VISVESVARAYA NATIONAL INSTITUTE OF TECHNOLOGY, NAGPUR

ENGLISH PREMIER LEAGUE CLASSIFIER

BY:

SHUBHAM SHEKHAR SHAH

SHUBHAM VIJAY SAXENA

A THESIS SUBMITTED IN PARTIAL FULFILLMENT FOR THE
DEGREE OF BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING

UNDER THE GUIDANCE OF :

DR. UMESH A. DESHPANDE

DR.VINAY KULKARNI (TRDCC)

SAPAN HS(TRDCC)

DEEPALI KHOLKAR(TRDCC)

Declaration of Authorship

English Premier League Classifier is work carried out by us under the guidance of Dr. U.A. Deshpande,Department of Computer Science and Engineering, Visvesvaraya National Institute of Technology, Nagpur. This project has not been submitted to any other Institute or University for any degree or diploma.

Shubham Shah

Shubham Saxena

Certificate

This is to certify that " **English Premier League Classifier** " being submitted by Shubham Shah and Shubham Saxena in the partial fulfillment of the requirement for the completion of the course for the degree of Bachelor of Technology in Computer Science and Engineering, is a record of the students' own work carried out by them under my supervision and guidance. This work is comprehensive, complete and fit for evaluation.

Dr. O.G. Kakde
Head of Department

Dr. U.A Deshpande
Project Guide

Acknowledgement

We owe our gratitude to Dr. Vinay Kulkarni, TRDDC Pune, for granting us the opportunity to work on a research project offered by TRDDC in the first place. We wish to express our sincere thanks to our project guide Dr. U.A. Deshpande, Head of the Department, Department of Computer Science and Engineering, VNIT Nagpur, for motivating us to pursue the TRDDC offer and also for his invaluable guidance and continuous support till the end of this project. We are indebted to Sapan Hs and Deepali Kholkar, TRDDC Pune, for leading us throughout the beginning of the project phase without whose guidance, we would not even have had a start. We wish to pay our sincere thanks to the Department of Computer Science and Engineering, VNIT Nagpur, for granting us unrestricted access to the Department laboratory and for providing the requisite facilities needed to complete the project. Also, we are sincerely thankful to the entire faculty of Department of Computer Science and Engineering, VNIT Nagpur, for their support. Finally, we wish to thank all our colleagues, in this project group as well as others, for their constant encouragement and motivation to see through the project work.

Abstract

Football is undoubtedly the most watched sport in the world. Out of the many leagues that are played around the world we chose the English Premier League for our project.The Premier League as defined by Wikipedia is the top level of the English football league system. Contested by 20 clubs, it operates on a system of promotion and relegation with the English Football League (EFL).The Premier League is the most-watched sports league in the world, broadcast in 212 territories to 643 million homes and a potential TV audience of 4.7 billion people.

The aim of this project is from Match Reports of the English Premier League to be able to detect the Players , Managers and the Stadium involved.The project focuses on the scraping of the reports for the matches ;processing of the unstructured reports and then makes use of the CRF(Conditional Random Field) Model which has been tested to achieve an accuracy of over 80 percent in case of Players and 85 percent in case of the Managers.

Further by using these reports,extracting relationships between players/managers and team names by training the CRF model to do so by annotating the reports appropriately.

CONTENTS

3.Proposed technique

3.1 Creation of Data Set

3.1.1 Match Details

3.1.2 Match Report

3.1.3 Extracted Report

3.1.4 Scraping

3.1.5 Individual Report

3.2 Processing the Data Set

3.2.1 Use of the Tokenizer

3.2.2 Annotations

3.2.3 The Annotated Report.

4.Experimentation And Results

4.1 Training of the Model

4.1.1 CRF Classifier over Naive Bayes Classifier

4.1.2 Prerequisites of the CRF Classifier

4.1.3 Training of the CRF Model

4.1.4 Training

4.1.5 Training One Model from Multiple Files

4.2 Testing and Validation

4.2.1 Cross Validation.

4.2.2 K-fold Cross Validation.

4.2.3 Match Details.

CHAPTER 1:INTRODUCTION

English Premier League :

The premier league is contested by 20 Clubs .During the course of a season (from August to May) each club plays the others twice (a double round-robin system), once at their home stadium and once at that of their opponents', for a total of 38 games. Teams receive three points for a win and one point for a draw. No points are awarded for a loss. Teams are ranked by total points, then goal difference, and then goals scored. If still equal, teams are deemed to occupy the same position. If there is a tie for the championship, for relegation, or for qualification to other competitions, a play-off match at a neutral venue decides rank.

DataSet :

The reports of the matches in the English Premier League are to be scrapped from the ESPN Website. An example link of the report can be given as:

http://www.espn.in/football/report?gameId=480894

The total number of games played in a season in the English Premier League amount to 380 with each team playing 38 games.

Similar Existing Approach :

A very similar problem statement that we found on the Internet was to predict the line-ups for the next game based on statistics available for the previous games.However their approach differed from our problem statement because they used the dataset in the form of events.For example instead of using unstructured text their reports was in the event form of <EVENT-PLAYER1-TEAM-MATCH TIME>.For eg:- if a goal was scored by a player in the match in the 66^{th} then the event would look like:-

<GOAL>

Player_Name:Paul Pogba

Team:Manchester United

Time:66’

</GOAL>

So now the main focus of these reports were basically extracting such events and getting them stored in a tabular form.Now to predict the next match line up,the data would be fed to a RNN which would have a memory of the last few matches,say 5 matches.Thus the current form of the player would be taken into account.Also since all data is available in a database format,including player positions and form,the match prediction would follow from the result of the RNN.

A recurrent neural network (RNN) is a class of artificial neural network where connections between units form a directed graph along a sequence. This allows it to exhibit dynamic temporal behavior for a time sequence. Unlike feedforward neural networks, RNNs can use their internal state (memory) to process sequences of inputs. This makes them applicable to tasks such as unsegmented, connected handwriting recognition or speech recognition.

Overview of the Approach

Scraping of the ESPN Reports for the English Premier League matches to form the Dataset. The scraping involves extraction of only the textual content from the reports and hence excluding the videos,leaderboard,score timelines that are a part of the reports.

Now the obtained dataset that is in the form of Unstructured text needs to be converted to structured form for further processing.Annotating the entities PLAYER,STADIUM,MANAGER now has to be done for performing the Named Entity Recognition on the reports.

Once the Entities have been extracted a CRF Model is trained based on the Entities and the corresponding assigned labels. Example: Paul Pogba in the Structured Report form will have the label as PLAYER.

After training of the CRF Model now it can be tested on the new reports that can be in the textual form as well or the structured form and the Results are obtained in terms of the Precision and Recall values for each of the entities.

CHAPTER 2:

CONDITIONAL RANDOM FIELDS

Background

Conditional random fields (CRFs) are a class of/ statistical modeling method often applied in machine learning for structured prediction.

Whereas a discrete classifier predicts a label for a single sample without considering "neighboring" samples, a CRF can take context into account and then predicts sequences of labels for sequences of input samples.

So the CRF provided functionality of taking the context into account would provide better classifications of the entities based on the textual content of the reports.

Example: <Name> scored a goal for <Team>.

Then the CRF is very likely to classify the <Name> to be a PLAYER based on the textual content “scored a goal for” .

Thus the primary advantage of CRFs is the relaxation of the independence assumption. Independence assumption says that the variables don’t depend on each other and they don’t affect each other in any way.

Approaches for Classification

Naive Bayes Classifier

A directed graphical model (generative) which factorises the joint distribution $p(x1,...,xm,y)$ as a product of conditionals $p(xi|x1,...,xi-1,y)$.

