Recommendation Letter from Thesis Supervisor

The thesis entitled ***Fake News Detection in Social Media Using Machine Learning*** submitted by the students

1. Arnab Sen Sharma
2. Maruf Ahmed Mridul

is a record of research work carried out under my supervision and I, hereby, approve that the report be submitted in partial fulfillment of the requirements for the award of their Bachelor Degrees.

Signature of the Supervisor:
Md Saiful Islam
Date: 18th March, 2018

Certificate of Acceptance of the Thesis

The thesis entitled ***Machine Learning Approach for Fake News Detection*** submitted by the students

1. Arnab Sen Sharma
2. Maruf Ahmed Mridul

on 18th March, 2018 is, hereby, accepted as the partial fulfillment of the requirements for the award of their Bachelor Degrees.

Head of the Dept.	**Chairman, Exam. Committee**	**Supervisor**
Dr Mohammad Reza Selim	M. Jahirul Islam, PhD, PEng	Md Saiful Islam
Professor	Professor	Assistant Professor
Department of CSE	Department of CSE	Department of CSE

Abstract

Easy access of Internet and hyperactivity of users in various social media platforms has given rise to the extensive spread of fake news. Internet has largely replaced traditional news media. Many people, specially a huge portion of youth depend on Internet and social media as the primary source for news consumption because of their easy access, low cost and 24/7 availability. But, some people are taking the advantages of these to promote the spread of fake news. The purpose of this research is to detect those fake news for both Bangla and English languages. We propose a hybrid model to address the problem that works on different features and combines different models and classifiers to find the best solution. A better performance is expected as multiple features and classifiers will be used.

Keywords: Fake News Detection, Natural Language Processing, TF-IDF, Fact-checking, Machine Learning.

Acknowledgements

We would like to thank the Department of Computer Science and Engineering, Shahjalal University of Science and Technology, Sylhet 3114, Bangladesh, for supporting this research. We are very thankful to our honorable supervisor Md Saiful Islam for his worthy supports and directions.

Contents

List of Figures

List of Tables

Abbreviations

NLP	:	Natural Language Processing
SVM	:	Support Vector Machine
ANN	:	Artificial Neural Network
CNN	:	Convolutional Neural Network
RNN	:	Recurrent Neural Network
PAC	:	Passive Aggressive Classifier
TF-IDF	:	Term Frequency - Inverse Document Frequency
CBOW	:	Continuous Bag Of Words
NLTK	:	Natural Language Toolkit

Chapter 1

Introduction

"A lie gets halfway around the world before the truth has a chance to get its pants on."

—Winston Churchill

The threat of fake news is not a new concept. But the term "fake news" was almost non-existent in the general context prior to the presidential election of the USA held in 2016[12]. But in the recent years it has become a real threat that can not be ignored anymore.

1.1 What is Fake News

The term fake news has been used in multiple dimensions and there are multiple definitions of the term. For example, the New York Times defines fake news as - *"a made up story with an intention to deceive."* According to this definition a news would not be categorized as *fake* if the news is the result of an honest mistake.

Wikipedia says that - *Fake news is a type of yellow journalism or propaganda that consists of deliberate misinformation or hoaxes spread via traditional print and broadcast news media or online social media.*[13]

We propose an automated system based on Machine Learning and Natural Language Processing to address the problem.

1.3 Our Approach to the Solution

First of all, till now, there is no existing work regarding fake news detection for Bangla news. We want to start this. Our aim is to do the task for Bangladeshi news. As, news spread both in Bangla and English here, we want to prepare our system in such a way that it will work for both of the languages.

Next, almost everyone has chosen one feature for the task of fake news detection. We plan to merge multiple features targeting a higher accuracy. We propose a hybrid model that will not focus on only one feature, rather take on count some additional features and will work for two different languages.

The Headline and body text of the news, publisher, who likes the news, the reviews of the news everything will be considered as important features for the task.

We'll feed these features to multiple models and classifiers (PAC, CNN, RNN, Logistic Regression etc) and calculate the final result considering all the predictions from these.

We hope that the result will be satisfactory.

Chapter 2

Background Study

2.1 Word Embedding

Word embedding simply refers to vector representation of words. In our problem the input is a news article or some words. Now every machine learning model are not capable of processing string or text as input. These models expects vectors or numbers as input. So, transformation of word to vector is a crucial part. There are several techniques to perform this task. But these techniques are of two types.

1. Frequency based
2. Prediction based

2.1.1 Frequency based Word Embedding

Frequency based Word Embedding techniques mainly focus on the frequency of a word in a text or article to embed a particular word in a vector. There are several such techniques like TF-IDF, Countvectorizer etc.

2.1.1.1 Countvectorizer

In this technique a whole article is converted to a vector, where the dimension of the vector is the number of unique words in the corpus Countvectorizer simply counts the occurrence of each word in an article and puts this count in the respective field of that word in the vector. Lets consider the following two sentences.

1. Abul is very talented but lazy(D1)
2. Babul is hardworking but not very talented(D2)

So, the unique words in our corpus are
[’Abul’ , ’Babul’ , ’but’ , ’is’ , ’very’ , ’not’ , ’talented’ , ’lazy’ , ’hardworking’] According to Countvectorizer technique the dociments D1 and D2 would be vectorized as following

	Abul	Babul	but	is	very	not	talented	lazy	hardworking
D1	1	0	1	1	1	0	1	1	0
D2	0	1	1	1	1	1	1	0	1

Table 2.1: Countvectorizer

2.1.1.2 TF-IDF

Here TF stands for ’Term Frequency’ and IDF stands for ’Inverse Document Frequency’. It is used in text mining as an weighting factor for features. TF is going to be upweighted by the number of times a term occurs in an article. And IDF is upweighted by the number of times a term occurs in the whole dataset. The equation representing the TF-IDF weight of a term **t** for a particular document **d** (given the dataset **D** and the number of documents in the dataset **N**) is given by the equation

$$tf - idf(t,d) = tf(t,d) * idf(t,D)$$
$$Here,$$
$$tf(t,d) = \text{Frequency of } t \text{ in } d$$
$$idf(t,D) = log(\frac{N}{|\{d \epsilon D : t \epsilon d\}|})$$

Like Countvectorizer, TF-IDF also converts a whole article or document to a vector. But in stead of frequency count each field contains the TF-IDF value of the word in the document.

