PROJECT REPORT

On

Fake News Detection Using NLP

Submitted By

Sakshi Chandel
UE173087

Saloni Mohan
UE173088

Samrat Verma
UE173089

Mentored By

Dr. Naveen Aggarwal

Computer Science and Engineering
University Institute of Engineering and Technology
Panjab University, Chandigarh-160014, INDIA
2020

ACKNOWLEDGEMENT

We would like to express our deepest gratitude to all those who provided us the opportunity to work on this project and guided us in achieving the desired results. A special gratitude to Dr. Naveen Aggarwal, whose contribution in stimulating suggestions and encouragement, helped us to coordinate our project especially in writing this report.

We would also like to thank our fellow batch mates who were always ready to help us and improved our project by their suggestions.

ABSTRACT

Fake news encapsulates pieces of news that may be hoaxes and is generally spread through social media and other online media. This is often done to further or impose certain ideas and is often achieved with political agendas. Such news items may contain false and/or exaggerated claims, and may end up being virilised by algorithms, and users may end up in a filter bubble. With the widespread dissemination of information via digital media platforms, it is of utmost importance for individuals and societies to be able to judge the credibility of it. Fake news is not a recent concept, but it is a commonly occurring phenomenon in current times. The consequence of fake news can range from being merely annoying to influencing and misleading societies or even nations. A variety of approaches exist to identify fake news. This project focuses on the deep learning approach. The model is created using NLP and Deep Learning. The Library tensorflow and keras are used to create the model using Bi-LSTM. The models classify whether news is fake or not based on the data it is trained on. The result is either 0 or 1. “1” if the news is fake and “0” if it is not. A comparative study is made by changing the parameters and achieving the highest accuracy using Bi-LSTM.

TABLE OF CONTENTS

1. Introduction……………………………………………………………1
    a. Background……………………………………………………..2
    b. Dataset…………………………………………………………..2
2. Specification and Design………………………………………………3
    a. Objectives……………………………………………………….3
    b. Requirements……………………………………………………3
    c. Libraries…………………………………………………………3
    d. Algorithms………………………………………………………4
    e. Data Pre-processing……………………………..……………7
    f. Model Training…………………………………..……………8
3. Progress Till Date Along With Results……………………..…….……11
    a. Implementation…………………………………………...…….11
    b. Result……………………………………………………...…….11
4. Future Work Plan………………………………………………………12
5. Conclusion………………………………………………………………12
6. References………………………………………………………………13

1. INTRODUCTION

Fake news is a piece of information that is factually incorrect or misrepresents a fact. It sometimes very closely mimics the real news with a slight tilt. It contains-

- Disinformation- an Information which is false and deliberately created to harm a person, social group, organisation or country
- Misinformation- Information that is false but not created with the intention of causing harm.
- Mal-information- Information that is based on reality, used to inflict harm on a person, social group, organisation or country.

Fake news is spread like wildfire through social media platforms such as Facebook, Instagram, and Twitter and so on. Due to less media literacy and repeated exposure to fake news, people tend to believe them.

At best, tech companies such as Google, Facebook, and Twitter have attempted to address this particular concern. However, these efforts have hardly contributed towards solving the problem as the organizations have resorted to denying the individuals associated with such sites the revenue that they would have realized from the increased traffic. Users, on the other hand, continue to deal with sites containing false information and whose involvement tends to affect the reader’s ability to engage with actual news.

These problems make it necessary to eliminate falsified information, however doing so is technically challenging. There are several reliable technologies in the form of Natural Language Processing and Machine Learning that can be used for the purpose of creating models to combat these problems. It is more desirable than the basic countermeasure of comparing websites against a list of labelled fake news sources.

In this project, we aim to build a deep learning model to detect fake news based on the content of the article itself. This is achieved through using different types of training datasets to refine the algorithms. After acquiring a dataset we go on to implement some of the machine learning algorithms with the help of NLP and Deep Learning concepts. Lastly, we provide the end result in the terms of 0 or 1, which represents real or fake news respectively.

A. Background

Data generated from conversations, declarations or even tweets are examples of unstructured data. Unstructured data doesn’t fit neatly into the traditional row and column structure of relational databases, and represent the vast majority of data available in the actual world. It is messy and hard to manipulate. Nevertheless, thanks to the advances in disciplines like machine learning a big revolution is going on regarding this topic. Nowadays it is no longer about trying to interpret a text or speech based on its keywords (the old fashioned mechanical way), but about understanding the meaning behind those words (the cognitive way). This way it is possible to detect figures of speech like irony, or even perform sentiment analysis. It is a discipline that focuses on the interaction between data science and human language, and is scaling to lots of industries.

