Ain Shams University

Faculty of Computer & Information Science

Information Systems Department

Fake News and COVID-19

By:

Abdelmoniem Ahmed Amer	[Information Systems]
Abdelrahman Hegazi Atia	[Information Systems]
Rafeek Nehad Mostafa	[Information Systems]
Rana Mostafa Hassan	[Information Systems]
Omar Mansour Saad	[Information Systems]

Under Supervision of:

Prof. Dr. Tarek Fouad Gharib

TA. Esraa Karam

TA. Radwa Moustafa

July 2021

Acknowledgement

First and foremost, all praise and due to Allah Subhanahu wa Ta'ala.

We have to thank our supervisors, Prof. Dr. Tarek Fouad Gharib, T.A Esraa Karam and T.A Radwa Moustafa for the help, guidance, and encouragement they provided throughout the year.

Next, we’d like to thank all the helpful classmates who supported us with knowledge or advice.

We’d also like to thank our friends, families for their continuous support throughout the year.

Abstract

Fake news is a phenomenon that has a significant impact on our social life, in particular in the political world. Fake news detection is an emerging research area which is gaining interest but involved some challenges due to the limited number of resources available. The coronavirus disease (COVID-19) pandemic is perhaps the greatest global health challenge of the last century. Accompanying this pandemic is a parallel infodemic, including the online marketing and sale of unapproved, illegal, and counterfeit COVID-19 health products including testing kits, treatments, and other questionable cures.

Enabling the proliferation of this content is the growing ubiquity of internet-based technologies, including popular social media platforms that now have billions of global users. In this project, an approach to detect the fake COVID-19 news in social media is proposed. The proposed approach is conducted in two phases. The first phase is the collection of COVID-19 related articles using different resources of datasets. The second phase is the analysis of data while using natural language processing and different machine learning techniques such as SVM, CNN, and LSTM, where each model achieved accuracy value 74.48%, 89.55% and 94.65% respectively.

Table of Contents

Table of Figures

List of Tables

List of Abbreviations

BERT	Bidirectional Encoder Representation from Transformers
RoBERTa	Robustly optimized BERT approach
CT-BERT	COVID-Twitter-BERT
BiLSTM	Bidirectional LSTM
BNB	Bernoulli Naive Bayes
BoW	Bag of Word
CNN	Convolutional Neural Network
Doc2vec	Document to vector
kNN	K-Nearest Neighbor
LR	Logistic Regression
LSTM	Long short-term memory
LSVM	Linear Support Vector Machines
NB	Naïve Bayes
RF	Random Forest
RNN	Recurrent Neural Network
SVM	Support vector machine.
TF	Term Frequency
TF-IDF	Term Frequency-Inverse Document Frequency
Word2vec	Word to vector

Chapter1

Introduction

1.1 Motivation

In the age of technology, a tremendous amount of data is being generated online every day. However, an unprecedented amount of the data flooded on the Internet is fake, this data is generated to attract the audience, to influence beliefs and decisions of people, increase the revenue generated by clicking, and affect major events such as political elections. So, detecting spread of misinformation such as, rumors, hoaxes, fake news, propaganda, spear phishing, and conspiracy theories, became an important task.

The extensive spread of fake news can have a serious negative impact on individuals and society. First, fake news can break the authenticity balance of the news ecosystem. Second, fake news intentionally persuades consumers to accept biased or false beliefs. Fake news is usually manipulated by propagandists to convey political messages or influence. Third, fake news changes the way people interpret and respond to real news. To help mitigate the negative effects caused by fake news, both to benefit the public and the news ecosystem, it's critical that we develop methods to automatically detect fake news on social media.

1.2 Problem Definition

With COVID-19 pandemic a pressing need for tools to combat the spread of misinformation has surfaced. Since the pandemic affects the global community, there is a wide audience seeking information about the topic, whose safety is threatened by adversarial agents invested in spreading misinformation for political and economic reasons.

Furthermore, it is also difficult to be completely accurate and factual, leading to disagreements that get aggravated with misinformation. This difficulty is compounded by the rapid evolution of knowledge regarding the disease

1.3 Objective

proposing an approach to detect fake news of COVID-19 in social media while using natural language processing and machine learning techniques.

1.4 Document Organization

The rest of the document is organized as follows:

Chapter 2: Background and Related Work

This chapter presents background information related to the targeted research area and a comparison between related works.

Chapter 3: Analysis and Design

This chapter includes a System overview accompanied by system architecture, analysis, and design.

Chapter 4: Implementation

This chapter includes a detailed description of all the functions in the system, a detailed description of all the techniques and algorithms implemented, UI design, and the obtained results.

Chapter 5: User manual

This chapter describes in detail how the user can install the required programs and use the project.

Chapter 6: Conclusion and Future work

This chapter includes a summary of the whole project along with a comparison of the results obtained using the different techniques and the conclusion of the best results reached, and what can be achieved in the near future to improve the performance.

Chapter 2

Background and Related Work

2.1 Background

The fake information of nowadays is inflicting diverse issues in a few media from satirical posts to fabricated news and authority's propaganda software. False news and shortage of media confidence are growing troubles in our lifestyle, with main repercussions. Obviously, "fake news" is a purposely misleading story, but currently blurring the controversy on social media is converting its which means. Some of them are now using the word to disregard the proof about their desired viewpoints. Work is an increasing number of focusing on automated faux information identity and fact-checking, as the need grows due to the harmful information unfolding on social media. Following media coverage, Facebook changed into the epicenter of plenty of complaints. They have also delivered a characteristic to mark false information on the internet when a user sees it; they have additionally publicly stated that they are have working on robotically figuring out these posts. Of course, this isn't a smooth assignment. **[1]**

The content of fake news is rather variety in terms of topics, styles, and media platforms, and fake news attempts to distort the truth with diverse lingual styles while simultaneously mocking true news. For example, fake news may cite true evidence within the in-correct context to support a non-factual claim. Thus, existing hand-crafted and data-specific textual features are generally not sufficient for fake news detection. Other auxiliary information must also be applied to improve detection, such as knowledge base and user social engagements. Second, exploiting this auxiliary information actually leads to another critical challenge: the quality of the data itself. **[2]**

Fake news is usually related to newly emerging, time-critical events, which may not have been properly verified by existing knowledge bases due to the lack of corroborating evidence or claims. In addition, users' social engagements with fake news produce data that is big, incomplete, unstructured, and noisy. Effective methods to differentiate credible users, extract useful post features and exploit network interactions are an open area of research and need further investigations. **[3]**

2.2 Related work

In **[4]** a fake news stance detection model, based on the headline and the body of the news irrespective of the previous studies which only considered the individual sentences or phrases. The proposed model incorporates principal component analysis (PCA) and chi-square with CNN and LSTM, in which PCA and chi-square extract the quality features which are passed to the CNN-LSTM model. First, the data preprocessing phase: convert text to lower case, remove stop words, stemming and tokenization. Then, the output from the preprocessing enters the feature extraction phase where TF-IDF is applied. In last phase the LSTM and

CNN models is applied for the output from the previous phase. The best accuracy was 97.8% and was achieved by CNN+LSTM model.

In **[5]** with COVID-19 pandemic we are also fighting an infodemic. Fake news and rumors are rampant on social media. Believing in rumors can cause significant harm. This is further exacerbated at the time of a pandemic. To tackle this, they curate and release a manually annotated dataset of 10,700 social media posts and articles of real and fake news on COVID-19. This model starts with data preprocessing phase as in this phase it removes all the links and stop words. The next phase TF-IDF is applied for the output from the previous phase and the input from this phase enters the SVM model to start classifying and prediction phase. The achieved accuracy was 93.32%.

In **[6]** the deep learning techniques they used are The Modified-LSTM (one to three layers) and The Modified GRU (one to three layers). In particular, it carry out investigations of a large dataset of tweets passing on data with respect to COVID-19. In this study, they separate the dubious claims into two categories: true and false. They compare the performance of the various algorithms in terms of prediction accuracy. The six machine learning techniques are decision trees, LR, kNN, RF, SVM, and NB. Two feature extraction methods were used (TF-ID with N-gram) to extract essential features from the four benchmark datasets for the baseline machine learning model and word embedding feature extraction method for the proposed deep neural network methods. The best accuracy was 97.8% and it was obtained by SVM and RF model.

In **[7]** they classify the information into two categories: credible or non-credible. The two classifications of tweet credibility are based on various features, including tweet- and user-level features. They conduct multiple experiments on the collected and labeled dataset. The results obtained with the proposed framework reveal high accuracy in detecting credible and non-credible tweets containing COVID-19 information. They used 4 models: kNN, RF, SVM and NB. The best accuracy was 96.57% and was achieved by Unigram and RF model.

In **[8]** they were able to collect a huge amount of dataset consists of 3.263M Arabic and English tweets, and they described in detail their data collection steps and how they conducted the annotation process, also their Exploratory Data Analysis (EDA) was able to show the main features of the COVID-19-Fakes dataset. For example, they weren't only able to explore the insights of the collected tweets but also, visualizing numeric and categorical features in them. They were able to define specifically the verified and not verified users which tweet these huge number of tweets besides their language, country, even their location. They found out whether the tweet is a truth or a misleading news by comparing their truth ground they have collected from trusted organization to the text in tweets, Data preprocessing: Convert text to lower case, Remove stop word. Feature extraction: TF, TF-IDF and N-gram. The models they used are: decision tree, kNN, LR, LSVM, SVM, and BNB.