2.1.2 Prediction based Word Embedding or Word2Vec

Word2Vec is the process of transforming words to vectors preserving some of their syntactical and semantic correlations. In CountVectorizer and TF-IDF the vectors of two words are similar or closer to each other based on their occurrence in a particular document and they do not preserve any syntactical and semantic correlations. Word2Vec tries to determine the meaning of a word and understand its correlation with other words by looking at it’s context. For example, lets take two sentences R̈ange Rover is great car.änd R̈ange Rover is a wonderful vehicle.; then Word2vec will map similarities between the words ’great’ and ’wonderful’ and the words ’car’ and ’vehicle’.
Word2Vec uses cosine distance over euclidean distance to measure the similarity or distance between two words. Now, the relation between two words is defined by the vector offset between two words.

Lets take some pair of singular-plural words like cat and cats, dog and dogs.

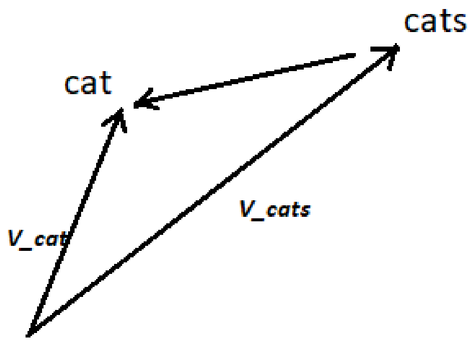

Figure 2.1: Word2Vec (a)

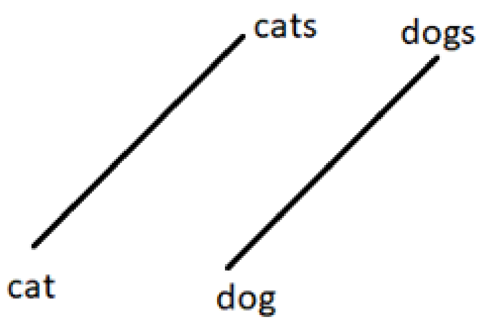

Figure 2.2: Word2Vec (b)

Here the singular-plural relation between the words cat and cats is represented by the cosine difference between the two words V_{cat} and V_{cats} (2.1) is given by the equation below

$$cosine(V_{cat}, V_{cats}) = \frac{V_{cat} * V_{cats}}{||V_{cat}|| * ||V_{cats}||}$$

And now the word pair dog and dogs have the same singular-plural relationship between them like cat and cats (2.2). So according to Word2Vec

$$V_{dog} - V_{dogs} = V_{cat} - V_{cats}$$

$$=> V_{dogs} = V_{cats} - V_{cat} + V_{dog}$$

So, if we know the particular relationship between two words and know one of the words, Word2vec can predict the other word.

Word2Vec is actually a combination of two algorithms or techniques. They are:

1. CBOW(Continuous Bag Of Words)

2. Skip-gram model

2.1.2.1 CBOW(Continuous Bag Of Words)

CBOW is a neural network with one hidden layer. The aim of CBOW is to predict the next word given a sequence of words. Lets take a sentence ’Maradona is a great footballer’. So, if CBOW is given the word sequence ’Maradona is a great’ it will predict the next word ’footballer’. The given word sequence is called *context*. Now, this context can be a single or multiple words. Let’s consider a context of four words ’Maradona is a great’.
At first the words are encoded into a one hot vector like the following.

	Maradona	is	a	great	footballer
Maradona	1	0	0	0	0
is	0	1	0	0	0
a	0	0	1	0	0
great	0	0	0	1	0
footballer	0	0	0	0	1

Table 2.2: Matrix for CBOW of the given example

Now each of these vectors are given as a input field in CBOW model. For this example the CBOW model will look like following.

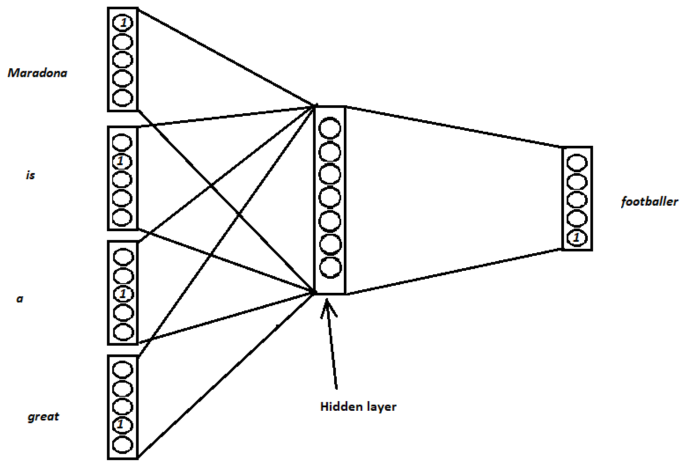

Figure 2.3: CBOW model for the given example

The activation function between input and hidden layer is *linear* and between hidden layer and output layer is *softmax*. A set of context-target pairs are fed to CBOW model. The CBOW model learns the weights between layers like normal neural network. Now the weights between the hidden layer and output layer acts as a vector for the target word.

2.1.2.2 Skip-gram model

The idea of Skip-gram model is somewhat reverse of CBOW. Given a word Skip-gram model predicts the possible contexts for this word. So the diagram of the model is like the reverse of CBOW model.