Simplify the model by making the Naïve Bayes assumption that observations are independent, yielding the definition for a Naïve Bayes Classifier.

$$p(y|\mathbf{x}) \propto p(y,\mathbf{x}) \approx p(y)\prod_{i=1}^{m} p(x_i|y)$$

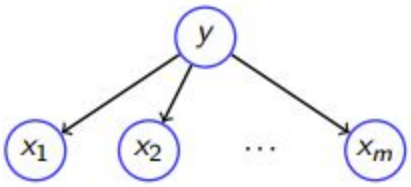


Figure: A Naive Bayes Classifier

Hidden Markov Models

HMMs are an extension of NB models to operate on label sequences.

Independence Assumption - each observation xi is assumed to only depend on the current class label yi.

It is however reasonable to assume there are dependencies between consecutive observations. Transition probabilities are used to capture this behaviour.

$$p(\mathbf{y},\mathbf{x}) = \prod_{i=0}^{n} p(y_i|y_{i-1})p(x_i|y_i)$$

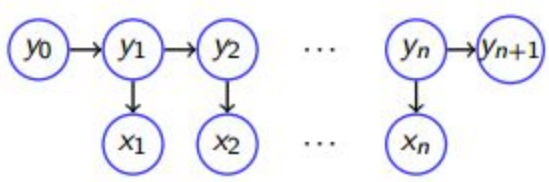


Figure: HMM Architecture

Maximum Entropy Models:

• An undirected graphical model (discriminative).

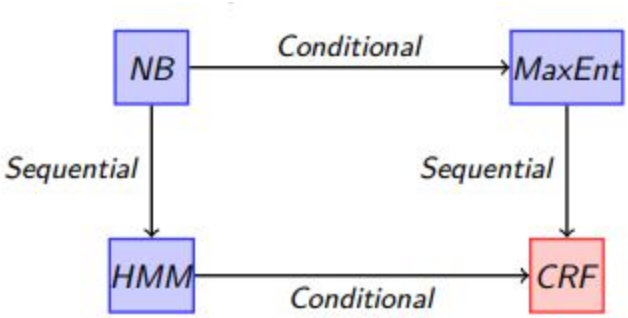
• No longer trained to maximise joint likelihood of data, but rather the conditional likelihood of the data.

• Factorises the joint distribution as a product of potential functions.

• Based on the principle of Maximum Entropy - model data so as to maximise the entropy given the inherent constraints of the training data

• A fundamental aspect of Maximum Entropy models is the representation of characteristics of the training data through a number of feature functions, fk (x,y).

Where do CRFs fit into the picture?



CRFs are discriminative sequential models which factorise the joint distribution into conditional potential functions.

CRFs address the independence assumption issue inherent to HMMs and the label bias problem inherent to MEMMs.

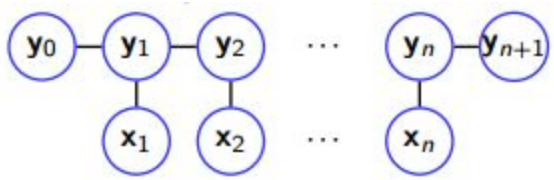


Figure: Basic Linear-Chain CRF architecture

Define the Linear-Chain CRF as a distribution of the form:

$$p_\theta(\mathbf{y}|\mathbf{x}) = \frac{1}{Z_\theta(\mathbf{x})} exp\left(\sum_k \lambda_k t_k(\mathbf{y},\mathbf{x}) + \sum_j \mu_j g_j(\mathbf{y},\mathbf{x})\right)$$

The feature functions tk and gj are assumed to be given and fixed, λk and μj are the associated Lagrangian multipliers.

The choice of an exponential family of distributions is natural within the Maximum Entropy framework employed for parameter estimation.

• Principal of parameter estimation for CRFs is based on that of Maximum Entropy models

Employ Conditional Maximum Likelihood training, i.e. maximise the conditional log likelihood of the training data (N training patterns, sequence length T, K feature functions)

$$p_\theta(\mathbf{y}|\mathbf{x}) = \frac{1}{Z_\theta(\mathbf{x})} exp\left(\sum_i \sum_k \lambda_k f_k(\mathbf{y}, \mathbf{x}, i)\right)$$

$$\mathcal{L}(\theta) = \sum_{p=1}^{N} log p(\mathbf{y}^{(p)}|\mathbf{x}^{(p)}) = \sum_{p=1}^{N}\sum_{i=1}^{T}\sum_{k=1}^{K} \lambda_k f_k(y_i^{(p)}, y_{i-1}^{(p)}, \mathbf{x}^{(p)}, i) - \sum_{p=1}^{N} log Z_\theta(\mathbf{x}^{(p)})$$

$$\frac{\partial \mathcal{L}(\theta)}{\partial \lambda_k} = \underbrace{\sum_{p=1}^{N}\sum_{i=1}^{T} f_k(y_i^{(p)}, y_{i-1}^{(p)}, \mathbf{x}^{(p)}, i)}_{\tilde{E}_{f_k}} - \underbrace{\sum_{p=1}^{N}\sum_{i=1}^{T}\sum_{y,y'} f_k(y, y', \mathbf{x}^{(p)}, i) p(y, y'|\mathbf{x}^{(p)})}_{E_{f_k}}$$

The derivative of the log likelihood function w.r.t fk is therefore equal to the difference E˜ fk −Efk .

• The empirical expectation E˜ fk is trivial to compute.

• The model expectation Efk is difficult to compute. The forward-backward algorithm is typically used to do so.

• Although this function is convex, no closed form solution exists. Iterative numerical techniques are required.

Feature Functions:

Binary feature functions may be extended to capture more interesting characteristics of underlying data.

• e.g. For POS tagging, fy,x = δ(x[0],upper(x[0]))δ(y,NP)

• Moment Constraints with binary feature functions acting on literal observations are natural for many applications (e.g. NLIP).

• It is also possible to construct sets of features for discrete valued observations, with delta functions centered at discrete points.

• More difficult to account for continuous valued features.

CRF Summary:

• CRFs estimate the distribution of a sequence of labels conditioned on an entire observation sequence.

• CRFs do not make conditional independence assumptions between elements of observation sequence (as with HMMs).

• CRFs are capable of performing at least as well as HMMs without any feature design effort.

• There are proven algorithms for parameter estimation in CRFs and HCRFs (LBFGS, RPROP, etc and Forwards-Backwards).

• Arbitrary combinations of input features can be considered - binary, discrete and continuous feature data streams can be used.

• HCRFs are a natural extension of the framework which makes it possible to use the CRF framework for more complex tasks.

Applications of CRF

1. 1.Part-of-Speech Tagging - (Lafferty et al. 2001)

    Improvement with HMM-like features from 5.7% to 5.6% classification error.

2. Named Entity Recognition - (McCallum and Li 2003)
3. Shallow Parsing - (Sha and Pereira 2003)
4. Object Recognition - (Quattoni et al. 2004)
5. Biomedical NER - (Settles, 2004)
6. Information Extraction - (Peng and McCallum 2004)
7. Phonetic Recognition - (Morris and Fosler-Lussier 2006)

    • Consistently showed 1-1.5% improvement over HMM baseline

8. Word alignment for Machine Translation - (Blunsom and Cohn 2006)

CRF Classifier over Naive Bayes Classifier

Naive Bayes classifiers are a family of simple "probabilistic classifiers "based on applying Bayes' theorem with strong (naive) independence assumptions between the features.

Thus Naive Bayes Classifier treats each of the feature(the tokenized words) to be independent when calculating the probabilities of classification.