In **[9]** they report a methodology to analyze the reliability of information shared on social media pertaining to the COVID-19 pandemic. The best approach is based on an ensemble of three transformer models (BERT, ALBERT, and XLNET) to detecting fake news. The methodology: data preprocessing: Emoticon Conversion, Handling of Hashtags, Stemming, Text cleaning. Feature extraction: n-gram level, TF-IDF vectors. Model: LSTM, BiLSTM, CNN. The best accuracy in this model is BiLSTM with 97.8%.

In **[10]** they proposed an ensemble method of different pre-trained language models such as BERT, Roberta, Ernie, etc. with various training strategies including warm-up, learning rate schedule and k-fold cross-validation. We also conduct an extensive analysis of the samples that are not correctly classified. The methodology their approach followed was: preprocessing removes stop words, remove URLs, feature extraction uses a Glove word vector to convert text to vector, classification model text-RNN based on BiLSTM, Text-Transformers based on transformers. They achieved accuracy value equal 98.5% by Text-Transformers module.

In **[11]** the task is composed of two sub-tasks namely 1-text-based, and 2-structure-based fake news detection. For the first task, we propose six different solutions relying on Bag of Words (BoW) and BERT embedding. As for the classification models NB, LR, Graph Neural Networks. The best accuracy is Graph Neural Network with 95%.

In **[12]** they address the problem of detecting hostile and fake content in the script as a multi-class, multi-label problem. Using NLP techniques, we build a model that makes use of an abusive language detector coupled with features extracted via BERT and FastText models and metadata. Methodology: preprocessing Tokenization and removal user name, hashtag, URLs, stop words, emoji, punctuation, new line, Unicode characters. Feature extraction Word2Vec, BoW, TF-IDF, classification model RF, LR, SVM. the SVM model achieve the best accuracy with 93.74%.

In **[13]** the evolution of social media platforms has empowered everyone to access information easily. Social media users can easily share information with the rest of the world. This study follows a methodology consists of: preprocessing lowercasing the words, replacing irrelevant symbols with spaces, Removing stop words. Feature extraction Word2Vec, TF-IDF. Classification model NB, LR, RF, SVM, Boosting models. The best accuracy achieves with SVM with 94.1%.

In **[14]** they propose approach using the transformer-based ensemble of COVID-Twitter-BERT (CT-BERT) models. This study follows a methodology consists of: preprocessing removing or tokenizing hashtags, translating in the lowercase, replace the emoji with short textual description. Further, we converted texts into the form of a token counts matrix BoW. Classification model BERT, RoBERTa, CT-BERT, the best accuracy achieved with BERT-based with 98.69% of the weighted F1-Score.

Table 1 shows a summary for the related works described above.

The methodology we went with: in data preprocessing phase: removing stop words, converting to lower case, normalization, stemming, text cleaning and removing emotions. In feature extraction phase we used embedding and Doc2vec. The Models: SVM, CNN and LSTM.

Table 1 Summary for the related works

Reference number	4	5	6	7	8	9	10	11	12	13	14
BoW								✓	✓		✓
TF					✓						
TF-IDF	✓	✓	✓		✓	✓			✓	✓	
N-gram			✓		✓	✓					
Word2vec									✓	✓	
BERT											✓
RoBERTa											✓
CT-BERT											✓
Glove word vector							✓				
CNN	✓					✓					
Decision Tree			✓		✓						
kNN			✓	✓	✓						
Graph Neural Network								✓			
LR			✓		✓			✓	✓	✓	
RF			✓	✓					✓	✓	
RNN							✓				
Text-Transformers							✓				
SVM		✓	✓	✓	✓				✓	✓	
LSVM					✓						
LSTM	✓					✓					
BiLSTM						✓	✓				
NB			✓	✓				✓		✓	
BNB					✓						

Chapter 3

Analysis and Design

3.1 System Overview

3.1.1 System Architecture

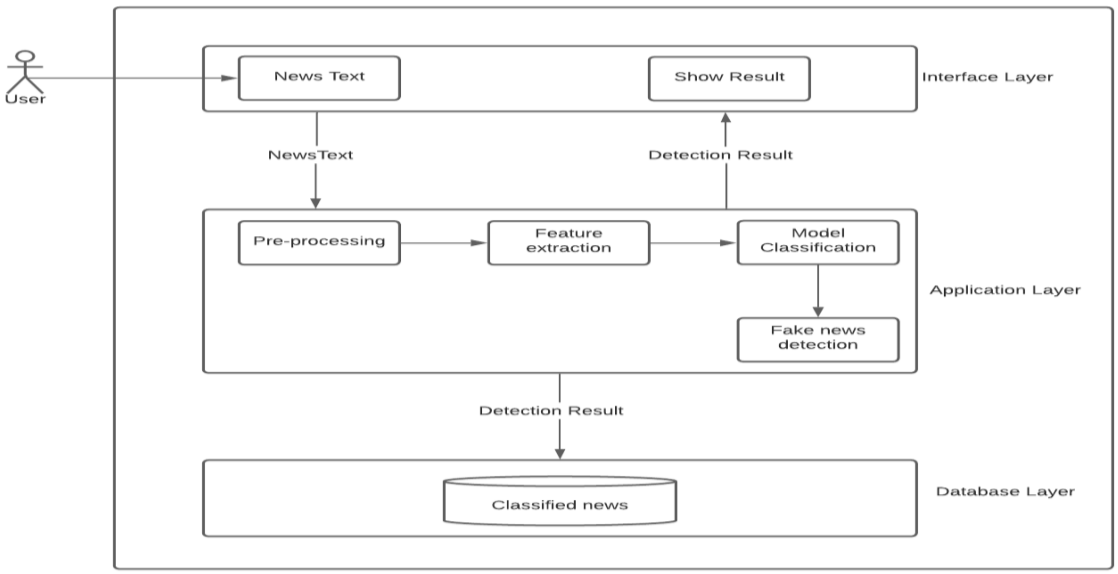


Figure 3.1 System architecture

The proposed architecture shown in fig. 3.1 consists of:

- Interface Layer

    1- News Text: User Input (Text)

    2- Show Result: System Output (Fake/Real)

- Application Layer

    1- Pre-processing: Prepare data for feature extraction. Remove stop words, apply stemming & lemmatization, Convert all word to lower case & splits input text into tokens.

2- Feature Extraction: Apply document-to-vector (Doc2vec), text encoding & embedding on input and extract features.

3- Model Classification: Classify the input from feature extraction to specific class using support vector machine (SVM), convolutional neural network (CNN) & long short-term memory (LSTM).

4- Fake news Detection: Detect if the input text Fake or Real.

- Database Layer

1- Classified news: Store result in the used dataset to use it in future detections for better accuracy.

3.1.2 Functional Requirements

1. The user can input text.
2. The user can download the used dataset by the system.
3. The user can search for COVID-19 news.
4. The user can see the prediction result.
5. The system can upload dataset.
6. The system can pre-process the input data.
7. The system can extract feature from the input data.

8. The system can apply classification models.
9. The system can predict if the input Fake/Real.
10. The system can store prediction result in the dataset.

3.1.3 Nonfunctional Requirements

1. The system is available to all users during any time.

3.1.4 System Users

A. Intended Users:

Any user can use the system & predict if the news fake/real.

B. User Characteristics

The user must have knowledge on how to use internet & the way around websites.

3.2 System Analysis & Design

3.2.1 Use Case Diagram

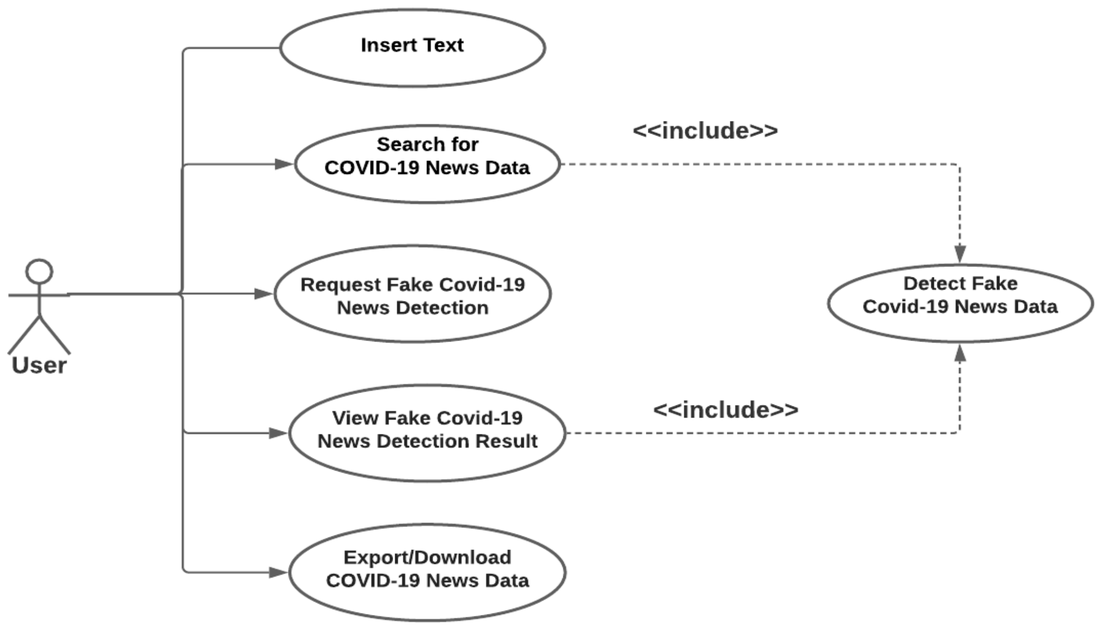


Figure 3.2 Use case diagram

Use Case Description:

1- Insert text: User input text which he wants to predict.