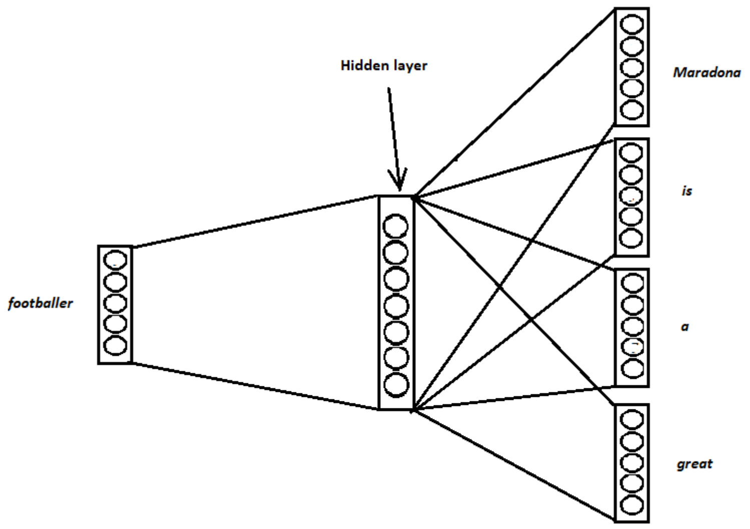

Figure 2.4: Skip-gram model for the previous example

The main idea is like CBOW. The words in an article or document are encoded into one hot vectors. One hot vector of one word acts as the input of the model. The contexts are the output. Like CBOW between input layer and hidden layer the activation function is *linear* and between hidden layer and output layer the activation function is *softmax*.
Given a set of word-context pairs what this model learns about a word is the possible contexts this word can be used in. After learning process the weight matrix between the hidden layer and the output layer acts as a lookup table for vector representations of words. If we multiply the encoded one hot vector of a word with this weight matrix we would get the embedded word2vec representation for the word.

$$\underset{\substack{\uparrow \\ \text{One Hot Vector}}}{[\ 0\ 1\ 0\ 0\ ]}\ \mathrm{X} \underset{\substack{\uparrow \\ \text{Weight Matrix}}}{\begin{bmatrix} w11 & w12 & w13 & \ldots\ldots & w1n \\ w21 & w22 & w23 & \ldots\ldots & w2n \\ w31 & w32 & w33 & \ldots\ldots & w3n \\ w41 & w42 & w43 & \ldots\ldots & w4n \end{bmatrix}} = \underset{\substack{\uparrow \\ \text{Embedded Vector}}}{[\ w21\ w22\ w23\ \ldots\ldots\ w2n\ ]}$$

Figure 2.5: Skip-gram Matrix

Lets take two words ’burger’ and ’pizza’. Both of these words are example of fast foods. So, in our corpus these two words would be used in similar contexts. So the cosine distance between their vectors would be small. Whereas the words ’burger’ and ’aeroplane’ are two different words and they are used in different contexts. So the distance between their vectors would be large.

2.2 Decision Tree

Decision Tree is a type of supervised learning method. This is a tree like model that is used for regression and classification tasks.This is something that predicts the value of a target variable analyzing simple decision rules inferred from the data features.[14][15] The following figure illustrates the concept of decision tree.

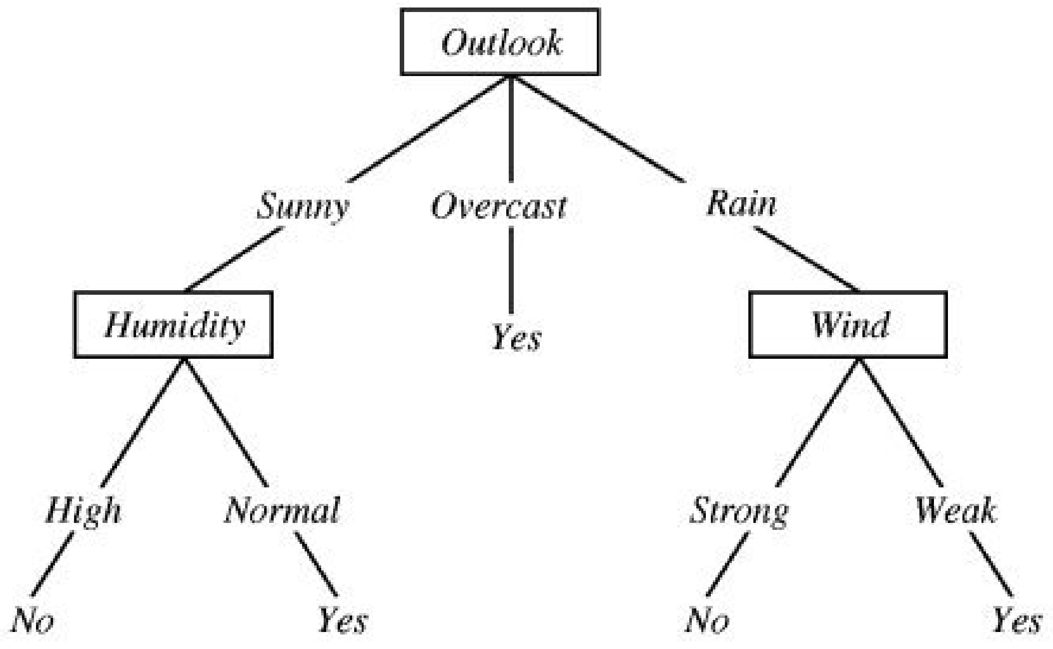

Figure 2.6: Decision Tree[10]

Decision Trees work if the dataset is formed in such a way that the outputs are given for some specific scenario and we want to predict an output based on a portion of the scenarios present on the dataset. A very popular classifier named Random Forest Classifier uses the concept of decision tree.

2.2.1 Random Forest Classifier

The term forest stands for multiple trees (decision trees in this case). A random forest classifier classifies things based on the decisions from many decision trees. Some mutually independent decision trees are formed from the given dataset. They generates a decision and vote for their decided class. Random Forest Classifier finally decides which should be the output class and it declares the class winner which has the maximum vote. In the task of fake news detection, this concept might be useful for classifying news.