CRF Classifier on the other hand considers the contextual relations between the features thus providing better classification over Naive Bayes Classifier.

Prerequisites of the CRF Classifier

The training data should be in tab-separated columns, with minimally the word tokens in one column and the class labels in another column.

The data file, the meaning of the columns, and what features to generate are to be provided via a properties file or can be provided using the command-line arguments as well.

Characteristics of CRF Classifier

The "map" property is used to define the meaning of the columns.

Columns are numbered from 0.

One column should be called "answer" and has the NER class, while another should be called "word" and has the tokens.So in our case,The word itself is the entity which needs to be tagged.The answer in this case is one of the three i.e:PLAYER,MANAGER or STADIUM.For all other words the answer is “O” which denotes others meaning they do not belong to a specific answer class.Basically the CRF takes in the input and the properties from the properties file and classifies the word as one of the above.

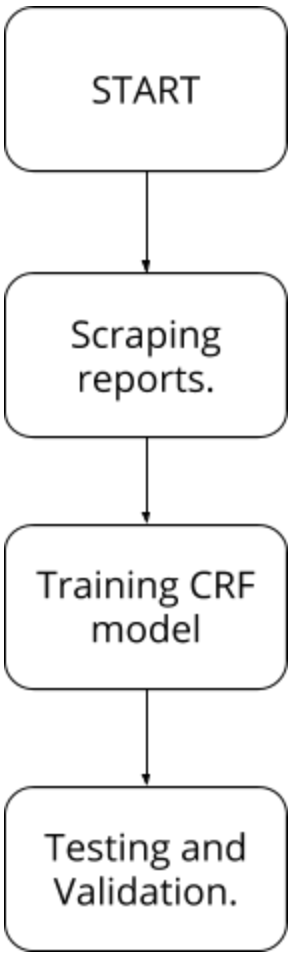
One more point to be taken into consideration is that the feature vector does not contain POS tags as the POS tags have no effect on the training data.This is due to the fact that the Stanford NER uses a labelling method very similar to the POS tagging and hence the training results produce almost exactly the same output.

CHAPTER 3:

PROPOSED TECHNIQUE

The following are the steps defining the workflow of the project:

1. Creation of Dataset
2. Processing the Dataset
3. Training and Testing the Model.



Creation of Dataset

The 380 match reports of the 2016-17 Season of the English Premier League are been scraped from the ESPN Site using Beautiful Soup.

Beautiful Soup is a Python package for parsing HTML and XML documents. It creates a parse tree for parsed pages that can be used to extract data from HTML, which is useful for web scraping. It is available for Python 2.6+ and Python 3.Web pages are usually stored in a tree format with the <HTML> tag as the root followed by the <HEAD> and <BODY> tags as the it's children which are further subdivided into tables,paragraphs and other such elements which are required to create a web-page.The web-page we scraped was also in such a tree format.What Beautiful Soup does is that it takes the URL of the webpage as it's input and parses the page to create the tree like format.Once the tree is parsed and generated,it is very easy to scrape the desired data using this tree like format.Beautiful Soup uses a top-down approach to create the tree.It will start parsing from the first tag till it finds the corresponding closing tag in it.So in this case the root becomes the <HTML> tag.Next it recursively keeps on adding tags to its children if the tag lies after the start and before the closing tag.When the Soup finds an ending tag it closes of the branch and moves to the parent thereby generating the tree.On encountering a new tag and moves down in the new branch.Once the bottom most branch is reached it will add the data as a leaf node to the tree.Hence the whole web document is stored as a tree and now is easily parsable.

So in our case we parse the report according to the tree generated by the Soup to get the report.To find out where the required data lies,one can easily inspect the source of the webpage to find the structure of the page and parse the Soup accordingly.

An example of a Match Report been scraped:

Report Details:

URL: http://www.espn.in/football/report?gameId=450650

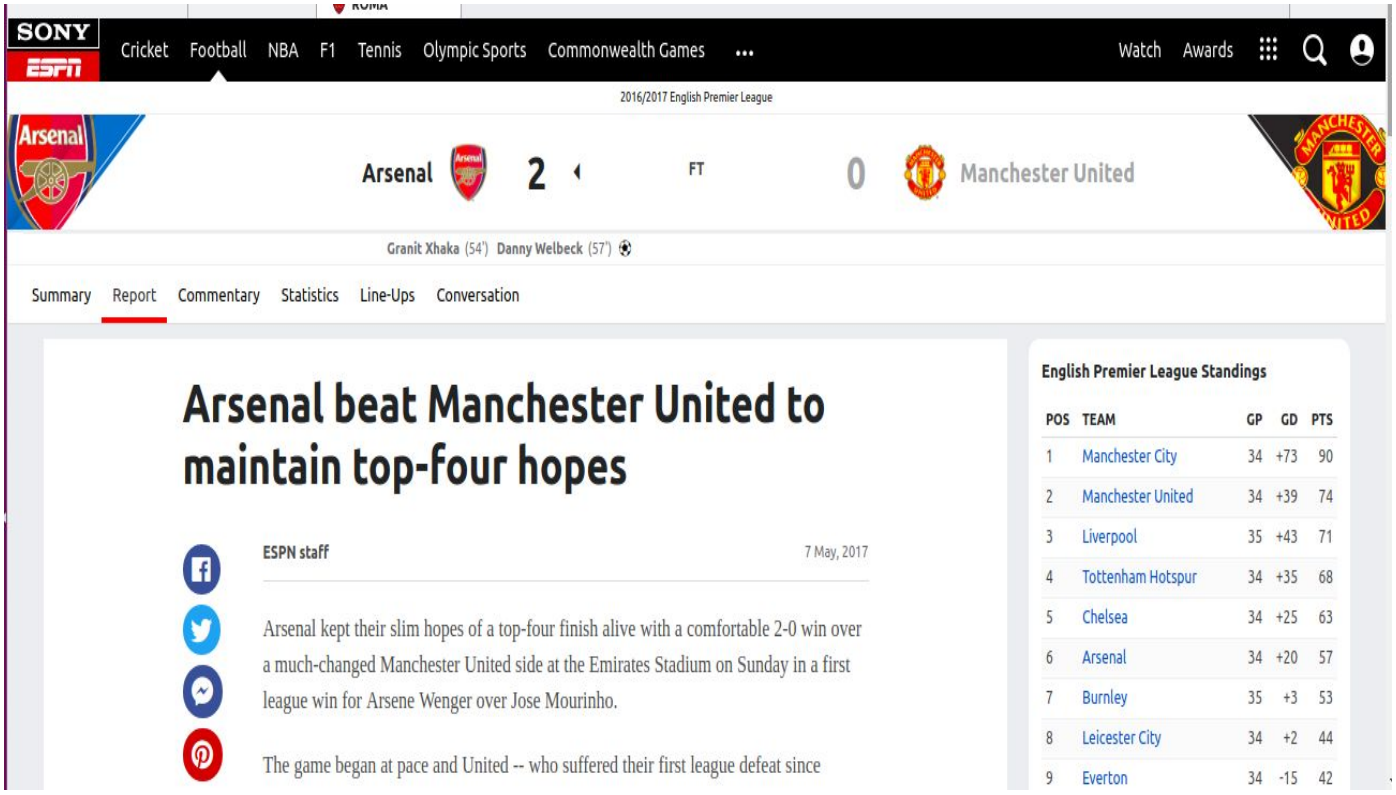
The above game featured a match between Arsenal and Manchester United,the two Premier League giants.The match was held at the home stadium of Arsenal which the The Emirates Stadium and was held on the 7th of May,2017.The report shows the status of the match as completed with the full time score being 2-0 in favor of the home team.In the reports the home team is always written first followed by the

away team which helps in identifying which is the Home team and which is the Away Team,Arsenal and Manchester United respectively in this case.