2- Request fake COVID-19 news detection: The user requests the system to perform detection on the input he entered.

3- View fake COVID-19 news detection result: The system shows the final prediction result for the user.

4- Detect fake COVID-19 news: Predict if the news text fake/real.

5- Search for COVID-19 news data: The user can search for specific news using keyword.

6- Export/Download COVID-19 news data: User can download the used dataset by the system.

3.2.2 System Sequence Diagram

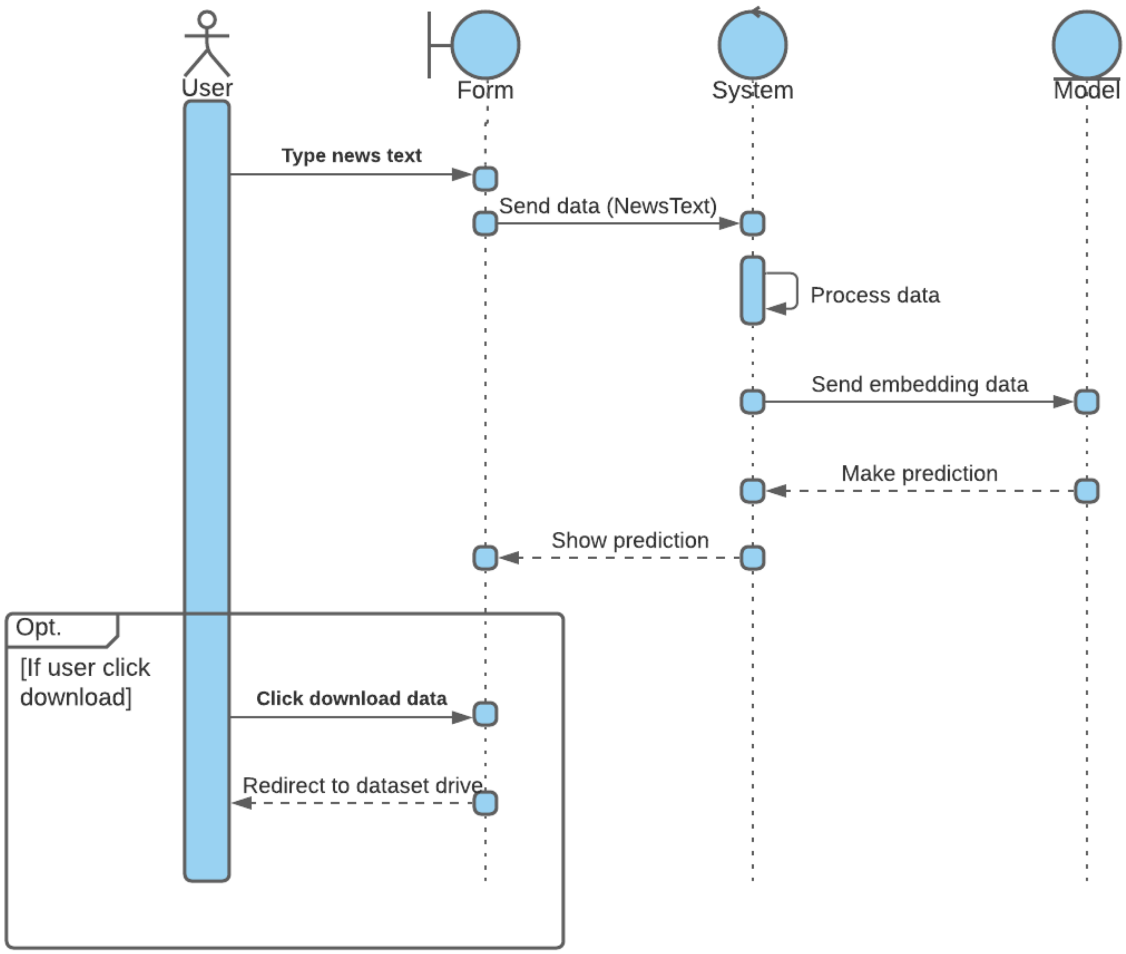


Figure 3 3 System sequence diagram

Chapter 4

Implementation

4.1 Algorithms and Techniques

The details of the examined machine learning techniques are revealed in the following subsections.

4.1.1 Support Vector Machine (SVM)

SVM is a discriminative classifier formally defined by a separating hyperplane. Given labeled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new examples. In two-dimensional space this hyperplane is a line dividing a plane in two parts where in each class lay in either side. Suppose you are given plot of two label classes on graph as shown in Figure 4.1

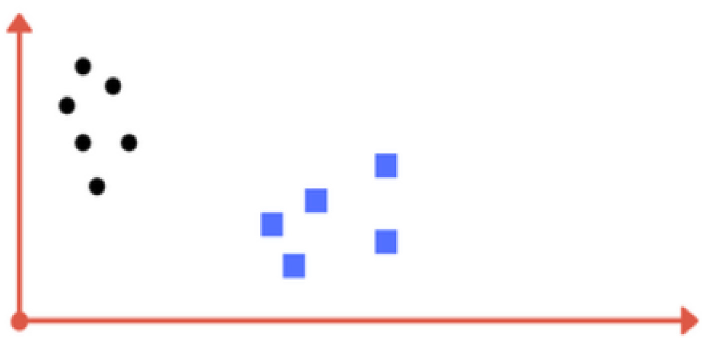


Figure 4.1 Plot of two label classes

You might have come up with something similar to figure 4.2. It fairly separates the two classes. Any point that is left of line falls into black circle class and on right falls into blue square class. Separation of classes. That's what SVM does. It finds out a line/ hyper-plane (in multidimensional space that separate outs classes).

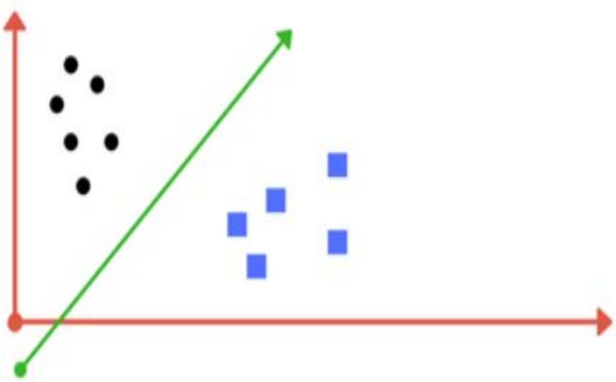


Figure 4.2 Simple cut to divide into two classes

Now consider what if we had data as shown in Figure 4.3 below. Clearly, there is no line that can separate the two classes in this x-y plane. So, what do we do? We apply transformation and add one more dimension as we call it z-axis. Let's assume value of points on z plane, $w = x^2 + y^2$. In this case we can manipulate it as distance of point from z-origin. Now if we plot in z-axis, a clear separation is visible, and a line can be drawn as shown in Figure 4.4.

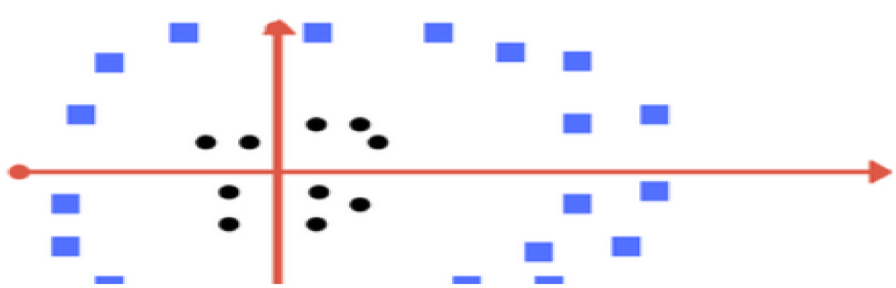


Figure 4.3 Plot of two label classes

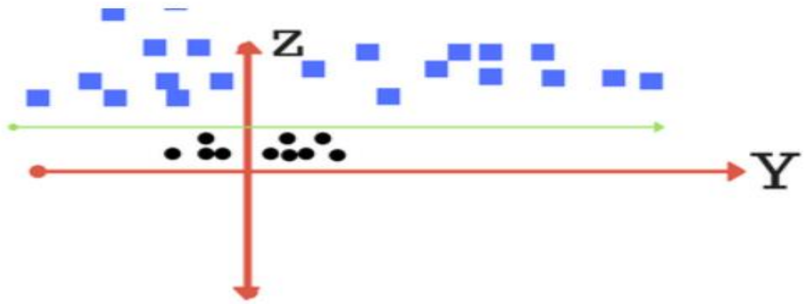


Figure 4.4 Plot of zy axis.