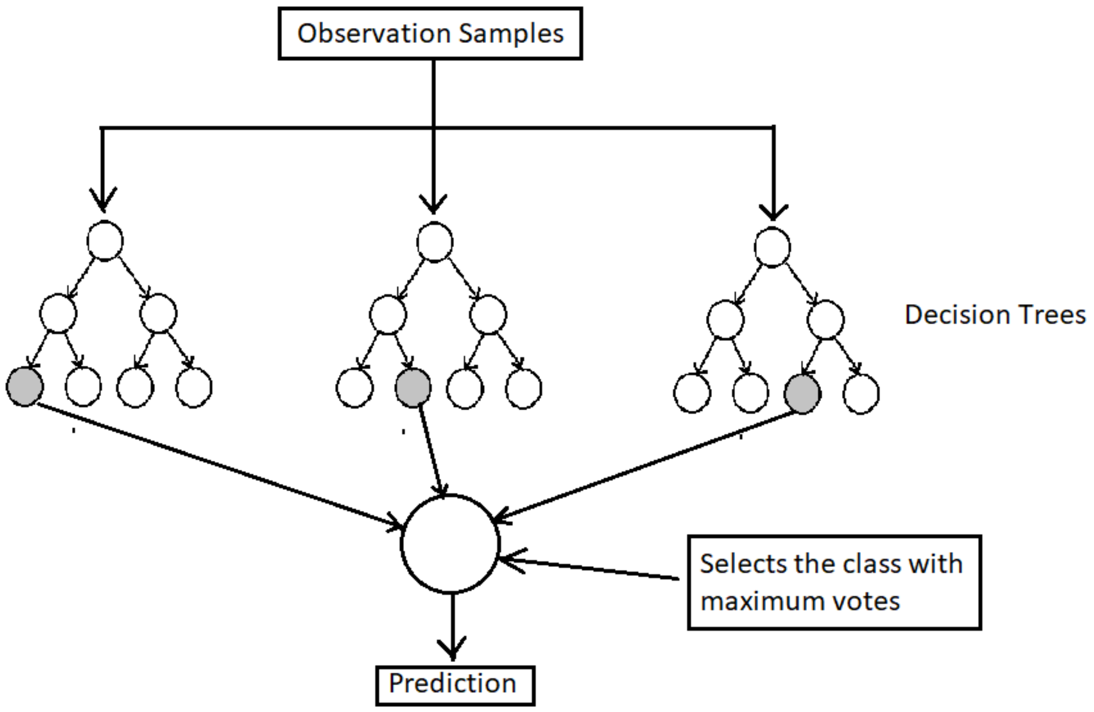

Figure 2.7: Random Forest Classifier

2.3 Support Vector Machines(SVMs)

These are a set of supervised learning methods. An SVM simply finds a separating line for a set of observation samples of two types. So, SVMs can be said to be used for the tasks classification, regression etc. It is a linear binary classifier as it divides only two regions with a line, but still can perform non-linear task very well with a bit of modification. So, the definition of SVM is given in a much more generalized way.

Wikipedia says that - *"A support vector machine constructs a hyperplane or set of hyperplanes in a high- or infinite-dimensional space, which can be used for classification, regression, or other* *tasks like outliers detection"*[16].

So, it works fine for higher dimensional space because of the sparsity of data points in higher dimensions, but fails for too much features(greater than number of data points) in lower dimensions.

In the task of fake news detection, as a news should be classified into one of the two classes (fake or real), SVMs can be used.

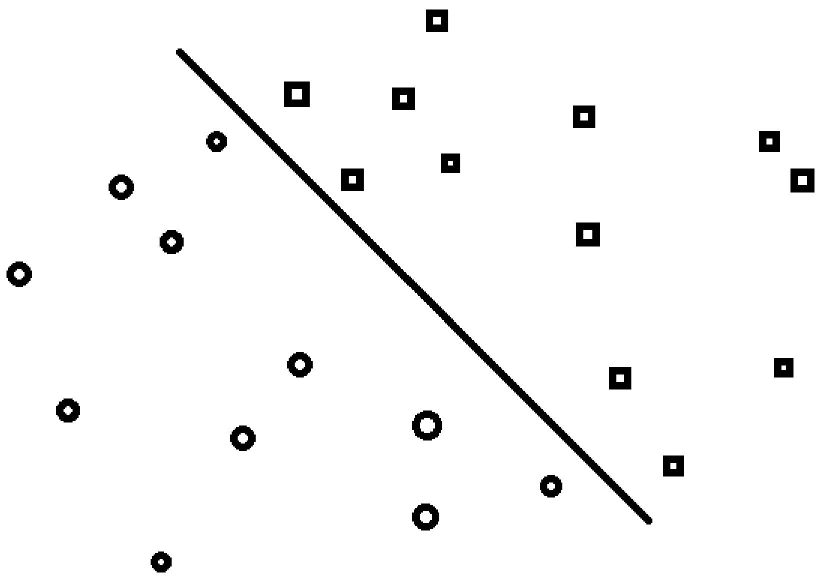

Figure 2.8: SVM

2.4 Passive Aggressive Classifier(PAC)

When the size of the dataset gets bigger and it becomes inefficient to store them in memory, a big problem arises. Traditional classifiers need the dataset stored. So, in this case, these don't seem to perform well. Passive Aggressive Classifier has been introduced to solve this problem. The main idea behind PAC is that, it learns from an example and throws it out. So, it's memory efficient. It doesn't need all the training data to be stored. Except this, PAC works like an SVM.

In our task, we need a huge dataset consisting of words from many news. So, this classifier might be helpful for us.

2.5 Artificial Neural Networks

Artificial Neural Networks are something that were invented to make machines Artificially Intelligent. These are inspired from actual biological neural networks and work similarly. The inputs in these networks are called neurons and the are interconnected by synaptic connections with an weight. These neurons passes signals to each other observing informations and gradually form a strong connection that is intelligent enough to produce correct output for the inputs just like human brains. This is a very sophisticated and powerful idea and is an extremely useful tool for machine learning based applications. We found Recurrent Neural Network(RNN) and Convolutional Neural Network(CNN) (Among many of the types of artificial neural networks) useful for our fake news detection task.