Today NLP is booming thanks to the huge improvements in the access to data and the increase in computational power, which are allowing practitioners to achieve meaningful results in areas like healthcare, media, finance and human resources, among others. A hybrid approach has already been taken by Abdullah-All-Tanvir to identify fake news using natural language processing from popular twitter threads [1]. Z. Jin and J. Cao suggested verification of news by exploring social views that conflicts with other people [2]. Ray Oshikawa conducted a survey on Natural Language Processing for Fake News Detection [3]. Zhixuan Zhou has also shown how Fake News Detection via NLP is Vulnerable to Adversarial Attacks [4].

B. Dataset

Most crucial thing to train a deep learning model is the availability of a quality dataset of a large quantity. This issue was solved by the number of datasets present on Kaggle. Our only task was to combine those datasets to form a dataset with better quality where each tuple has a label. Further that dataset was divided into two parts for training (a dataset that we feed into our machine learning algorithm to train our model) and testing (also known as validation dataset is a dataset that we use to validate the accuracy of our model but is not used to train the model).

2. SPECIFICATION AND DESIGN

A. Objectives

- Developing the basic understanding of how deep learning models work.
- Developing the basic understanding of how natural language processing works.
- Literature review for current available alternatives and possible new options.
- Acquiring/Generating dataset.
- Implementing current available alternatives for comparative analysis.
- Implementing new neural models for binary classification according to the proposed method.
- Experimenting with the proposed model with various different datasets.
- Comparative analysis between the existing models and our new proposed model.
- Visualizing results for ease of understanding.

B. Requirements

System Requirements (For Developers)-

- A machine with processor equivalent i5 7th Generation
- At Least 8 Gb Ram
- Python 3.8
- Jupyter Notebook

C. Libraries

Tensorflow: TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks.

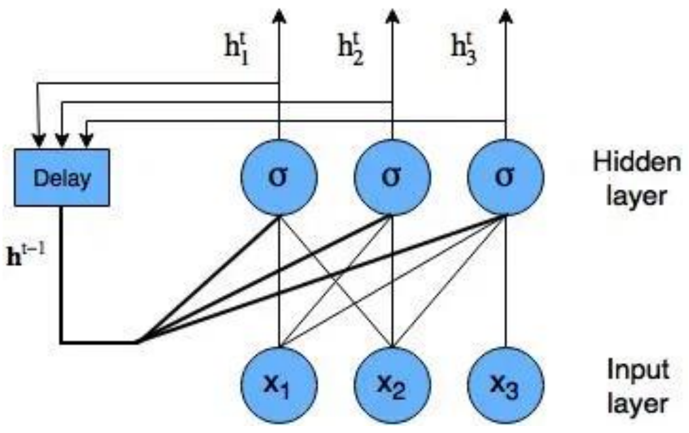
Keras: Keras is an open-source neural-network library written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, R, Theano, or PlaidML. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible.

D. Algorithms

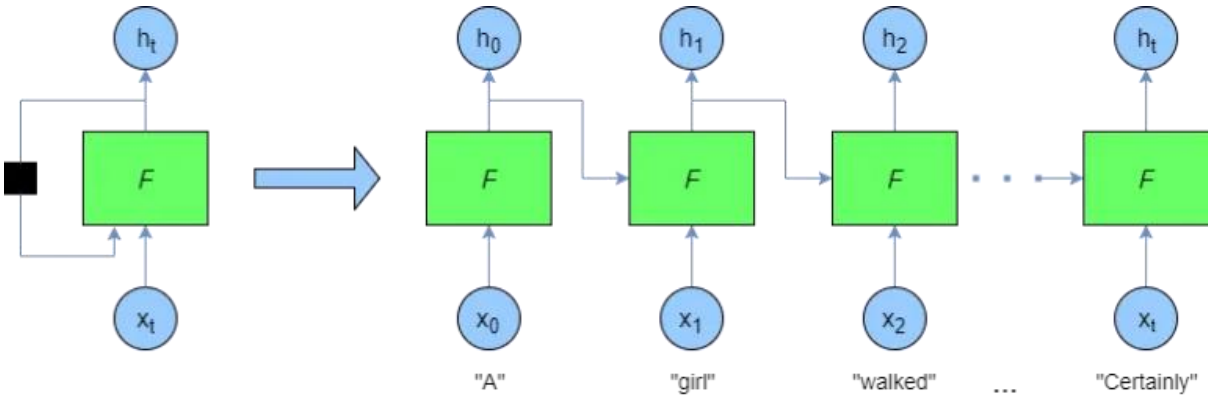
A neural network is a series of algorithms that endeavours to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates. In this sense, neural networks refer to systems of neurons, either organic or artificial in nature. Neural networks can adapt to changing input; so the network generates the best possible result without needing to redesign the output criteria. A neural network works similarly to the human brain's neural network. A "neuron" in a neural network is a mathematical function that collects and classifies information according to a specific architecture. The network bears a strong resemblance to statistical methods such as curve fitting and regression analysis. A neural network contains layers of interconnected nodes. Each node is a perceptron and is similar to a multiple linear regression. The perceptron feeds the signal produced by a multiple linear regression into an activation function that may be nonlinear. In a multi-layered perceptron (MLP), perceptrons are arranged in interconnected layers. The input layer collects input patterns. The output layer has classifications or output signals to which input patterns may map. Hidden layers fine-tune the input weightings until the neural network's margin of error is minimal. It is hypothesized that hidden layers extrapolate salient features in the input data that have predictive power regarding the outputs.