SVM core tries to achieve a good margin. A margin is a separation of line to the closest class points. A good margin is one where this separation is larger for both the classes. Figures below gives to visual example of good and bad margin. A good margin allows the points to be in their respective classes without crossing to other class. **[15]**

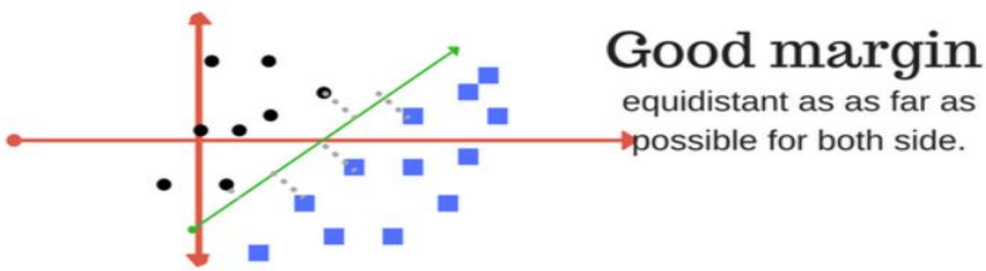


Figure 4.5 Good margin

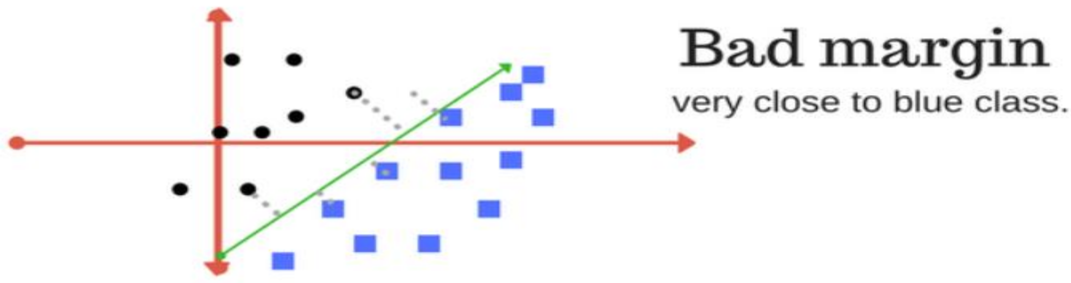


Figure 4.6 Bad margin

- Advantages
    1. SVM works relatively well when there is a clear margin of separation between classes.
    2. SVM is more effective in high dimensional spaces.
    3. SVM is effective in cases where the number of dimensions is greater than the number of samples.
    4. SVM is relatively memory efficiency.

- Disadvantages
    1. SVM algorithm is not suitable for large data sets.
    2. SVM does not perform very well when the data set has more noise.
    3. Gives no probabilistic explanation for the classification.

4.1.2 Recurrent Neural Network (RNN)

Humans don't start their thinking from scratch every second. You understand each word based on your understanding of previous words. You don't throw everything away and start thinking from scratch again. Traditional neural networks can't do this, and it seems like a major shortcoming. For example, imagine you want to classify what kind of event is happening at every point in a movie. It's unclear how a traditional neural network could use its reasoning about previous events in the film to inform later ones. Recurrent neural networks address this issue. They are networks with loops in them, allowing information to persist.

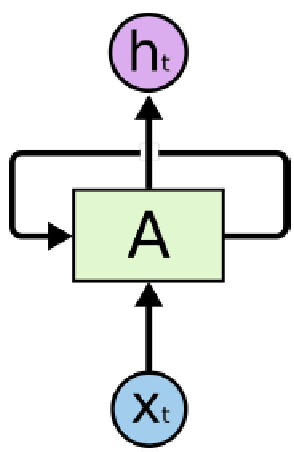


Figure 4.7 Recurrent Neural Networks have loops

As demonstrated in Figure 4.7, a chunk of a neural network, A, looks at some input, xt, and outputs a value ht. A loop allows information to be passed from one step of the network to the next.

RNN is to make use of sequential information, they are called *recurrent* because they perform the same process for every element of a sequence, with the output being depended on the previous computations and it is already known that they have a "memory" which captures information about what has been calculated so far as shown in Figure 4.8.

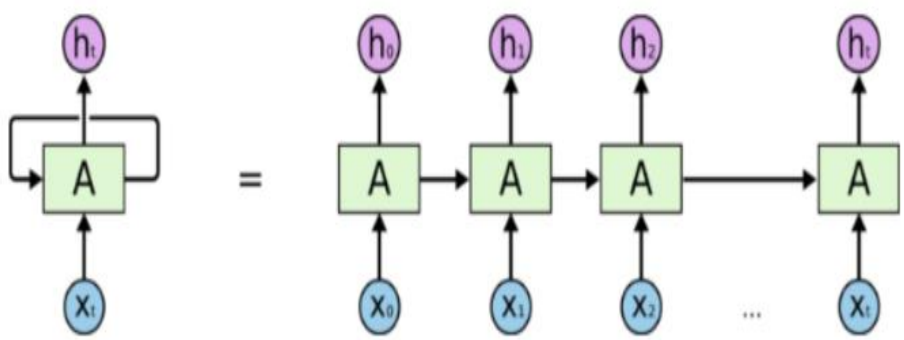


Figure 4.8 Architecture of RNN

4.1.3 Long Short-Term Memory (LSTM)

RNN's have troubles about the short-term memory. If a sequence is long enough, they have a hard time carrying information from earlier time steps to later ones. So, if you are trying to process a paragraph of text to do predictions, RNN's may leave out important information from the beginning. Therefore, these causes the need of Long Short-Term Memory (LSTM) which is a special kind of RNN's, capable of learning long-term dependencies. LSTM's have skills to remember the information for a long period of time.

All recurrent neural networks have the form of a chain of repeating modules of the neural networks. In standard RNNs, this repeating module will have a very simple structure, such as a single tanh layer as shown in figure 4.9.

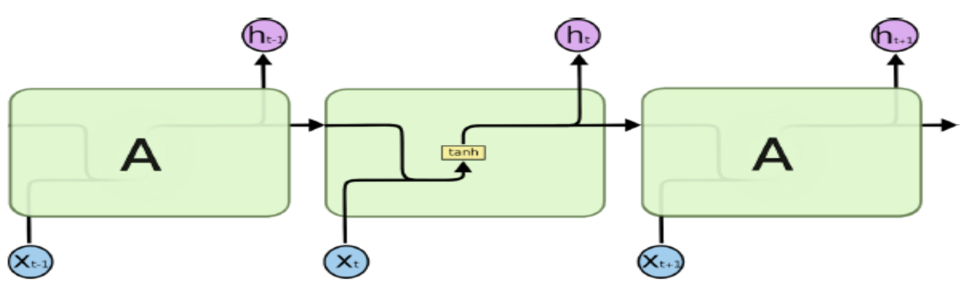


Figure 4.9 The repeating module in an RNN contains four

LSTMs also have this chain-like structure, but the repeating module has a different structure. Instead of having a single neural network layer, there are four, interacting in a very special way as shown in figure 4.10.

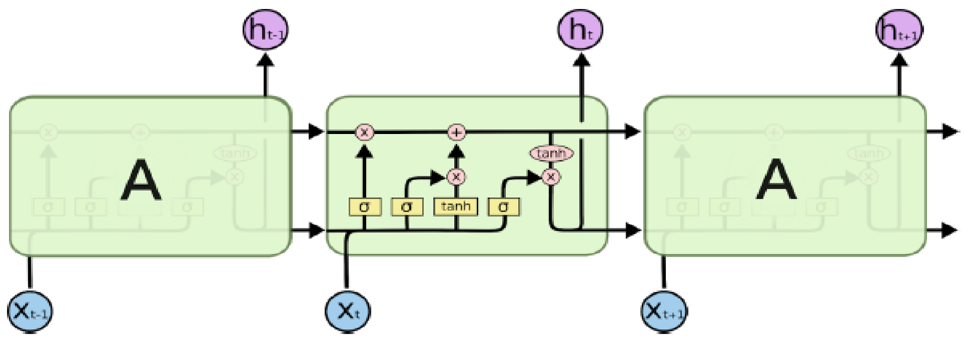


Figure 4 10 The repeating module in an LSTM contains four

The core concept of LSTM’s is the cell state, and its various gates. The cell state act as a transport highway that transfers relative information all the way down the sequence chain. You can think of it as the “memory” of the network. The cell state, in theory, can carry relevant information throughout the processing of the sequence. So even information from the earlier time steps can make its way to later time steps, reducing the effects of short-term memory. As the cell state goes on its journey, information gets added or removed to the cell state via gates. The gates are different neural networks that decide which information is allowed on the cell state. The gates can learn what information is relevant to keep or forget during training.

Let's dig a little deeper into what the various gates are doing, shall we? So, we have three different gates that regulate information flow in an LSTM cell. A forget gate, input gate, and output gate. Don't worry about the details of what's going on. We'll walk through the LSTM diagram step by step later. For now, let's just try to get comfortable with the notation we'll be using.

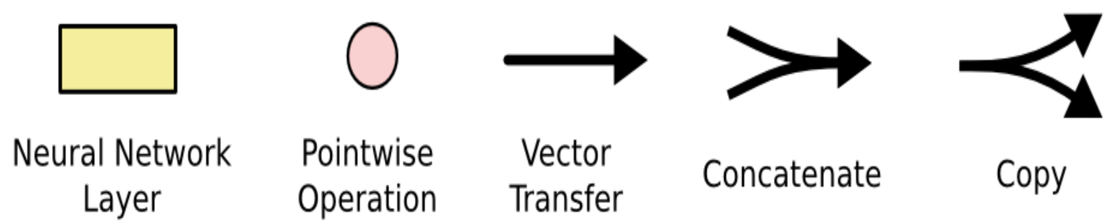


Figure 4.11 Different types of notations in LSTM

As shown in figure 4.11, each line carries an entire vector, from the output of one node to the inputs of others. The pink circles represent pointwise operations, like vector addition, while the yellow boxes are learned neural network layers. Lines merging denote concatenation, while a line forking denote its content being copied and the copies going to different locations.

LSTM step-by-step:

1. Forget Gate

First, we have the forget gate. This gate decides what information should be thrown away or kept. Information from the previous hidden state and information from the current input is passed through the sigmoid function. Values come out between 0 and 1. The closer to 0 means to forget, and the closer to 1 means to keep.