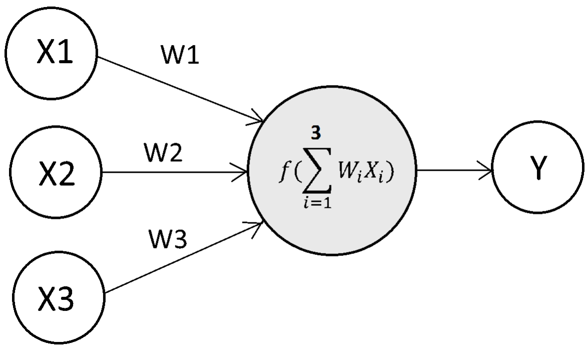

Figure 2.9: Neural Network

2.5.1 Recurrent Neural Network (RNN)

RNN is a type of neural network that is introduced to make use of patterns and sequential information. Normally, we assume that the inputs in a neural network are independent of each other, which may not be a good idea all the time. Sometimes, previous outputs may be necessary to predict current output. This is where recurrence should be introduced. RNN is used in these cases. It takes necessary previous or current outputs as inputs to produce current output. This may have some hidden layers along with the input and output layers. These hidden layers help the network to predict better.

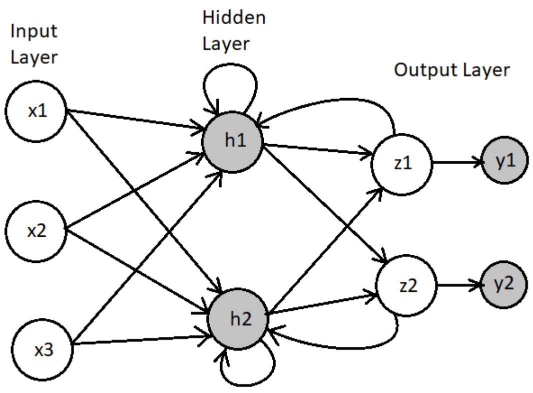

Figure 2.10: Recurrent Neural Network

2.5.2 Convolutoinal Neural Network (CNN)

CNN are responsible for major breakthroughs in image classification and at the core of most of the Computer vision systems today. The input layer of CNN expect the input to be a multi dimensional vector. So, CNN has some convolution layers at the beginning. At each of the convolution layers *convolution* is applied to the multi dimensional vector. Then the results are *feed-forwarded* in local connections. Output of a filter acts as an input for the neuron of next layer. Each layer applies different filters and combines their result and passes the result to the next layer. This process is called *pooling*. Then there is a number of normal hidden layers of neural networks are added. They learn the feature vectors and classify/performs the task at hand.

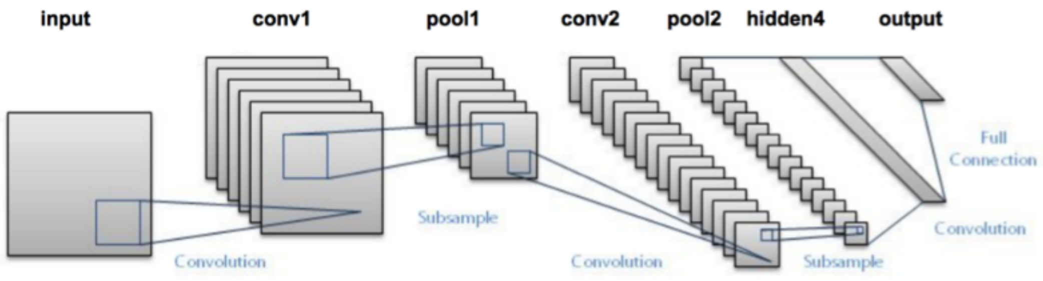

Figure 2.11: Convolutional Neural Network[11]

2.5.2.1 CNN in NLP

Now how can we apply CNN in our task at hand that is to detect whether a news is fake or not. Recently CNNs are used heavily in NLP. CNN expects the input to be a multidimensional image but we have a one dimensional vector from a word after *word embedding*. So, intead of single words we feed CNN whole sentences or documents. From a 20 word document or sentence where each word is embedded to a 200 dimension vector, we can get a 20*200 matrix. This two dimensional vector can act as an image and can be fed as an input to CNN.
From various experiments and researches it was found that CNN performs quite well in generalizing the relationships between words in a sentence. CNN performs better than the simple *bag of words* and prone to less inaccurate assumptions.

2.6 Literature Review

For the task of fake news detection, there are some existing research works mainly for English news. Most of the cases focus on a specific feature to analyze the result. The traditional approaches include-

2.6.1 Linguistic Approaches

2.6.1.1 Deep Syntax Analysis[1]

This method is implemented through Probability Context Free Grammars(PCFG). Some rewrite rules are formed from being transformed from sentential forms. These rewrite rules are then used to make a parse tree with a certain probability assigned. Finally, the parse tree is used to determine whether the syntactical form if the input text is normal or not. Sometimes, intentionally produced fake news do not follow the natural syntax. So, the use of the parse tree to check the syntax of fake news is quite logical. And, this method works with an accuracy of 85-91% (Feng et al.,2012).

2.6.1.2 Semantic Analysis[1]

This method analyzes the truthfulness of the body text of the news based on the reviewers. A truthful review writer is more likely to make true comments about a news. Additionally, A fake news writer who has absolutely no knowledge about the incident/event he/she is writing about may include contradictions or omissions of the facts of the similar topics. This method requires the presence of information about the truthfulness of a review writer in the dataset. Thought this is a problem, this method is pretty good in terms of accuracy. It can predict falsehood approximately 91% accurately.

2.6.1.3 Categorization[2]

According to this procedure, to detect fake news, three subtask should be completed for three types of fake news - fabrication, hoaxing and satire.