A recurrent neural network is a neural network that attempts to model time or sequence dependent behaviour. This is performed by feeding back the output of a neural network layer at time t to the input of the same network layer at time t + 1.

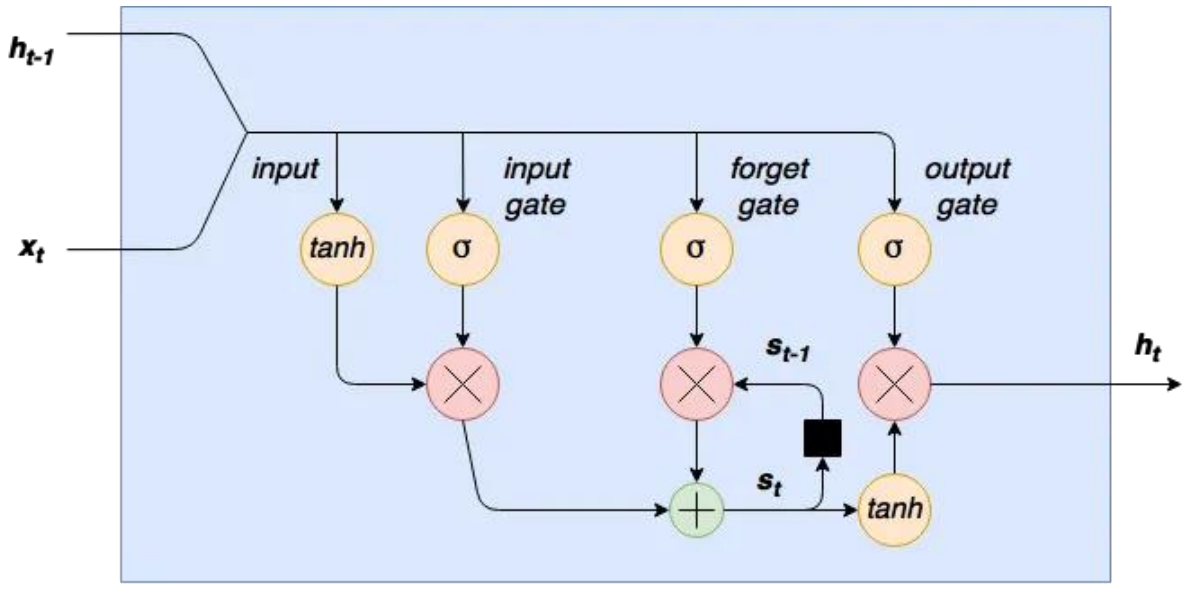
It looks like this:



Recurrent neural networks are "unrolled" programmatically during training and prediction, so we get something like the following:

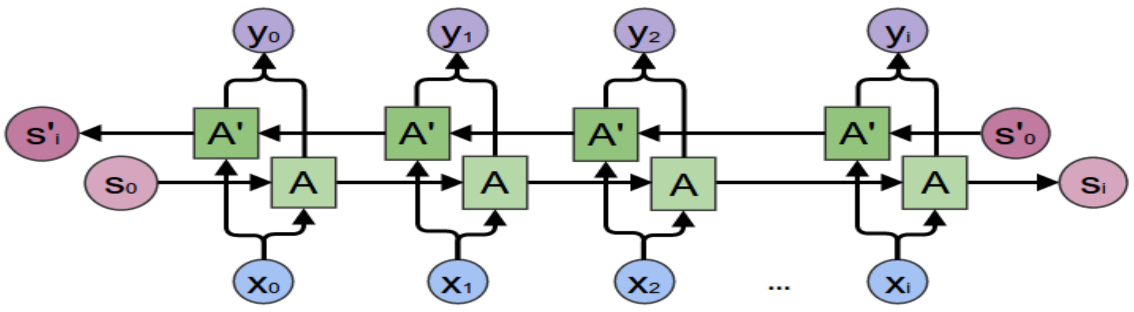


LSTM network: An LSTM network is a recurrent neural network that has LSTM cell blocks in place of our standard neural network layers. These cells have various components called the input gate, the forget gate, and the output gate.