2. Input Gate

To update the cell state, we have the input gate. First, we pass the previous hidden state and current input into a sigmoid function. That decides which values will be updated by transforming the values to be between 0 and 1. 0 means not important, and 1 means important. You also pass the hidden state and current input into the tanh function to squish values between -1 and 1 to help regulate the network. Then you multiply the tanh output with the sigmoid output. The sigmoid output will decide which information is important to keep from the tanh output.

3. Cell State

Now we should have enough information to calculate the cell state. First, the cell state gets pointwise multiplied by the forget vector. This has a possibility of dropping values in the cell state if it gets multiplied by values near 0. Then we take the output from the input gate and do a pointwise addition which updates the cell state to new values that the neural network finds relevant. That gives us our new cell state.

4. Output Gate

Last, we have the output gate. The output gate decides what the next hidden state should be. Remember that the hidden state contains information on previous inputs. The hidden state is also used for predictions. First, we pass the previous hidden state and the current input into a sigmoid function. Then we pass the newly modified cell state to the tanh function. We multiply the tanh output with the sigmoid output to decide what information the hidden state should carry. The output is the hidden state. The new cell state and the new hidden is then carried over to the next time step.

To review, the Forget gate decides what is relevant to keep from prior steps. The input gate decides what information is relevant to add from the current step. The output gate determines what the next hidden state should be.

4.1.4 Convolutional Neural Network (CNN)

CNN were initially developed in the neural network image processing community where they achieved break-through results in recognizing an object from a pre-defined category (e.g., cat, bicycle, etc.).

A Convolutional Neural Network typically involves two operations, which can be thought of as feature extractors: **convolution** and **pooling**.

The output of this sequence of operations is then typically connected to a fully connected layer which is in principle the same as the traditional multi-layer perceptron neural network (MLP).

Let us first understand the term neural networks. In a neural network, where neurons are fed inputs which then neurons consider the weighted sum over them and pass it by an activation function and passes out the output to next neuron as shown in figure 4.12.

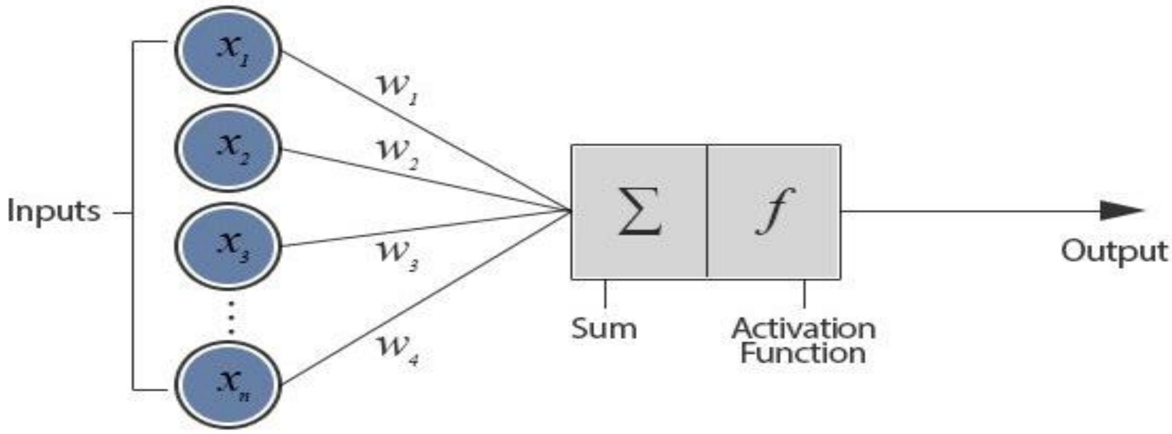


Figure 4. 12 Neural network understanding

Now, a convolutional neural network is different from that of a neural network because it operates over a volume of inputs.

Each layer tries to find a pattern or useful information of the data. An example of multi-channel input is that of an image where the pixels are the input vector and RGB are the 3 input channels representing channel as shown in figure 4.13.

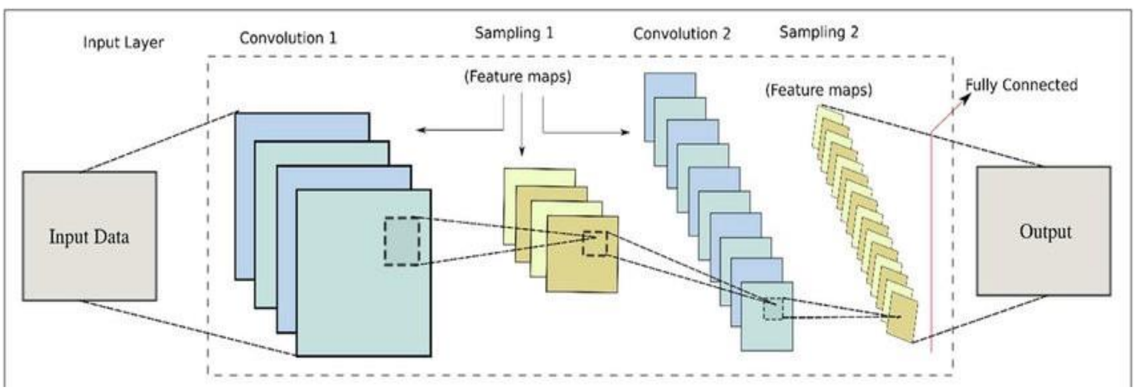


Figure 4. 13 Feature extraction and classification

This is what the architecture of a CNN normally looks like. It will be different depending on the task and dataset we work on. There are some terms in the architecture of a convolutional neural networks that we need to understand before proceeding with our task of text classification.

Convolution: It is a mathematical combination of two relationships to produce a third relationship. Joins two sets of information.

Convolution over input: We slide over input data the convolution to extract features by applying a filter/ kernel (both can be used interchangeably). This is important in feature extraction. There are some parameters associated with that sliding filter like how much input to take at once and by what extent should input be overlapped.

- **Stride**: Size of the step filter moves every instance of time.
- **Filter count**: Number of filters we want to use.

When we are done applying the filter over input and have generated multiple feature maps, an activation function is passed over the output to provide a non-linear relationship for our output. Figure 4.14 Is an example of activation function can be ReLu

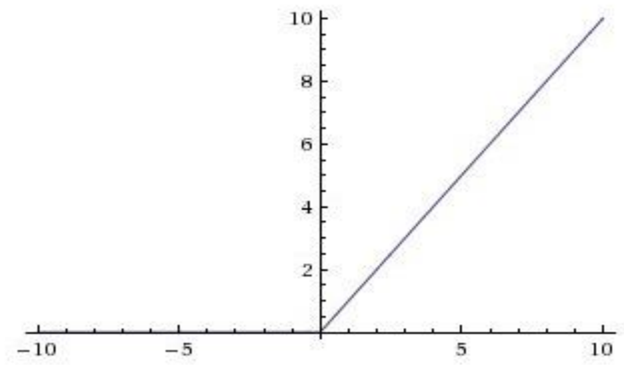


Figure 4.14 Relu activation function

Now, we generally add padding as shown in figure 4.15 surrounding input so that feature map does not shrink. If we don't add padding, then those feature maps which will be over number of input elements will start shrinking and the useful information over the boundaries start getting lost.

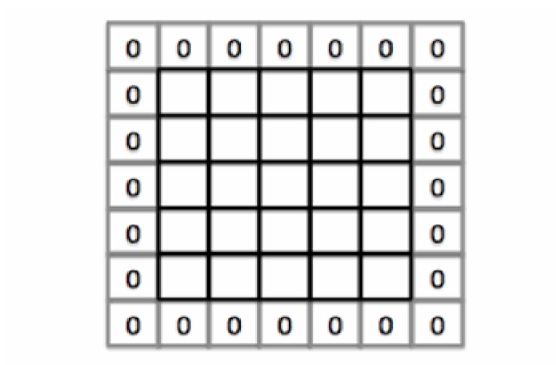


Figure 4.15 Padding

It also improves the performance by making sure that filter size and stride fits in the input well.

We are not done yet. We need something that helps us to reduce this high computation in the CNN and not overfit the data. Overfitting will lead the model to memorize the training data rather than learning from it.

Pooling

We use a pooling layer in between the convolutional layers that reduces the dimensional complexity and keeps the significant information of the convolutions. One example is of max pooling layer. It finds the maximum of the pool and sends it to the next layer as we can see in the figure 4.16.