- Fabrication : This is the kind of fake news which include crime stories. gossip columns about celebrities, rumors about junk foods etc. These are the results of *Yellow Journalism*.
- Hoaxing : These are the news mistakenly validated by traditional news outlets and are a type of deliberate falsification. These are kind of serious pranks that go beyond merely playful and cause material loss or harm to the victim[2];
- Satire : This is the method of exposing and criticizing people's stupidity or vices by using humor, ridicule or exaggeration. Mainly politics and media are the major issues to be topic of satire. Satire is also a very popular way to make a news fake.

So, Some works are done which independently detected these forms of news eventually resulting in detecting fake news.

2.6.2 Network Approaches

2.6.2.1 Linked Data[1]

Fact checking using the knowledge networks formed by interconnecting the linked data is a popular approach. A well formed knowledge network reduces a fact-checking task to a simple network analysis task. A news can be classified into true or false finding the shortest path from a word to another linked word. The shortest the distance between a word and a known linked word, the higher the likelihood that the corresponding statement is true. This approach gives an accuracy in the range of 61% to 95% for different subject areas.

2.6.2.2 Twitter Graphs[3]

Some research works are done only for the Twitter news. Analyzing the characteristics of the authors, making a graph connecting similar users who have been active in spreading a fake news, facts were checked. Besides, depth of the conversation tree acted as an important feature for false detection. Higher depth indicates higher contradiction in the news. And, there's a huge chance for the contradictory news to be fake. Analyzing the relationship and characteristics of the users who are arguing in the conversation, a prediction is made for the news and this gives quite a satisfactory result.

2.6.2.3 Credibility Network[4]

This is also an approach for the news of twitter. Tweets regarding an issue are connected. Connection are of two types, supporting and opposing. Tweets of same viewpoints(positive/negative) are linked as supporting and different viewpoints are linked as opposing. A weight is assigned to the link based on the strength of connection. Then, a result is calculated from the value found from the credibility network after the convergence of propagation of credibility values. The overall accuracy for this procedure is around 84%.

2.6.3 Naive Bayes[5][6]

Some approaches simply used a naive bayes classifier to classify a news.This is a total probabilistic approach. Firstly 1-grams have been taken from the news context. Then a little bit of preprocessing has been done. In the preprocessing stage, the stop-words(which do not make any sense to describe the news, such as - a, the, to etc.) are eliminated to make the bag of important words. A paper[5] proposed to do a stemming (considering make/making etc. the same word) to increase accuracy. Then the words are fed to a naive bayes classifier to classify the news as fake or real. The accuracy got by the system without preprocessing is nearly 70%[5] and 78.6% with preprocessing in Indonesian Language[6].

2.6.4 CSI Model[7]

CSI stands for Capture Score and Integrate. This is a hybrid model that focuses on each one of the context, the responses and the source of a news instead of traditional choice of a single feature. In this approach, sense making words from the body text of the news are taken and fed to a RNN to detect whether it is fake or not. Again, the response that means reviews of the news are taken as feature and used to classify the news. And finally, the source of the news acts as an important feature for the classification. The information (whether the source generally provides true or false news) of various news source should be in the dataset for this task. Then, this information helps a lot to detect fake news. CSI model is the merged form of these three things. The accuracy for this model is 89.2% for twitter data and 95.3% for weibo data.

2.6.5 Fact-Checking Using Likes (Logistic Regression)[8]

This procedure uses a dataset containing news and the identification of the individuals who liked those. The individuals are given a weight depending on whether they generally likes real or fake news. So, the users are divided into two classes- true likers and false likers. Now, analyzing the likes of true and false likers a news get, the news is categorized as fake or real. A problem arises in this case. If a liker of a news is not listed in the dataset, he/she is not counted and if a news gets only the likes of individuals who are not in the dataset, it can not be classified. To get rid of this problem, an additional step has been taken. A bipartite graph is formed keeping the users in a set and the posts in another set. The connections in the graph are formed by connecting all of the pairs of the nodes. Then, the weights of the users are let to propagate till convergence. Thus, posts with likers not in the dataset make a link with the posts that have known users and it solves the previously arisen problem. This is called the Harmonic Algorithm. The first approach performs with a remarkable accuracy of around 99% and the improvement Harmonic approach extends it to 99.4%.

We also got another dataset from **Kaggle**[19]. We would like to thank it's uploader Megan Risdal for this dataset. We used this dataset for testing our model. This dataset was not exactly formatted as we want our dataset to be. So, we had to modify this dataset a bit to match the dataset we used to train our model.

3.2 Preprocessing

The collected dataset might be mixed with some noisy and unnecessary data. So, we'd to get rid of them through a bit of preprocessing. Our preprocessing consisted of the following steps.

3.2.1 Ignoring stopwords

We ignored the common words 'a' , 'an' , 'the' , 'what' , 'which' , ... etc. Because, these words appear in almost every article and do not provide any significant information needed to classify an article as 'FAKE' or 'REAL'. For now, we just ignored the stopwords available in English language as the stopword list for english is available. We intend to continue this work for Bangla language too.

3.2.2 Stemming

In this step we stemmed each and every word of an article, because we need the word tuples like 'make' , 'making' , 'makes' to refer to same word 'make'. For stemming we used 'SnowballStemmer' and 'PorterStemmer' of pythons *Natural Language Toolkit (**NLTK**)* for stemming. We observed that 'SnowballStemmer' works better than 'PorterStemmer'. So, we used 'SnowballStemmer' for stemming. For now we are only stemming english words, as majority of the news articles in our dataset is of English language.

Chapter 4

Experiments and Results

4.1 Tools Used

We used Anaconda with python-3.5 as our main programming platform. We used various python modules and packages like pandas, sklearn made available to us by Anaconda. We also used Natural Language Toolkit(***NLTK***), Word2Vec library by Gensim. We also used TensorFlow for building and experimenting with various models.