Bi-LSTM (Bi-directional long short term memory): Bidirectional recurrent neural networks (RNN) are really just putting two independent RNNs together. This structure allows the networks to have both backward and forward information about the sequence at every time step

Using bidirectional will run your inputs in two ways, one from past to future and one from future to past and what differs this approach from unidirectional is that in the LSTM that runs backward you preserve information from the future and using the two hidden states combined you are able in any point in time to preserve information from both past and future.



What they are suited for is a very complicated question but Bi-LSTMs show very good results as they can understand the context better,

Bi-LSTM is general architecture that can use any RNN model. We apply forward propagation 2 times, one for the forward cells and one for the backward cells.
Both activations (forward, backward) would be considered to calculate the output y^ at time t

$$\hat{y}^{<t>} = g(W_y\,[\,\overrightarrow{a}^{<t>}, \overleftarrow{a}^{<t>}\,] + b_y)$$

E. Data Pre-processing

In order to get the text data into the right shape for input into the Keras LSTM model, each unique word in the corpus must be assigned a unique integer index. Then the text corpus needs to be reconstituted in order, but rather than text words we have the integer identifiers in order. Finally, the original text file is converted into a list of these unique integers, where each word is substituted with its new integer identifier. This allows the text data to be consumed in the neural network.

- The data has been pre-processed that includes the following steps :
    - Dropping Null or missing values.
    - **Stemming:** It basically cuts off the extra part of a word and converts it into its root form.
    - Substituting all the punctuations present in the dataset.
    - **Stop words removal**: Stop words are very commonly used words (a, an, the, etc.) in the documents. These words do not really signify any importance as they do not help in distinguishing two documents. So, removing is a good way to clean the data.
    - **Lower casing**: Converting all words to lowercase.
    - **Word Embedding**: A word embedding is a class of approaches for representing words and documents using a dense vector representation. In an embedding, words are represented by dense vectors where a vector represents the projection of the word into a continuous vector space.

The position of a word within the vector space is learned from text and is based on the words that surround the word when it is used. The position of a word in the learned vector space is referred to as its embedding.

- ○ One hot encoding creates a matrix that has 0 or 1 values according to the word. Keras provides the one_hot() function that creates a hash of each word as an efficient integer encoding.
- ○ **Padding**: Padding the one hot embedded sentences with “0” value to make them of the same length. We have used pre-padding for our model which will add 0s in the beginning of sentences.

F. Model Training

1. Using LSTM (Long Short-Term Memory)

Initially our model was trained using LSTM. An LSTM network is a recurrent neural network that has LSTM cell blocks in place of our standard neural network layers. These cells have various components called the input gate, the forget gate, and the output gate. The final output from the last point of time is used as the prediction result. It is used for learning long-distance dependency between word sequences in short texts.

LSTM is created using the tensorflow and a sequential model API of keras.
The Layers in this model are:

- Embedding layer
- 1 LSTM layer(100 neurons)
- Dense layer

To compile the model, we have used parameters such as loss, optimizer and metrics.

We are using ‘binary_crossentropy' for loss which is cross-entropy applied in cases where there are only two categories, of which only one is true. The optimizer which we have used is the Adam optimizer – an effective “all-round” optimizer with adaptive stepping. At last, a metric is specified – ‘accuracy', which can let us see how the accuracy is improving during training.

Finally, the model is trained using a number of epochs (50) and batch size (64) as the final arguments.

After 40 epochs, training data set accuracy for the LSTM model was around **0.8785**, while the loss was **0.7540**.

2. Using Bi-LSTM

The Bi-LSTM neural network is composed of LSTM units that operate in both directions to incorporate past and future context information. Bi-LSTM can learn long-term dependencies without retaining duplicate context information. Therefore, it has demonstrated excellent performance for sequential modelling problems and is widely used for text classification. Unlike the LSTM network, the Bi-LSTM network has two parallel layers that propagate in two directions with forward and reverse passes to capture dependencies in two contexts.