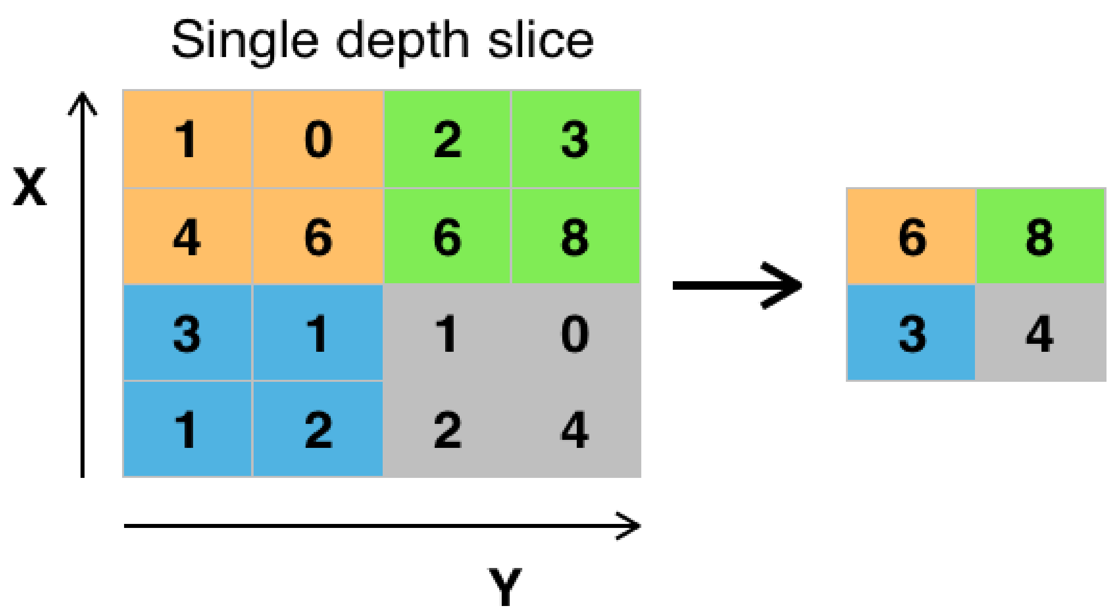


Figure 4.16 Max pooling

Sometimes a Flatten layer is used to convert 3-D data into 1-D vector. In a CNN, the last layers are fully connected layers i.e., each node of one layer is connected to each node of the other layer.

Fully Connected

The two processes described before i.e.: convolutions and pooling, can been thought of as a feature extractor, then we pass these features, usually as a reshaped vector of one row, further to the network, for instance, a multi-layer perceptron to be trained for classification.

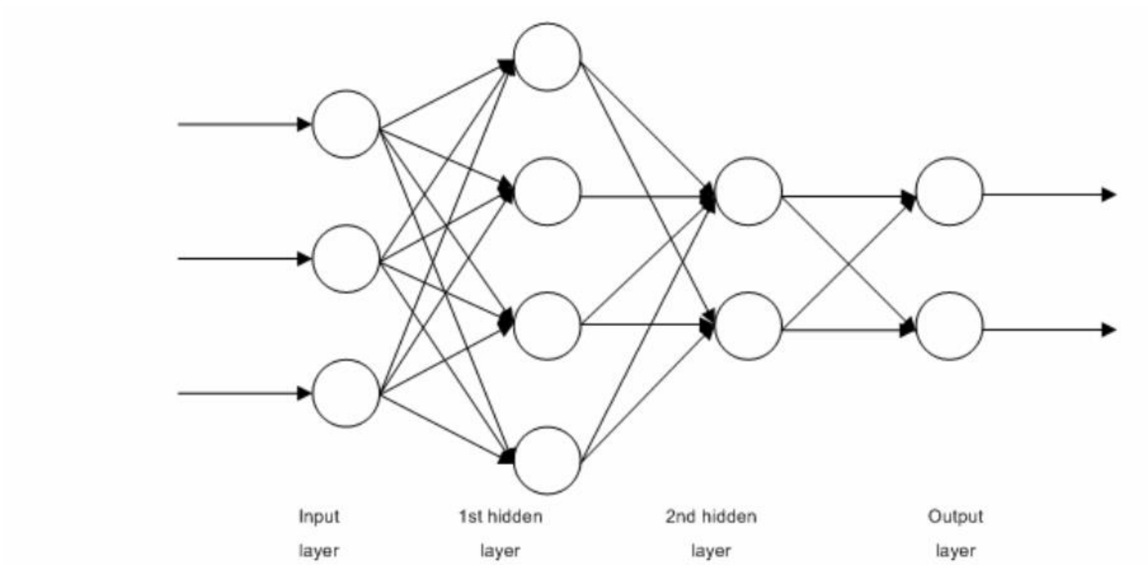


Figure 4.17 Example of multi-layer perceptron network

CNN with text classification

1- Text Data Preprocessing:

Text Preprocessing mainly includes operations such as stop word removal, normalization, stemming and lowercasing. We apply on our training data this preprocessing to use as the input of CNN text classification model. **[16]**

2- CNN Text Classification Model

Word embedding in the CNN-based text classification model is a method of converting words in the text into digital vectors. In orders to use standard machine learning algorithm to analyze them, it is necessary to take these vectors converted into numbers as input in digital form. The structure of the CNN text classification model is generally as shown in figure 4.31**.** In order to store word embeddings, we need a V*D matrix, where V is the size of vocabulary, and D is the dimension of word embedding. The dimension of word embedding is a user-defined hyperparameter. The larger D is stronger the expressive ability of word embedding. In the model, the matrix is called embedding layer. According to the text length and the classification performance indicators, multiple convolutional layers and pooling layers can be adopted to build the model. In general, for short text with a length of less than 50 words, we recommend using one convolutional pooling layer. Two convolutional pooling layers can be adopted for text with a length of 500 words or less. The structure and hyperparameters of the model can be determined with the deep learning model training method. The convolutional pooling layer is generally followed by a densely connected layer. Among them, common general pooling methods include max-pooling and average pooling. In our approach max pooling method is used to achieve the effect of obtaining the maximum value, in each feature graph by the feature graph unit before the classification layer through global maximum pooling. Finally, the classification layer is connected according to the actual classification number, which can be two-class or multi-class. **[15]**

4.2 Implementation

4.2.1 Preprocessing

1-Removing stop words

First let's know what are stop words? Stop words are the words in any language which does not add much meaning to a sentence. They can safely be ignored without sacrificing the meaning of the sentence.

EX: "This is a sample sentence, showing off the stop words filtration". After removing stop words "This sample sentence, showing stop words filtration".

2- Converting to lowercase

Is one of the simplest and most effective form of text preprocessing. Why it's important? You might treat a word which is in the beginning of a sentence with a capital letter different from the same word which appears later in the sentence but without any capital latter. This might lead to decline in the accuracy.

EX: "This is a Sample Sentence, Showing of the Stop WORDS Filtration". After converting to lowercase letters "this is a sample sentence, showing of the stop words filtration".

3- Normalization

Is the process of transforming a text into a canonical (standard) form. For example, the word "kinda" can be transformed to "kind of".

4- Stemming

Stemming is the process of removing a part of a word or reducing a word to its stem or root. For example, these words “playing”,“plays”, “played “ after applying stemming ,these words have the same root “play”.

5- Noise removal

Is one of the most essential text preprocessing steps, it’s removing special characters, punctuation from the text. For example, Convert “<corona-virus>?” to “corona virus”.

6- Removing emojis

Is the process of Removing emoji from sentence. For example, Converting “covid-19 not real 👎 ”to “covid-19 not real”

4.2.2 Feature extraction

- Embedding

  Is the collective name for a set of language modeling and feature learning techniques in natural language processing (NLP) where words or phrases from the vocabulary are mapped to vectors of real Numbers.

- Doc2vec

  Doc2Vec is a model developed in 2014 based on the existing Word2Vec model, which generates vector representations for words. Word2Vec represents documents by combining the vectors of the individual words, but in doing so it loses all word order information. Doc2Vec expands on Word2Vec by adding a" document vector" to the output representation, which contains some information about the document as a whole, and allows the model to learn some information about word order. Preservation of word order information makes Doc2Vec useful for our application, as we are aiming to detect subtle differences between text documents.

4.2.3 Dataset

We used COVID DATASET **[16]** that consist of 12,000 records of news and hydrate tweets, almost equally distributed to the true and fake categories. Our dataset is collected from Zenodo Website that have a lot of fake news. The datasets comprise the text of news/twitter and label. A class label with values '0' for true news and '1' for fake news.

One of the most important features of this dataset is that it contains many words, which feeds our tokenizer in collecting many words to give each of them its own number. The datasets were selected based on their complementary attributes in terms of text types (long news with average length of about 100 words vs. twitter posts with an average length of just under 20 words) **[18]**

4.2.4 Evaluation Metrics

Accuracy: is the ratio of number of correct predictions to the total number of input samples as shown in eq1

$$Accuracy = \frac{True\ Positive + True\ Negative}{TotalSample} \qquad (1)$$

Recall: the ability of a classification model to identify all relevant as shown in eq2.

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \qquad (2)$$

Precision: the ability of a classification model to return only relevant instances as shown eq3.

$$\text{precision} = \frac{True\ Positive}{True\ Positive + False\ Positive} \qquad (3)$$

4.2.5 Experimental Results

Table 2 The performance of the proposed approach

Model	Accuracy	Recall	Precision
SVM	75.2%	56.2%	75.75%
CNN	89.55%	89.67%	84.41%
LSTM	94.65%	92.13%	93.99%

The proposed approach was tested using three models: SVM, CNN, and LSTM. The accuracy, recall, and precision factors of each model were measured to assess performance. As shown in Table 2, it is clear that LSTM outperformed all the other models. the accuracy for LSTM was up to 94.65%, CNN achieved 89.55%, while SVM had the worst performance with an accuracy of 75.2%.