Match Details

HOME TEAM	ARSENAL
AWAY TEAM	MANCHESTER UNITED
SCORE	2-0
MATCH STATUS	FULL TIME
DATE	7th MAY,2017

Match Report



Extracted Report:

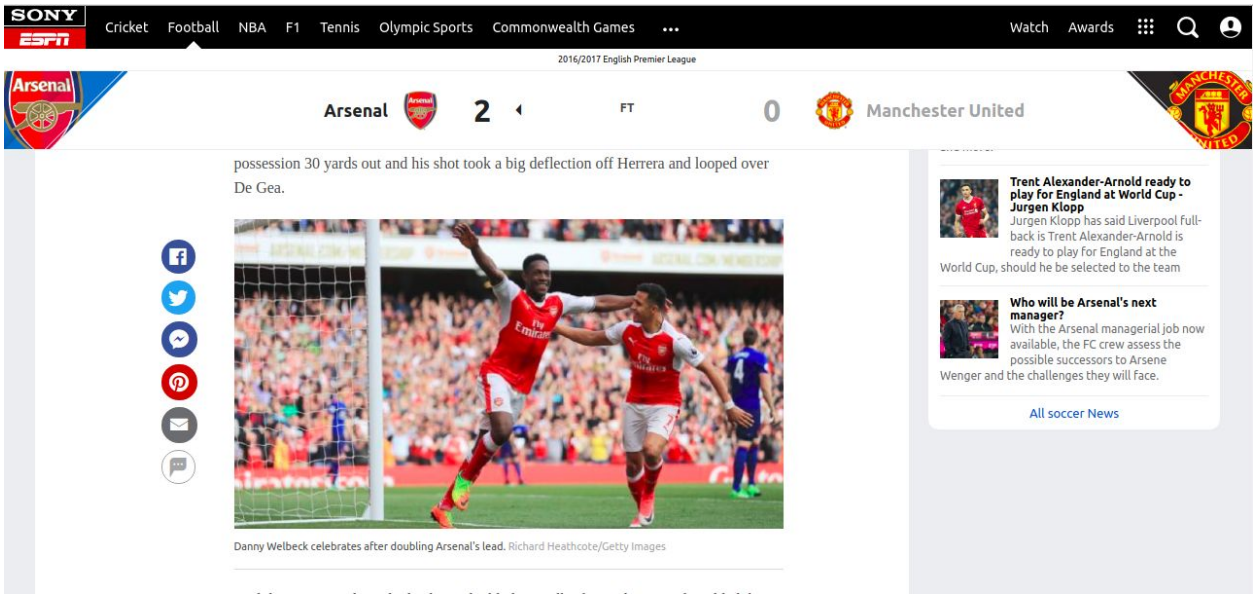
```
Arsenal VS Manchester United
May 7, 2017
Arsenal beat Manchester United to maintain top-four hopes

Arsenal kept their slim hopes of a top-four finish alive with a comfortable 2-0 win over a much-changed Manchester United side at the Emirates
Stadium on Sunday in a first league win for Arsene Wenger over Jose Mourinho.
The game began at pace and United -- who suffered their first league defeat since October -- were first to show when, only two minutes in, Juan
Mata crossed and Wayne Rooney drifted away from his marker but steered a header over.
Within three minutes, Kieran Gibbs got down the left for the hosts and Welbeck's attempt to control saw the ball fall to Sanchez, who got past
a defender and had his effort stopped by David De Gea.
Back came United, and Rooney slid a ball through to Anthony Martial, whose strike was turned behind by Petr Cech -- and before 10 minutes had
been played there was another opening as Aaron Ramsey shot low and De Gea made an outstanding save.
As the quarter of an hour mark approached, Danny Welbeck almost got through for Arsenal but was denied by Ander Herrera after he had drifted
away from Phil Jones and cut into the area.
But United were beginning to enjoy more of the possession and, with 21 minutes gone, a deep Mata free kick to the far post was headed on by
Henrikh Mkhitaryan only to elude his teammates.
Mata was involved in another opening, getting away down the right but crossing too close to Cech as Mkhitaryan waited unmarked in the middle.
Then Welbeck had an effort on the turn blocked by Jones six yards out after chaos from an Arsenal corner, and on the half-hour Chris Smalling
made a crucial intervention at the near post after Ramsey had got away and crossed low.
Arsenal kept pressing, and after some fine footwork by Alex Oxlade-Chamberlain had bought him space on the edge of the area his shot drew
another good stop from De Gea.
But United should have led on 32 minutes when Rob Holding's dangerous pass back into his own area was seized upon by Rooney as Laurent
Koscielny hesitated, only for Cech to block.
The second half started in sleepy fashion, with youngster Axel Tuanzebe called into its first piece of action as he fended off a raid from
Sanchez down the Arsenal left.
But after 54 minutes the home side broke through when Granit Xhaka picked up possession 30 yards out and his shot took a big deflection off
Herrera and looped over De Gea.
And three minutes later the lead was doubled as Welbeck struck against his old club, heading home a pinpoint Oxlade-Chamberlain cross.
Mourinho reacted to the double strike with a double change, taking off Mkhitaryan and replacing him with Jesse Lingard and, moments later,
introducing Marcus Rashford for Herrera.
Rooney forced Cech into action with a curling free kick from 25 yards after 65 minutes, but United were doing little to worry Arsenal as
Wenger's side sat comfortably on their lead.
Ramsey won the ball back deep in United territory as the Gunners looked to add a third, but when he laid it back to Sanchez the cross was
blocked.
Ozil crossed from the left but just ahead of the waiting Welbeck before Rooney shot wide as United laboured in vain for a way back into the
match.
Xhaka was replaced by Francis Coquelin for Arsenal and Rooney shot over again when he had both time and space on the edge of the box.
Oxlade-Chamberlain was replaced by Hector Bellerin and Olivier Giroud came on for Welbeck as the home side made a pair of changes, while Scott
McTominay made his senior United debut when he came on for Mata.
```

Scraping

Beautiful Soup(a python library) was made use of to extract only the textual content from the Reports. The report contained various other entities like Leaderboard,Press Conference Videos,Match Images,Social Media Advertising all of which were undesired.

The following image shows the same:



Individual Report

Each individual scraped report was stored as a textual file with the filename following the convention:

HTvsAT:Date_Year.txt

Where:

HT: Abbreviation of the Home Team

Example :Arsenal is abbreviated to ARS.

FT: Abbreviation of the Home Team

Example :Manchester United is abbreviated to MAN.