The Layers in this model are:

- Embedding layer
- 1 Bi-LSTM layer(100 neurons)
- Dense layer

Using the same parameters which were used in LSTM, accuracy for Bi-LSTM model was found to be around **0.8989** with the loss of **0.7494.**

Vocab_size: It is required to reduce the probability of collisions from the hash function.

Vocab_size	7000	6000	5000
Accuracy	0.8788	0.8818	0.8798

Keras provides the one_hot() function that creates a hash of each word as an efficient integer encoding. We will estimate the vocabulary size of 6000, which is much larger than needed to reduce the probability of collisions from the hash function.

Word Embedding: Word embeddings are in fact a class of techniques where individual words are represented as real-valued vectors in a predefined vector space. Each word is mapped to one vector and the vector values are learned in a way that resembles a neural network, and hence the technique is often lumped into the field of deep learning.

Key to the approach is the idea of using a dense distributed representation for each word.

Each word is represented by a real-valued vector, often tens or hundreds of dimensions. This is contrasted to the thousands or millions of dimensions required for sparse word representations, such as a one-hot encoding.
Keeping Embedding_vector_feature = 100 and varying input_length, change in accuracy is shown in the table below

input_length	50	100	1000
accuracy	0.8495	0.8818	0.8992

Porter Stemmer: We have used **Porter Stemmer algorithm** for stemming. It does not follow linguistics rather a set of 05 rules for different cases that are applied in phases (step by step) to generate stems. It is known for its simplicity and speed.

Without Porter Stemmer	0.8992
With Porter Stemmer	0.9115

3. PROGRESS TILL DATE ALONG WITH RESULTS

A. Implementation

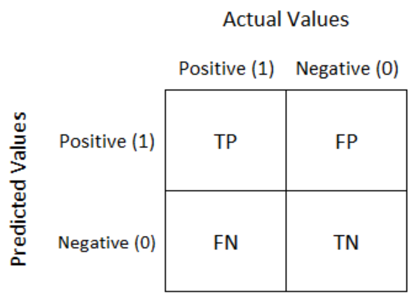
As it has been said earlier, the neural networks model is trained using Bi-LSTM. The data used to train the model was collected from kaggle. It was pre-processed using few of the natural language processing techniques. Only a single layer of Bi-LSTM consisting of 100 neurons has been used in the model. The resultant model is then tested using a testing dataset. Few experiments were done by changing the parameters and a comparative study was done to find the best parameters for the current model.
Final parameters used along with Porter Stemmer are:

- Vocab_size=6000
- Embedding_vector_feature = 10
- input_length=1000

B. Result

In order to test the trained Keras Bi-LSTM model, we can compare the predicted word outputs against what the actual word sequences are in the training and test data set.

We can use the confusion_matrix to get the number of data predicted true and predicted false. It is a performance measurement for machine learning classification problems where output can be two or more classes. It is a table with 4 different combinations of predicted and actual values.



		Actual Values	
		Positive (1)	Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN

The confusion matrix for Bi-LSTM it turned out to be:

```
array ([[3167,  252],
       [282, 2334]])
```

Accuracy

The accuracy score can be checked using ‘accuracy_score’ from sklearn library.

The score turned out to be **0.9115161557580779 or 91.15%** for Bi-LSTM.

4. FUTURE WORK PLAN

Future plans for the project are:

1. To train the model on a much larger dataset for better accuracy.
2. To train the model on a dataset of Indian News.
3. To use a different model like BERT that can increase the accuracy of the result.

5. CONCLUSION

Considering the rise in the number of fake news and how it can mislead a society or a nation, it has become really important to detect these news on social media. This project focuses on solving the concerned situation using a deep learning model which uses LSTM as its one of the layers. Even though this project gives high accuracy in its results, it has a scope for improvement.

6. REFERENCES

[1] Abdullah-All-Tanvir, Ehesas Mia Mahir, S M Asiful Huda & Shuvo Barua, “A Hybrid Approach for Identifying Authentic News Using Deep Learning Methods on Popular Twitter Threads”, 2020 International Conference on Artificial Intelligence and Signal Processing (AISP).

[2] Jin, J. Cao, Y. Zhang, and J. Luo, “News verification by exploiting conflicting social viewpoints in microblogs.” in AAAI, 2016, pp. 2972– 2978.

[3] Ray Oshikawa, Jing Qian, William Yang Wang, “A Survey on Natural Language Processing for Fake News Detection”, 2018 arXiv:1811.00770[cs.CL].

[4] Zhixuan Zhou1,2, Huankang Guan1, Meghana Moorthy Bhat2 and Justin Hsu2, “Fake News Detection via NLP is Vulnerable to Adversarial Attacks”, 2019 arXiv:1901.09657 [cs.SI].