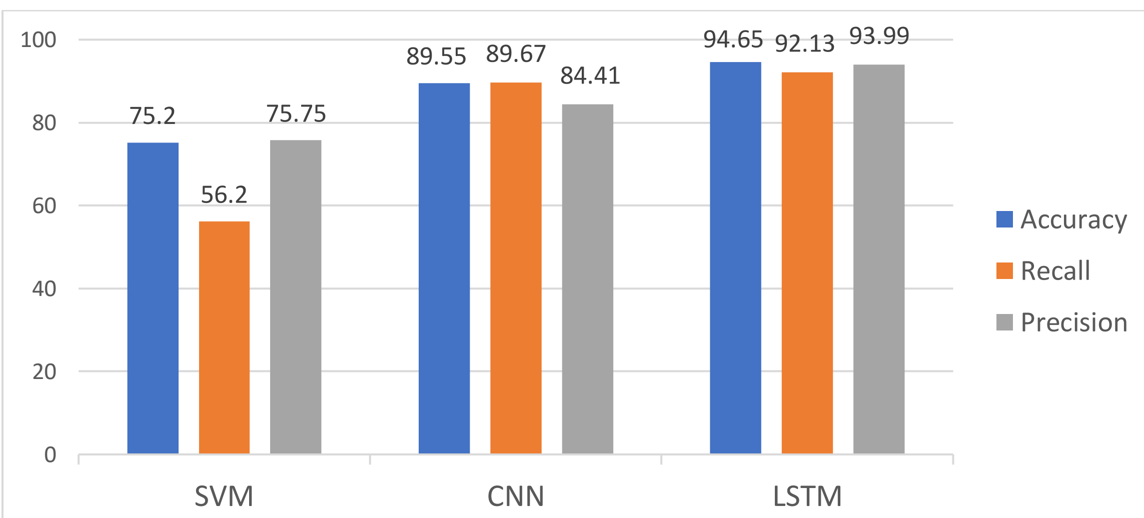


Figure 4.18 The performance of the proposed approach

4.2.6 Confusion Matrices of Models

- **SVM**

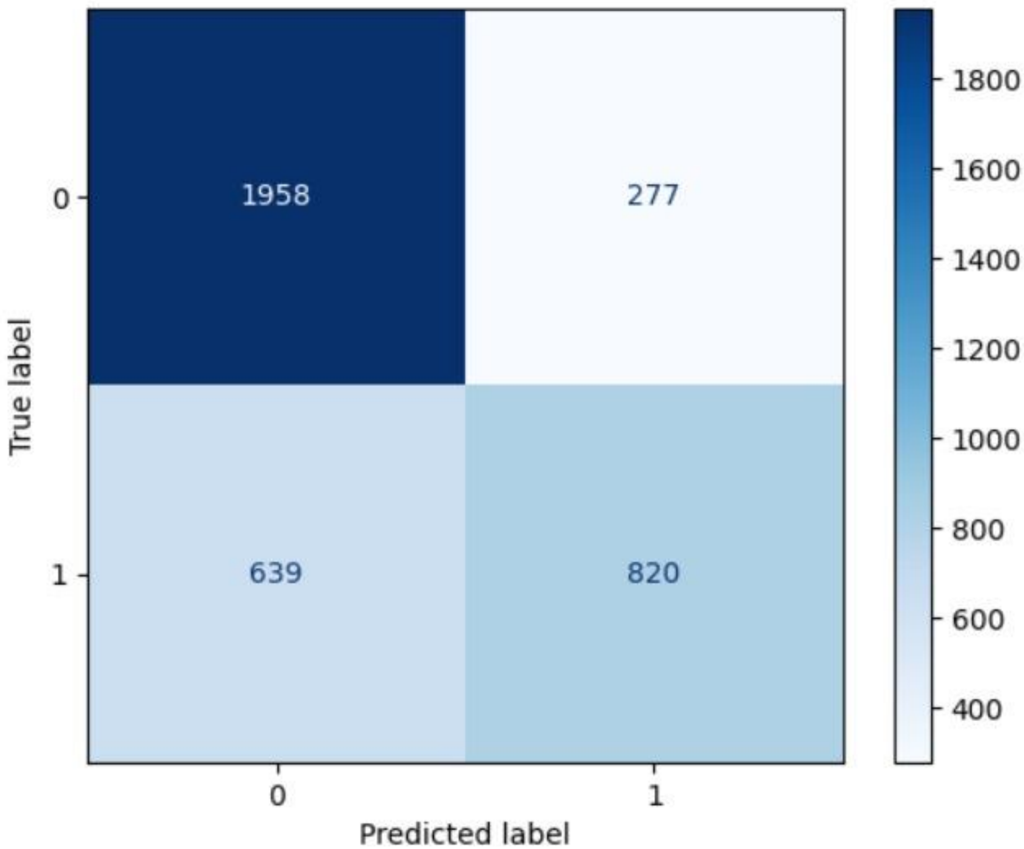


Figure 4.19 Confusion Matrix of SVM

- **CNN**

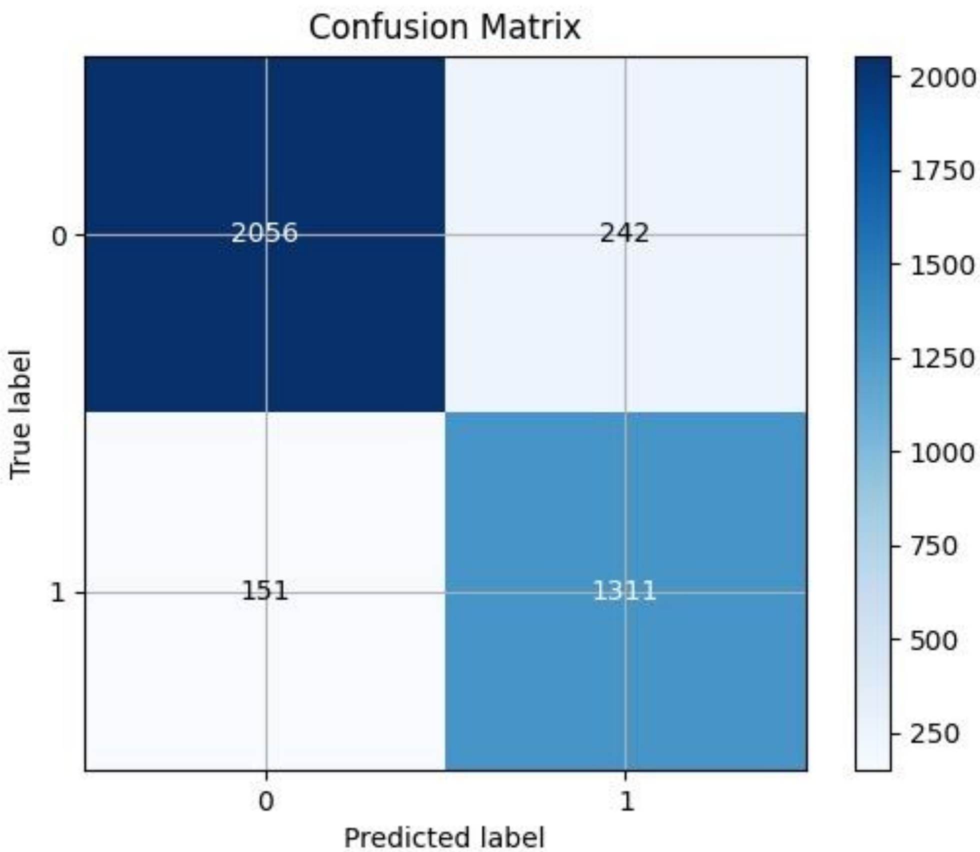


Figure 4.20 Confusion Matrix of CNN

- **LSTM**

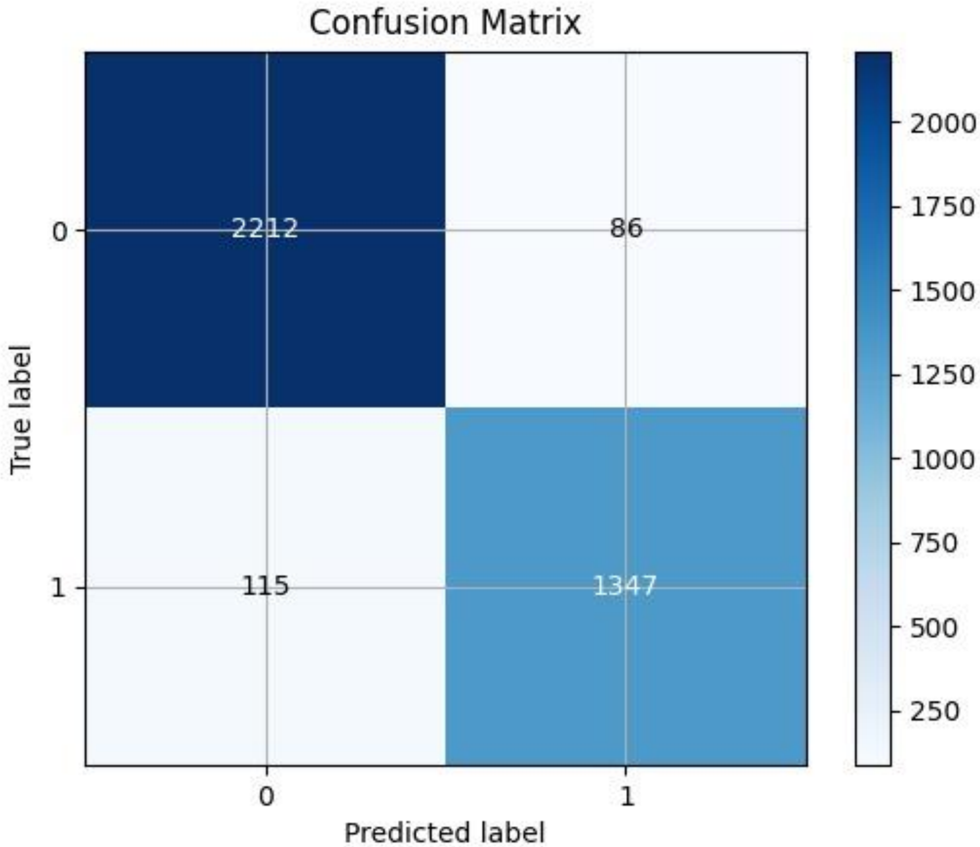


Figure 4.21 Confusion Matrix of LSTM

Chapter 5

User Manual

5.1 Installation Guide

1. Install Python version 3.9.4

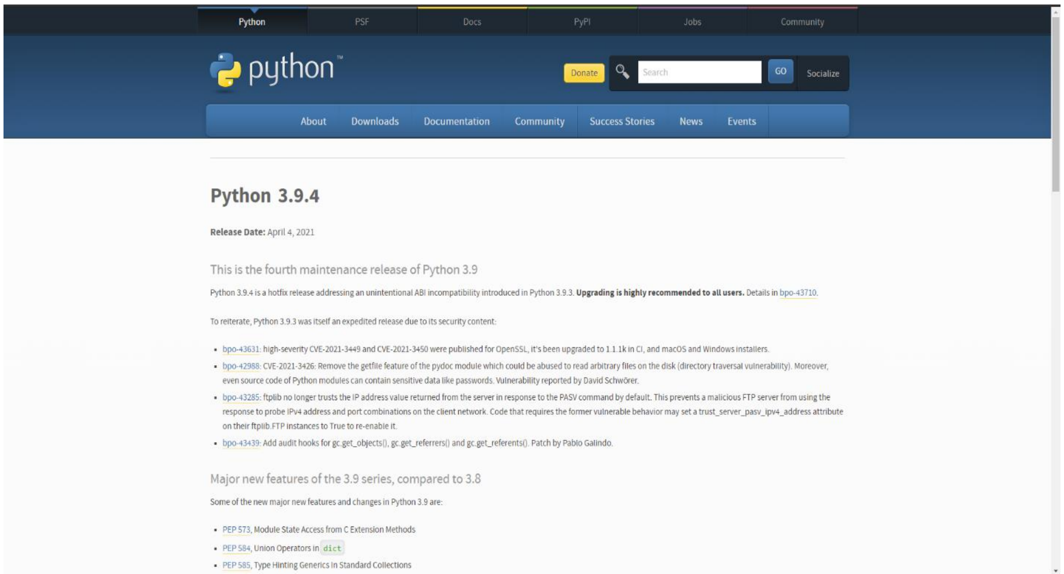


Figure 5.1 Install Python

2. Choose the appropriate version for your operating system

Files

Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Source release		cc8507b3799ed4d8baa7534cd8d5b35f	25411523	SIG
XZ compressed source tarball	Source release		2a3dba5fc75b695c45cf1806156e1a97	18900304	SIG
macOS 64-bit Intel installer	Mac OS X	for macOS 10.9 and later	2b974bfd787f941fb8f80b5b8084e569	29866341	SIG
macOS 64-bit universal2 installer	Mac OS X	for macOS 10.9 and later, including macOS 11 Big Sur on Apple Silicon (experimental)	9aa68872b9582c6c71151d5dd4f5ebca	37648771	SIG
Windows embeddable package (32-bit)	Windows		b4bd8ec0891891158000c6844222014d	7580762	SIG
Windows embeddable package (64-bit)	Windows		5c34eb7e79cfe8a92bf56b5168a459f4	8419530	SIG
Windows help file	Windows		aaacfe224768b5e4aa7583c12af68fb0	8859759	SIG
Windows installer (32-bit)	Windows		b790fdaff648f757bf0f233e4d05c053	27222976	SIG
Windows installer (64-bit)	Windows	Recommended	ebc65aaa142b1d6de450ce241c50e61c	28323440	SIG

Figure 5.2 Python versions