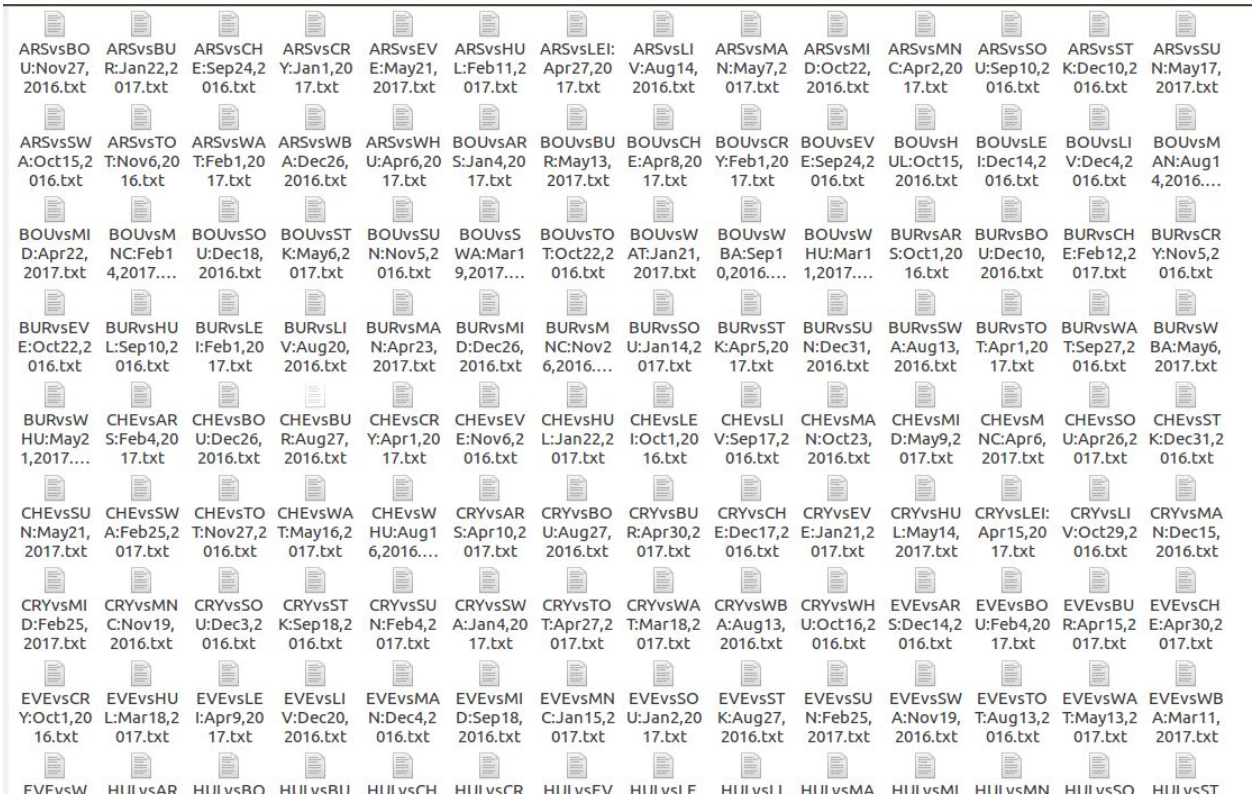
Date Format:MonthDay_Year

So the above match Report is stored as

ARSvsMAN:May7_2017.txt

Overall Reports:

All of the 380 Match Reports were extracted in the above mentioned approach.



Processing The Dataset:

Having each individual report processed the next steps involve:

TOKENIZATION:

Tokenization involves tokenizing i.e converting each word from the reports to tokens to be fed to the next steps.For this,the reports are selected individually and passed through the tokenizer.The tokenizer separates each word and stores them in a tsv format for further processing.

Stanford Named Entity Recogniser under the Stanford Natural Language Processing Group provides its tokenizer to convert to one token per line.

Use of the Tokenizer

```
java -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -loadClassifier
/home/shubham/FYProject/annotatedreports/ner-model.ser.gz -testFile
/home/shubham/FYProject/annotatedreports/MANvsHUL\:Feb2-2017.tsv >
/home/shubham/FYProject/annotatedreports/tested/MANHUL.tsv
```

Annotations

We then need to make training data where we label the entities Player , Manager and Stadium.

There are various annotation tools available, or you could do this by hand in a text editor. One way is to default to making everything an other and then to hand-label the real entities in a text editor.

Default Annotations:

The first step can be done with Perl command :

Example:

```
perl -ne 'chomp; print "$_\tO\n"'
/fyproject/tokenizedreports/ARSvsMAN:May7,2017.tok >
~/fyproject/annotatedreports/ARSvsMAN:May7,2017.tsv
```

Manual Annotations:

The entities :

- Player
- Manager
- Stadium

Were annotated..Of the total 380 reports,100 reports were chosen to represent the total data set and annotated to be further trained by the CRF classifier.Output of

the manually annotated reports and tokens can be seen with the following example of a match.

The Annotated Report

	A	B
13	Arsenal	O
14	edge	O
15	past	O
16	Bournemouth	O
17	Alexis	PLAYER
18	Sanchez	PLAYER
19	scored	O
20	twice	O
21	as	O
22	Arsenal	O
23	ended	O
24	their	O
25	usual	O
26	barren	O
27	November	O
28	run	O
29	with	O
30	victory	O
31	over	O
32	Bournemouth	O
33	at	O
34	the	O
35	Emirates	STADIUM
36	Stadium	STADIUM
37	on	O
38	Sunday	O
39	0	O
40	The	O
41	Chilean	O
42	struck	O

CHAPTER 4:EXPERIMENTATION AND RESULTS

Training of the CRF Model:

Properties File:

Stanford NER CRF allows all properties to be specified on the command line, but it is easier to use a properties file.

The Properties File for training the model is :

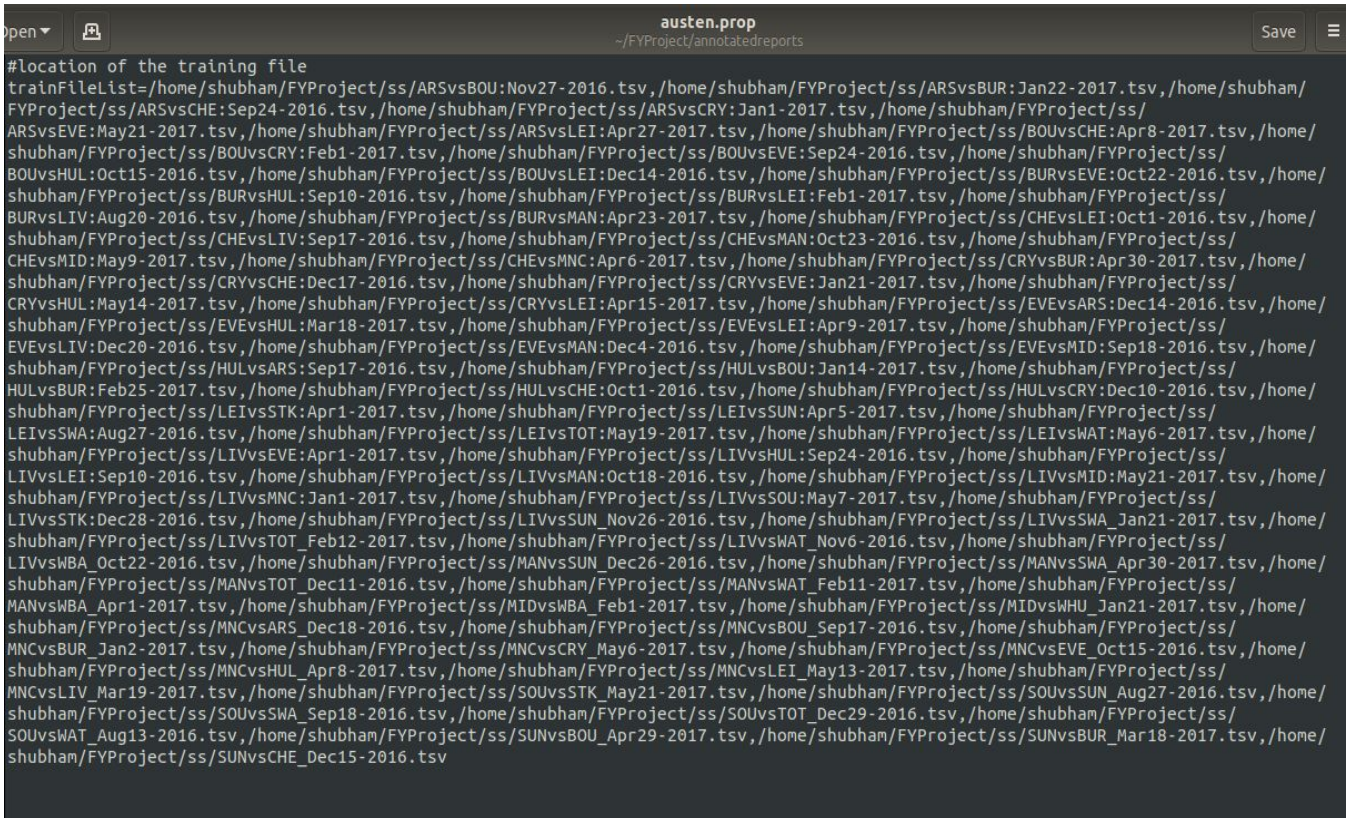
The first image shows the list of training files provided to the CRF properties file for the training.As you can see,100 reports have been provided as a part of the training data to the classifier.

