Factorized probabilistic models for labeling and segmenting sequence data," in International Conference on Machine Learning (ICML), 2004.
[12] Yoav Freund and Robert E. Schapire. "Large margin classication using the perceptron algorithm."Machine Learning, 37(3):277 to 296, 1999.
[13] F. C. Peng and A. McCallum, "Accurate information extraction from research papers using conditional random fields," in HLT-NAACL 2004: Main Proceedings, pp. 329–336, Association for Computational Linguistics, Boston, Mass, USA, 2004.
[14] A. Torralba, K. P. Murphy, and W. T. Freeman, "Contextual models for object detection using boosted random fields," in Advances in Neural Information Processing Systems, vol. 17, pp. 1401–1408, 2005.
[15] Crammer K., Dekel O., Keshet J., Shalev-Shwartz S., Singer Y., "Online Passive-Aggressive Algorithms", Journal of Machine Learning Research 7 (2006) 551–585.
[16] V. Vineet, J. Warrell, P. Sturgess, and P. H. S. Torr, "Improved initialisation and Gaussian mixture pairwise terms for dense random fields with mean-field inference," in Proceedings of the 23rd British Machine Vision Conference (BMVC '12), Surrey, UK, September 2012.
[17] S. Kumar and M. Hebert, "Discriminative random fields," International Journal of Computer Vision, vol. 68, no. 2, pp. 179–201, 2006.
[18] N. Piatkowski and K. Morik, "Parallel loopy belief propagation in conditional random fields," in Proceedings of the KDML Workshop of the LWA, Magdeburg, Germany, 2011.
[19] M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz, "Building a large annotated corpus of English: The Penn Treebank," Computational Linguistics, vol. 19, no. 2, no. 2, pp. 313–330, 1993.
[20] GMB (Groningen Meaning Bank) corpus, http://gmb.let.rug.nl/.
[21] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 23, no. 11, pp. 1222–1239, 2001.
[22] Confusion Matrix, http:// scikit-learn.org/ stable/ modules/ generated/sklearn.metrics.confusion_matrix.html
[23] Scikit-learn, http://scikit-learn.org/ stable/ auto_examples/ model_selection/ plot_precision_recall.html # sphx-glr-auto-examples-model-selection-plot-precision-recall-py
[24] I. Sutskever and T. Tieleman, "On the convergence properties of contrastive divergence," in Conference on Artificial Intelligence and Statistics (AISTATS), 2010.
[25] Erik F. Tjong Kim Sang. "Introduction to the CoNLL-2002 shared task:Language-independent named entity recognition."In Dan Roth and Antal van den Bosch, editors, Proc. 6th Conf. on Computational Natural Language Learning (CoNLL), pages 155 to 158, 2002.

Authors Profile

Mr.Rehan Khan pursed Bachelor of Technology in Electronics & Communication Engineering from Himachal Pradesh University, Shimla, India in 2014. He worked in Pioneer Electricals for one year and thereafter switched to National Institute of Electronics & Information Technology as Faculty for one year. He is currently pursuing Master of Technology in Computer Science from Himachal Pradesh University, Shimla, India. His main research work focuses on Artificial Intelligence, Natural Language Processing and algorithms.

Dr. A.J. Singh is a Professor in Department of Computer Science in Himachal Pradesh University, Shimla, India. He has been in this department since 1992. He has obtained his Bachelor of Engineering degree in Computer Technology from National Institute of Technology (MANIT) Bhopal, Master of Science in Distributed Information Systems from University of East London (UK) under British Government ODASS Scholarship and Ph.D degree from Himachal Pradesh University, Shimla, India. He worked on deputation to Royal Government of Bhutan (Education Division) under Colombo Plan for three years. He has published more than 50 rsearch papers, supervised six Ph.D. students, 9 students doing Ph.D. under his supervision and guided many M.Tech Dissertations. His areas of interests are Distributed systems (Networks and DBMS), and ICT for Development.