4.2 Plan of Attack

Now that we have acquired the dataset how to attack our task at hand. Our primary idea was to check whether we can determine if a news is 'FAKE' or not based the content of the article only, because we plan to detect all the fake news and rumors in the social media too where a news headline or source of the news is not always available. Our primary plan of attack is to count the number of each word in an article and based on this data come to decision. What we did is that, we gave a value to each unique word based on their frequency in known fake articles and their frequency in known real articles. Initially for each word this value is zero. Now for each occurrence of that word in a fake news article we substract a certain value and for each occurrence in a real news article we add a value. Finally after all articles are processed in this way we define those words which has a negative value as 'fake words' and those words that contain a positive value as 'real words'. Now based on these values we score a article.
Now based on these values we score an article. Our hypothesis is that, if the score of an article is negetive than the article is likely to be fake.

consuming. So we decided not to use SVM as a classifier in future for this specific task.

4.4.3 Random Forest Classifier

Now we used Random Forest Classifier to check whether we could get a better result than naive Bayes tree. Random Forest Tree is the combination of a set of decision trees each built on a random subset of data points. A decision tree splits the data points it is working on in some subsets based on a function *criterion*, which represents the quality of the split. We set this criterion value to *entropy* instead of *gini*, because it showed us better results. Also there is a consideration of how many trees we will use for the task. We set the number of trees to different numbers and got different accuracy measurements. The result are given below.

```
for number of trees= 8 accuracy: 0.832
for number of trees= 10 accuracy: 0.852
for number of trees= 12 accuracy: 0.849
for number of trees= 14 accuracy: 0.851
for number of trees= 16 accuracy: 0.867
for number of trees= 18 accuracy: 0.874
for number of trees= 20 accuracy: 0.877
for number of trees= 22 accuracy: 0.878
for number of trees= 24 accuracy: 0.877
for number of trees= 26 accuracy: 0.882
for number of trees= 28 accuracy: 0.882
for number of trees= 30 accuracy: 0.878
```

4.5 Testing the Models

For testing we used the dataset we acquired from **Kaggle**[19]. This dataset contains the metadata of 244 sites which were flagged as 'bullshit' by **BS Detector**[20] Chrome Extension. So, all of the news in the dataset are fake. This dataset contains 12399 fake articles. After some cleaning and removing some articles of different language we got 9940 articles. Our intention was to check how many of the articles our models recognize as 'FAKE'. This would not be a standard testing, because we are only checking the 'FAKE' news detecting capability of our models not the 'REAL' news detecting capability.

4.5.1 Naive Bayes

Out of 9940 fake articles, our Naive Bayes model tagged 7415 of them as 'FAKE'. Giving us the accuracy rate of 74.6%. Which is not so impressive.

Result of Naive Bayes model
FAKE 7415
REAL 2525
accuracy: 0.746

4.5.2 Random Forest Classifier

Out of 9940 fake articles, our Random Forest Classifier tagged 9115 of them as 'FAKE'. Giving us the accuracy rate of 91.7%. That is not bad considering we are only taking the content of article as our input feature.

Result of Random Forest Classifier
FAKE 9115
REAL 825
accuracy: 0.917

4.5.3 Passive Aggressive Classifier

Like before our Passive Aggressive Classifier has clearly beat all of our other models now too. Out of 9940 fake articles, our Random Forest Classifier tagged 9370 of them as 'FAKE'. Giving us an impressive accuracy rate of 94.3%. This is the highest accuracy rate we could achieve until now.

Result of Passive Aggressive Classifier
FAKE 9370
REAL 570
accuracy: 0.943

4.5.4 XGBoost

Out of 9940 fake articles, our XGBoost model tagged 9175 of them as 'FAKE'. Giving us the accuracy rate of 92.3%.

Result of Passive Aggressive Classifier
FAKE 9175
REAL 765
accuracy: 0.923

4.6 Critical Case Analysis

So, our Passive Aggressive Classifier is working great, giving us above 90% accuracy. But this model may fail in some easy critical cases. Our plan of attack was to measure the frequency of 'FAKE' words in an article to determine whether the article was 'FAKE' or 'REAL'. But. there are some serious faults in this approach. Lets say one of our 'FAKEST' word is 'UFO'. So, if a news article contains the word 'UFO' multiple times, there is a likelihood that this news will be classified as fake. Now, we tested our model giving different articles about 'UFO'. As expected, almost all of them were classified as 'FAKE'. But among the articles there were some actual scientific reports, fragments of wikipedia page etc which were actually 'REAL' news.
Also, we analyzed that most of the news articles we have trained and tested our models with correlate to international politics and business and almost all of them were collected during the time period 2015 to early 2017. So, if we give the model an article about some other topic collected in some other time there is a likelihood that this news would be classified wrongly.
So, we need to update our model in a timely manner and need more data and more features to do better.

4.7 Word2Vec approach

Till now we were considering only TF-IDF values of the unique word in a news article to classify the article as 'REAL' or 'FAKE'. But TF-IDF value depends on only the frequency of the word in the article and in the whole corpus. This TF-IDF value does not represent the semantic meaning of the word. So, surely Word2vec is a better way to represent a word. But, Word2Vec returns a vector for each word whereas our concern is the whole document. So, for each word we took the mean value of Word2Vec vector and then multiplied the value with the TF-IDF value of the word. This way we are loosing some data given by Word2Vec vector, but we are combining some attributes of Word2Vec and TF-IDF. So, now we have one vector per article and we can now train and test our models.
At first, we used the Passive Aggressive Classifier. It gave us the highest accuracy of 91.2% previously. But now Passive Aggressive Classifier gave us a very poor accuracy of 75.9%. The confusion matrix is given below.

Chapter 5

Future Work

5.1 Feature Extraction

The main features we'll use for the task are -

- **Headline :** This feature will be used to verify whether the body text discusses the same thing as the headline.
- **Body Text :** The body text of the news will act as a very important feature. We'll take 1-grams from the text as features.
- **Source :** We'll take the source of the news as a feature. The dataset will contain the information of how many fake news and real news have been published from a specific source.
- **Likes :** The users who likes a news will be extracted as a feature. It can be used to identify the behavior of a user (what the user generally likes - real or fake news).
- **Comments :** Comments will act as a strong feature, as its sentiment might be very helpful for the detection process. So, we'll extract these where comments are available.