The second image shows the list of properties that is used to train the reports stored in the tsv format.The following table explains the properties that have been used.

The list of properties can be seen from the Stanford documentation page.To train multiple files together the whole train files list is provided in the trainFileList Property which is a comma separated list of paths of the reports to be used for the training purposes.The trained model is stored to an NER whose path is provided in the serializeTo Property.When testing and generating output,the model is retrieved from this location.The model is stored in the tar format to save space.

PROPERTY	DESCRIPTION
useWord	Gives you feature for w
useNGrams	Make features from letter n-grams, i.e., substrings of the word
noMidNGrams	Do not include character n-gram features for n-grams that contain neither the beginning or end of the word
maxNGramLeng	If this number is positive, n-grams above this size will not be used in the model
usePrev	Gives you feature for (pw,c), and together with other options enables other previous features, such as (pt,c) [with useTags)
useNext	Gives you feature for (nw,c), and together with other options enables other next features, such as (nt,c) [with useTags)
usePrevSequences	Does not use any class combination features using previous classes if this is false.
maxLeft	The number of things to the left that have to be cached to run the Viterbi algorithm: the maximum context of class features used.
useTypeSeqs	Use basic zeroeth order word shape features.
useTypeSeqs2	Add additional first and second order word shape features
useTypeySequences	Some first order word shape patterns.

wordShape	Either "none" for no wordShape use, or the name of a word shape function recognized by WordShapeClassifier.lookupShaper (String)
useClassFeature	Include a feature for the class (as a class marginal). Puts a prior on the classes which is equivalent to how often the feature appeared in the training data.



```
#location where you would like to save (serialize to) your
#classifier; adding .gz at the end automatically gzips the file,
#making it faster and smaller
serializeTo = /home/shubham/FYProject/ss/ner-model.ser.gz

#structure of your training file; this tells the classifier
#that the word is in column 0 and the correct answer is in
#column 1
map = word=0,answer=1

#these are the features we'd like to train with

useClassFeature=true
useWord=true
useNGrams=true
#no ngrams will be included that do not contain either the
#beginning or end of the word
noMidNGrams=true
useDisjunctive=true
maxNGramLeng=6
usePrev=true
useNext=true
useSequences=true
usePrevSequences=true
maxLeft=1
#the next 4 deal with word shape features
useTypeSeqs=true
useTypeSeqs2=true
useTypeySequences=true
wordShape=chris2useLC
```

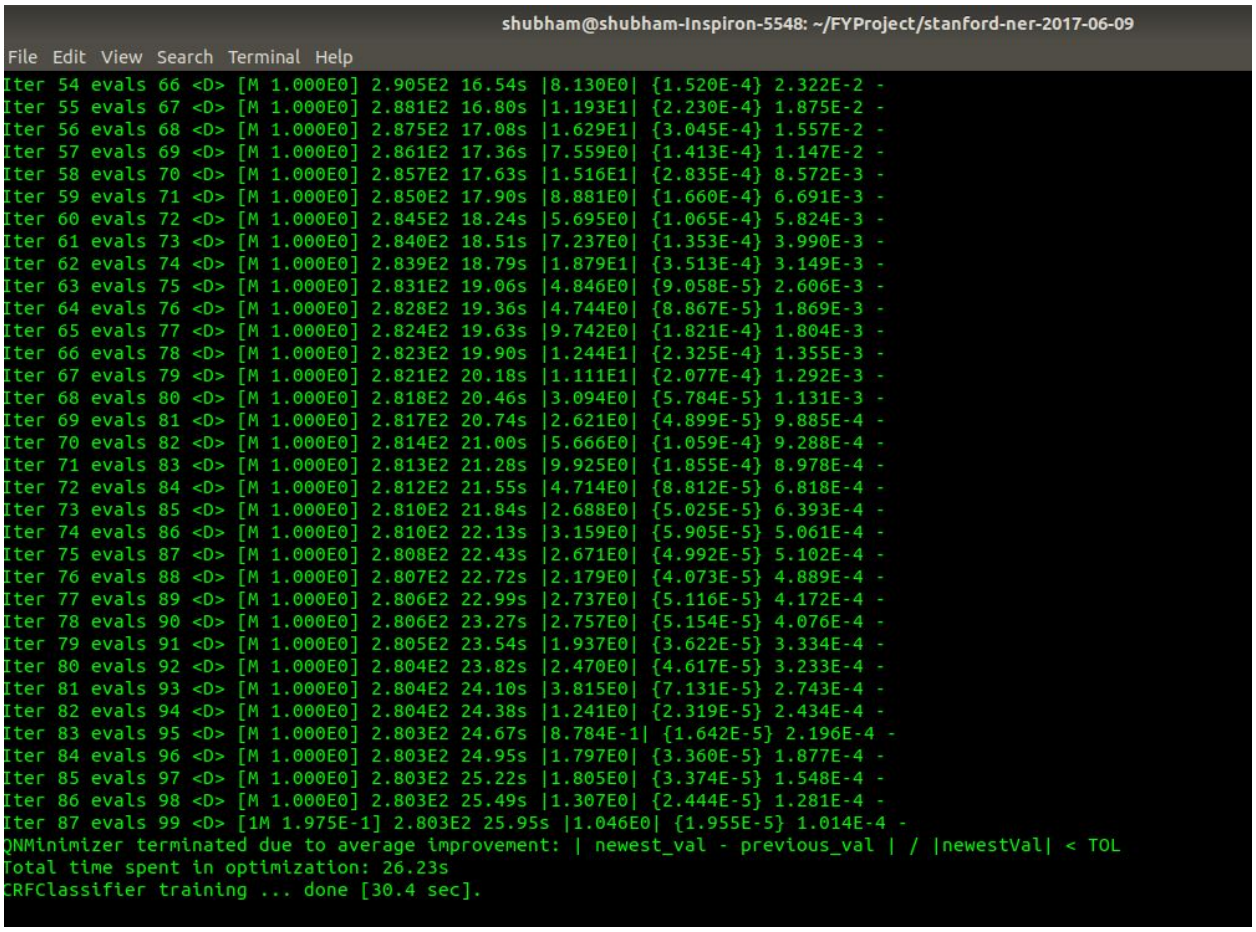
Training:

Once you have such a properties file, you can train a classifier with the command.The classifier continues the iterations until the error values reaches a below a certain threshold.The number of iterations used by the classifier are close to hundred.:

```
shubham@shubham-Inspiron-5548:~/FYProject/stanford-ner-2017-06-09$ java -cp stanford-ner.jar edu.stanford.nlp.ie.crf.CRFClassifier -prop /home/s
hubham/FYProject/annotatedreports/austen.prop
Invoked on Tue Apr 24 00:25:03 IST 2018 with arguments: -prop /home/shubham/FYProject/annotatedreports/austen.prop
usePrevSequences=true
useClassFeature=true
useTypeSeqs2=true
useSequences=true
wordShape=chris2useLC
useTypeySequences=true
useDisjunctive=true
trainFileList=/home/shubham/FYProject/ss/ARSvsBOU:Nov27-2016.tsv,/home/shubham/FYProject/ss/ARSvsBUR:Jan22-2017.tsv,/home/shubham/FYProject/ss/A
RSvsCHE:Sep24-2016.tsv,/home/shubham/FYProject/ss/ARSvsCRY:Jan1-2017.tsv,/home/shubham/FYProject/ss/ARSvsEVE:May21-2017.tsv,/home/shubham/FYProj
ect/ss/ARSvsLEI:Apr27-2017.tsv,/home/shubham/FYProject/ss/BOUvsCHE:Apr8-2017.tsv,/home/shubham/FYProject/ss/BOUvsCRY:Feb1-2017.tsv,/home/shubham
/FYProject/ss/BOUvsEVE:Sep24-2016.tsv,/home/shubham/FYProject/ss/BOUvsHUL:Oct15-2016.tsv,/home/shubham/FYProject/ss/BOUvsLEI:Dec14-2016.tsv,/hom
e/shubham/FYProject/ss/BURvsEVE:Oct22-2016.tsv,/home/shubham/FYProject/ss/BURvsHUL:Sep10-2016.tsv,/home/shubham/FYProject/ss/BURvsLEI:Feb1-2017.
tsv,/home/shubham/FYProject/ss/BURvsLIV:Aug20-2016.tsv,/home/shubham/FYProject/ss/BURvsMAN:Apr23-2017.tsv,/home/shubham/FYProject/ss/CHEvsLEI:Oc
t1-2016.tsv,/home/shubham/FYProject/ss/CHEvsLIV:Sep17-2016.tsv,/home/shubham/FYProject/ss/CHEvsMAN:Oct23-2016.tsv,/home/shubham/FYProject/ss/CHE
vsMID:May9-2017.tsv,/home/shubham/FYProject/ss/CHEvsMNC:Apr6-2017.tsv,/home/shubham/FYProject/ss/CRYvsBUR:Apr30-2017.tsv,/home/shubham/FYProject
/ss/CRYvsCHE:Dec17-2016.tsv,/home/shubham/FYProject/ss/CRYvsEVE:Jan21-2017.tsv,/home/shubham/FYProject/ss/CRYvsHUL:May14-2017.tsv,/home/shubham/
FYProject/ss/CRYvsLEI:Apr15-2017.tsv,/home/shubham/FYProject/ss/EVEvsARS:Dec14-2016.tsv,/home/shubham/FYProject/ss/EVEvsHUL:Mar18-2017.tsv,/home
/shubham/FYProject/ss/EVEvsLEI:Apr9-2017.tsv,/home/shubham/FYProject/ss/EVEvsLIV:Dec20-2016.tsv,/home/shubham/FYProject/ss/EVEvsMAN:Dec4-2016.ts
v,/home/shubham/FYProject/ss/EVEvsMID:Sep18-2016.tsv,/home/shubham/FYProject/ss/HULvsARS:Sep17-2016.tsv,/home/shubham/FYProject/ss/HULvsBOU:Jan1
4-2017.tsv,/home/shubham/FYProject/ss/HULvsBUR:Feb25-2017.tsv,/home/shubham/FYProject/ss/HULvsCHE:Oct1-2016.tsv,/home/shubham/FYProject/ss/HULvs
CRY:Dec10-2016.tsv,/home/shubham/FYProject/ss/LEIvsSTK:Apr1-2017.tsv,/home/shubham/FYProject/ss/LEIvsSUN:Apr5-2017.tsv,/home/shubham/FYProject/s
s/LEIvsSWA:Aug27-2016.tsv,/home/shubham/FYProject/ss/LEIvsTOT:May19-2017.tsv,/home/shubham/FYProject/ss/LEIvsWAT:May6-2017.tsv,/home/shubham/FYP
roject/ss/LIVvsEVE:Apr1-2017.tsv,/home/shubham/FYProject/ss/LIVvsHUL:Sep24-2016.tsv,/home/shubham/FYProject/ss/LIVvsLEI:Sep10-2016.tsv,/home/shu
bham/FYProject/ss/LIVvsMAN:Oct18-2016.tsv,/home/shubham/FYProject/ss/LIVvsMID:May21-2017.tsv,/home/shubham/FYProject/ss/LIVvsMNC:Jan1-2017.tsv,/
home/shubham/FYProject/ss/LIVvsSOU:May7-2017.tsv,/home/shubham/FYProject/ss/LIVvsSTK:Dec28-2016.tsv,/home/shubham/FYProject/ss/LIVvsSUN_Nov26-20
16.tsv,/home/shubham/FYProject/ss/LIVvsSWA_Jan21-2017.tsv,/home/shubham/FYProject/ss/LIVvsTOT_Feb12-2017.tsv,/home/shubham/FYProject/ss/LIVvsWAT
_Nov6-2016.tsv,/home/shubham/FYProject/ss/LIVvsWBA_Oct22-2016.tsv,/home/shubham/FYProject/ss/MANvsSUN_Dec26-2016.tsv,/home/shubham/FYProject/ss/
MANvsSWA_Apr30-2017.tsv,/home/shubham/FYProject/ss/MANvsTOT_Dec11-2016.tsv,/home/shubham/FYProject/ss/MANvsWAT_Feb11-2017.tsv,/home/shubham/FYPr
oject/ss/MANvsWBA_Apr1-2017.tsv,/home/shubham/FYProject/ss/MIDvsWBA_Feb1-2017.tsv,/home/shubham/FYProject/ss/MIDvsWHU_Jan21-2017.tsv,/home/shubh
am/FYProject/ss/MNCvsARS_Dec18-2016.tsv,/home/shubham/FYProject/ss/MNCvsBOU_Sep17-2016.tsv,/home/shubham/FYProject/ss/MNCvsBUR_Jan2-2017.tsv,/ho
me/shubham/FYProject/ss/MNCvsCRY_May6-2017.tsv,/home/shubham/FYProject/ss/MNCvsEVE_Oct15-2016.tsv,/home/shubham/FYProject/ss/MNCvsHUL_Apr8-2017.
tsv,/home/shubham/FYProject/ss/MNCvsLEI_May13-2017.tsv,/home/shubham/FYProject/ss/MNCvsLIV_Mar19-2017.tsv,/home/shubham/FYProject/ss/SOUvsSTK_Ma
```

```
shubham@shubham-Inspiron-5548: ~/FYProject/stanford-ner-2017-06-09
File  Edit  View  Search  Terminal  Help
serializeTo=/home/shubham/FYProject/ss/ner-model.ser.gz
maxNGramLeng=6
useNGrams=true
usePrev=true
useNext=true
maxLeft=1
map=word=0,answer=1
useWord=true
useTypeSeqs=true
numFeatures = 100899
Time to convert docs to feature indices: 2.4 seconds
numClasses: 4 [0=O,1=PLAYER,2=STADIUM,3=MANAGER]
numDocuments: 78
numDatums: 45647
numFeatures: 100899
Time to convert docs to data/labels: 1.1 seconds
numWeights: 834072
QNMinimizer called on double function of 834072 variables, using M = 25.
                An explanation of the output:
Iter            The number of iterations
evals           The number of function evaluations
SCALING         <D> Diagonal scaling was used; <I> Scaled Identity
LINESEARCH      [## M steplength]  Minpack linesearch
                    1-Function value was too high
                    2-Value ok, gradient positive, positive curvature
                    3-Value ok, gradient negative, positive curvature
                    4-Value ok, gradient negative, negative curvature
                [.. B]  Backtracking
VALUE           The current function value
TIME            Total elapsed time
|GNORM|        The current norm of the gradient
{RELNORM}       The ratio of the current to initial gradient norms
AVEIMPROVE      The average improvement / current value
EVALSCORE       The last available eval score

Iter ## evals ## <SCALING> [LINESEARCH] VALUE TIME |GNORM| {RELNORM} AVEIMPROVE EVALSCORE
```



An NER model will then be serialized to the location specified in the properties file (ner-model.ser.gz) once the program has completed.

Training One Model from Multiple Files:

Instead of setting the trainFile property or flag, set the trainFileList property or flag. Use a comma separated list of files.

Testing and Validation of the Model

Cross Validation:

Cross-Validation is a fair technique to properly estimate model prediction performance because it combines (averages) measures of fit (prediction error) to derive a more accurate estimate of model prediction performance.

K-fold Cross Validation:

In k-fold cross-validation, the original sample is randomly partitioned into k equal sized subsamples.

Of the k subsamples, a single subsample is retained as the validation data for testing the model, and the remaining k − 1 subsamples are used as training data.

The cross-validation process is then repeated k times (the folds), with each of the k subsamples used exactly once as the validation data.

The k results from the folds can then be averaged to produce a single estimation

The advantage of this method over repeated random sub-sampling is that all observations are used for both training and validation, and each observation is used for validation exactly once.

The 100 reports into 5 groups with 20 files each and then K-fold Cross Validation Technique has been used.

The test results in one of the Instances of the Validation Technique are:

```
ARSvsBOU:Nov27-2016
CRFClassifier tagged 584 words in 1 documents at 2834.95 words per second.
         Entity P       R       F1      TP      FP      FN
        MANAGER 1.0000  0.5000  0.6667  1       0       1
         PLAYER 0.9063  0.9063  0.9063  29      3       3
        STADIUM 1.0000  1.0000  1.0000  1       0       0
         Totals 0.9118  0.8857  0.8986  31      3       4



ARSvsBUR:Jan22-2017
CRFClassifier tagged 592 words in 1 documents at 2497.89 words per second.
         Entity P       R       F1      TP      FP      FN
        MANAGER 1.0000  0.6667  0.8000  2       0       1
         PLAYER 0.9268  1.0000  0.9620  38      3       0
         Totals 0.9302  0.9756  0.9524  40      3       1



ARSvsCHE:Sep24-2016
CRFClassifier tagged 387 words in 1 documents at 2303.57 words per second.
         Entity P       R       F1      TP      FP      FN
        MANAGER 1.0000  1.0000  1.0000  1       0       0
         PLAYER 0.9583  1.0000  0.9787  23      1       0
        STADIUM 0.0000  0.0000  0.0000  0       0       1
         Totals 0.9600  0.9600  0.9600  24      1       1



ARSvsCRY:Jan1-2017
CRFClassifier tagged 485 words in 1 documents at 2255.81 words per second.
         Entity P       R       F1      TP      FP      FN
         PLAYER 0.9667  1.0000  0.9831  29      1       0
        STADIUM 1.0000  1.0000  1.0000  1       0       0
         Totals 0.9677  1.0000  0.9836  30      1       0



ARSvsEVE:May21-2017
CRFClassifier tagged 715 words in 1 documents at 3042.55 words per second.
         Entity P       R       F1      TP      FP      FN
        MANAGER 0.3333  1.0000  0.5000  1       2       0
         PLAYER 0.9091  0.9524  0.9302  40      4       2
        STADIUM 1.0000  0.5000  0.6667  1       0       1
         Totals 0.8750  0.9333  0.9032  42      6       3



ARSvsLEI:Apr27-2017
CRFClassifier tagged 581 words in 1 documents at 2766.67 words per second.
         Entity P       R       F1      TP      FP      FN
        MANAGER 1.0000  1.0000  1.0000  2       0       0
         PLAYER 0.9355  1.0000  0.9667  29      2       0
        STADIUM 0.0000  0.0000  0.0000  0       0       1
         Totals 0.9394  0.9688  0.9538  31      2       1
```

Match Details:

HOME TEAM	STOKE CITY
AWAY TEAM	SWANSEA CITY
SCORE	3-1
MATCH STATUS	FULL TIME
DATE	1st NOV,2016

Output:

In the output, the first column is the input tokens, the second column is the correct (gold) answers, and the third column is the answer guessed by the classifier.

which	O	O
arrived	O	O
after	O	O
Wayne	PLAYER	PLAYER
Routledge	PLAYER	PLAYER
s	O	O
equaliser	O	O
.	O	O
came	O	O
via	O	O
an	O	O
Alfie	PLAYER	PLAYER
Mawson	PLAYER	PLAYER
own	O	O
goal	O	O
.	O	O
It	O	O
leaves	O	O
American	O	O
boss	O	O
Bob	MANAGER	MANAGER
Bradley	MANAGER	MANAGER
with	O	O
one	O	O
point	O	O
from	O	O
his	O	O

The code then evaluates the performance of the classifier for entity level precision, recall, and F1.

Results obtained are:

Entity	P	R	F1	TP	FP	FN
MANAGER	1.0000	1.0000	1.0000	4	0	0
PLAYER	0.9737	0.9024	0.9367	37	1	4
Totals	0.9762	0.9111	0.9425	41	1	4

Abbreviation	Quantity	Description
P	Precision	The precision is the ratio tp / (tp + fp) where tp is the number of true positives and fp the number of false positives
R	Recall	The recall is the ratio tp / (tp + fn) where tp is the number of true positives and fn the number of false negatives
F1	F1 Score	Harmonic mean of precision and recall.
TP	True Positives	Number of True Positives
FP	False Positives	Number of False Positives
FN	False Negatives	Number of False Negatives

CHAPTER 5:CONCLUSION

Insights Gained

CRF (Conditional Random Fields) perform and provide the most accurate results when you are working on data assuming dependence between the words.Choosing CRF over other Machine Learning methods is much better option since the CRF takes into consideration the

Significant Achievements

The CRF Model trained has been successful in classifying the entities with accuracies upto 80 percent in terms of PLAYER and 85 percent in terms of MANAGER.

Future Scope

The current project can be expanded to include the Relationship Extraction between the entities.
Also predicting the actual starting eleven and the predicting the match outcome from the results can be done.

Example : Predicting the Team for which a Player plays.

CHAPTER 6:REFERENCES

[1]English Premier League:
https://en.wikipedia.org/wiki/Premier_League

[2] Stanford Named Entity Recogniser:
https://nlp.stanford.edu/software/CRF-NER.html

[3]ESPN Site for Reports:
http://www.espn.in

[4]Cross Validation:
https://en.wikipedia.org/wiki/Cross-validation_(statistics)

[5]Beautiful Soup Documentation:
https://www.crummy.com/software/BeautifulSoup/bs4/doc/

[6] CRF Classifier:
https://en.wikipedia.org/wiki/Conditional_random_field
https://nlp.stanford.edu/nlp/javadoc/javanlp/edu/stanford/nlp/ie/crf/CRFClassifier.html