5.2 Extraction of Information from the Features

In this step we'll extract the necessary information from the preprocessed features. These information will be used to feed the models. Details about the informations and extraction process are discussed next.

- For headline, body text and comments we'll use word2vec procedure instead of TF-IDF to gain information from the texts. Because, TF-IDF only works with the number of occurrences of words. It doesn't preserve any syntactic or semantic information. But, word2vec does preserve the semantic and syntactic information.

So, we'll transform the words to a vector for further analysis using word2vec.

- For the use of Likers, we've to give a weight to every liker. This weight will be derived from the number of fake news and real news the user liked. After normalizing the weight will be in the range of [-1, 1]. More negative weight defines the user generally likes fake news and more positive weight defines the user generally likes real news.

  The weight of a news will be calculated based on the weights of its likers. This weight will determine the fakeness of the news.

- For the source of news, we'll provide a weight to it depending on the number of published fake and real news. The wight will be in the range [-1, 1] just like the likers. More negative weight defines the source generally publishes fake news and more positive weight defines the source generally publishes real news.

  Similarly as likers, the weight of a news in terms of source will be calculated based on the weight of its source. This weight will determine the fakeness of the news.

5.3 Models and Classifiers

Our target is to use different models and classifiers to predict the result and generate the final result based on analysis of the prediction. We plan to use PAC(Passive Aggressive Classifier), RNN(Recurrent Neural Network) and CNN(Convolutional Neural Network) as the models and classifiers. This prediction process from the information we got from preprocessed data follows some steps.

- First of all, the headline and the body text are brought to do semantic analysis. If the context absolutely opposes the headline we'll conclude from here considering this as a fake news. But, if the headline is a question or the body text discusses similar topic as the headline or even if there is no relation between headline and body text, we'll proceed to next step.

- **Step 1 :** We'll feed the vectors got from applying word2vec on the words of body text to **PAC, RNN** and **CNN**. Then we'll take the prediction which majority of the classifiers have given.

- **Step 2 :** We'll then do the same thing as ***step 1*** for the vectors got from applying word2vec on the words of comments.

- **Step 3 :** Next, we'll have a prediction from the **Logistic Regression**[8] done on the information got from the likers.

- **Step 4 :** After that, we'll have a prediction from the **Logistic Regression**[8] done on the information got from the source of the news.

- **Step 5 :** Finally we'll calculate the final result analyzing the results got from Steps 1, 2, 3 and 4.

  The calculation follows the equations below.

$$\text{Let } A, B, C \text{ and } D \text{ are the outputs of Steps 1, 2, 3 and 4 respectively.}$$
$$A, B, C, D \text{ are either 1 (if the prediction is 'real') or } -1 \text{(if the prediction is 'fake').}$$
$$\text{Let's say } W \text{ is the final output.}$$
$$\text{We calculate,}$$
$$W = 0.5 * A + 0.2 * B + 0.2 * C + 0.1 * D$$
$$\textbf{We conclude that the news is real if } W \geq 0 \textbf{ and fake if } W < 0$$

Chapter 6

Conclusion

Till now, we've experimented mainly with English news articles. But our main focus is to detect fake news in Bangla. To the best of our knowledge, there is no work done on Bangla fake news detection. As this is the first work, we can not guarantee a high accuracy. But, we will try our level best. The domination of fake news in online portals and social media is a threat now-a-days. People are being deceived badly as an effect of this. So, if we can make an efficient system for fake news detection, it will be very helpful.

References

[1] Y. C. Niall J. Conroy, Victoria L. Rubin, “Automatic deception detection: Methods for finding fake news,” vol. 52, pp. 1–4, 01 2015.

[2] N. J. C. Victoria L. Rubin, Yimin Chen, “Deception detection for news: Three types of fakes,” 11 2015.

[3] J. G. Cody Buntain, “Automatically identifying fake news in popular twitter threads,” 2017.

[4] Y. Z. J. L. Zhiwei Jin, Juan Cao, “News verification by exploiting conflicting social viewpoints in microblogs,” 2016.

[5] V. M. Mykhailo Granik, “Fake news detection using naive bayes classifier,” 2017.

[6] F. R. Inggrid Yanuar Risca Pratiwi, Rosa Andrei Asmara, “Study of hoax news detection using naive bayes classifier in indonesian language,” 2017.

[7] Y. L. Natali Ruchansky, Sungyong Seo, “Csi: A hybrid deep model for fake news detection,” 11 2017.

[8] M. L. D. V. S. M. L. d. A. Eugenio Tacchini, Gabriele Ballarin, “Some like it hoax: Automated fake news detection in social networks,” 4 2017.

[9] C. Wardle, “https://firstdraftnews.org/fake-news-complicated/.”

[10] “https://nullpointerexception1.wordpress.com/2017/12/16/a-tutorial-to-understand-decision-tree-id3-learning-algorithm/.”

[11] “https://www.analyticsvidhya.com/blog/2017/06/architecture-of-convolutional-neural-networks-simplified-demystified/.”

[12] B. SARLIN, “https://www.nbcnews.com/politics/politics-news/fake-news-went-viral-2016-expert-studied-who-clicked-n836581.”

[13] “https://en.wikipedia.org/wiki/fake_news.”

[14] “https://en.wikipedia.org/wiki/decision_tree.”

[15] “http://scikit-learn.org/stable/modules/tree.html.”

[16] “https://en.wikipedia.org/wiki/support_vector_machine.”

[17] “https:https://github.com/georgemcintire/fake_real_news_dataset.”

[18] “https://github.com/georgemcintire.”

[19] “https://www.kaggle.com/mrisdal/fake-news.”

[20] “https://github.com/selfagency/bs-detector.”