3. Install Visual Studio Code or Pycharm

   "we recommend install Visual Studio Code in order to avoid any bugs from Pycharm Versions"

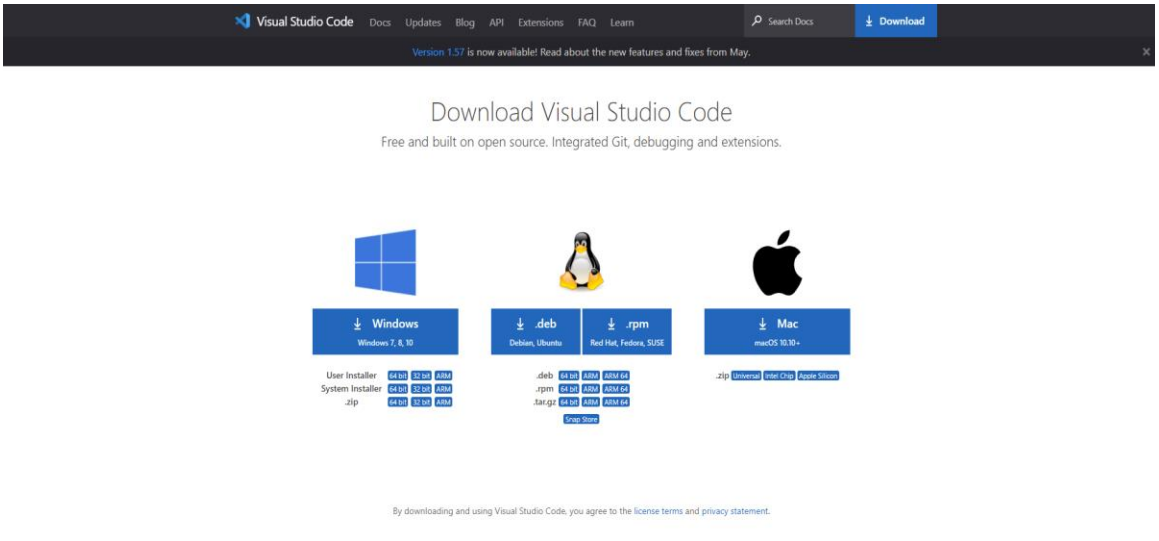


Figure 5.3 Install visual studio code

4. Download the project from this "https://github.com/Rana-Mostafa/Fake-news-and-COVID-19.git "

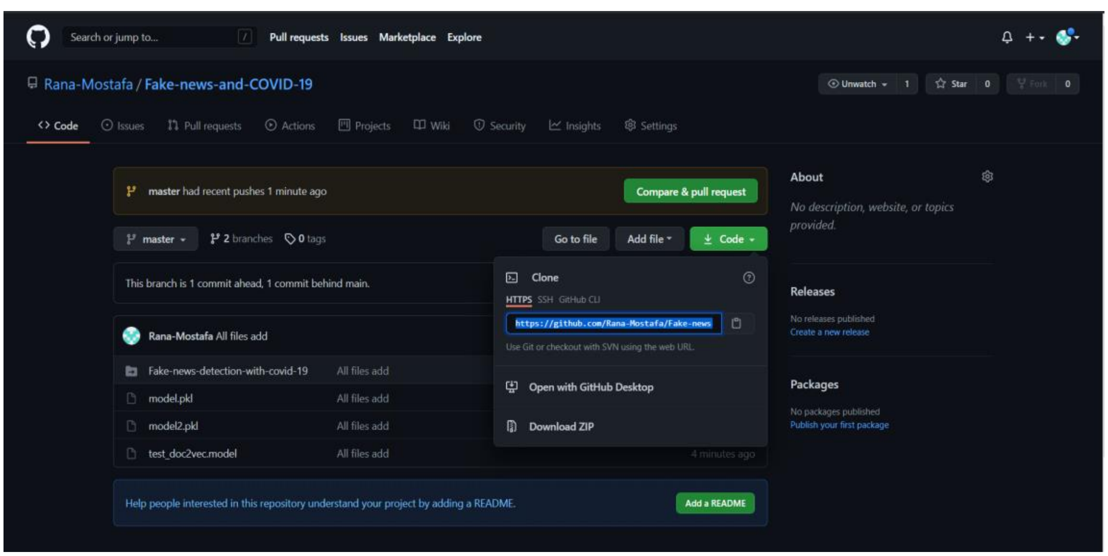


Figure 5.4 Project download

5. After you have opened the project

   5.1. If you face any problem with the interface design, you should check for the “css” files by going for that path : “Templates => home.html”& “Templates => advance_search.html”

      5.1.1. try to change the paths of the (.css files) to your own path which you have download the project at:

```
<!-- CSS  -->
<link href="https://fonts.googleapis.com/icon?family=Material+Icons" rel="stylesheet">
<link href="..\static\css\materialize.css" type="text/css" rel="stylesheet" media="screen,projection"/>
<link href="..\static\css\style.css" type="text/css" rel="stylesheet" media="screen,projection"/>
```

Figure 5.5 Change path steps

   5.2. Now before you start to run the project “install these commands in the terminal section”

Pip install keras

Pip install numpy

Pip install tensorflow

Pip install pandas

Pip install pickle

Pip install flask

Pip install sklearn

Pip install nltk

Figure 5.6 CMD console steps

5.3. You will need to train the model just once, in order to save time each time you open and run the project

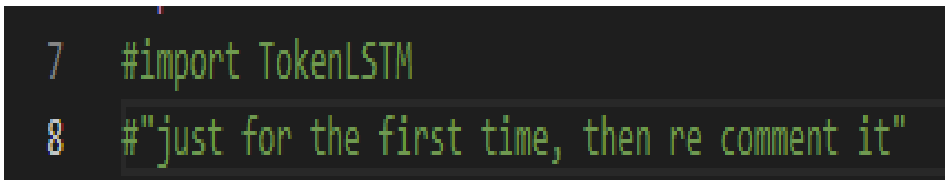


Figure 5.7 Import model step

6. Now you are ready to run the full project

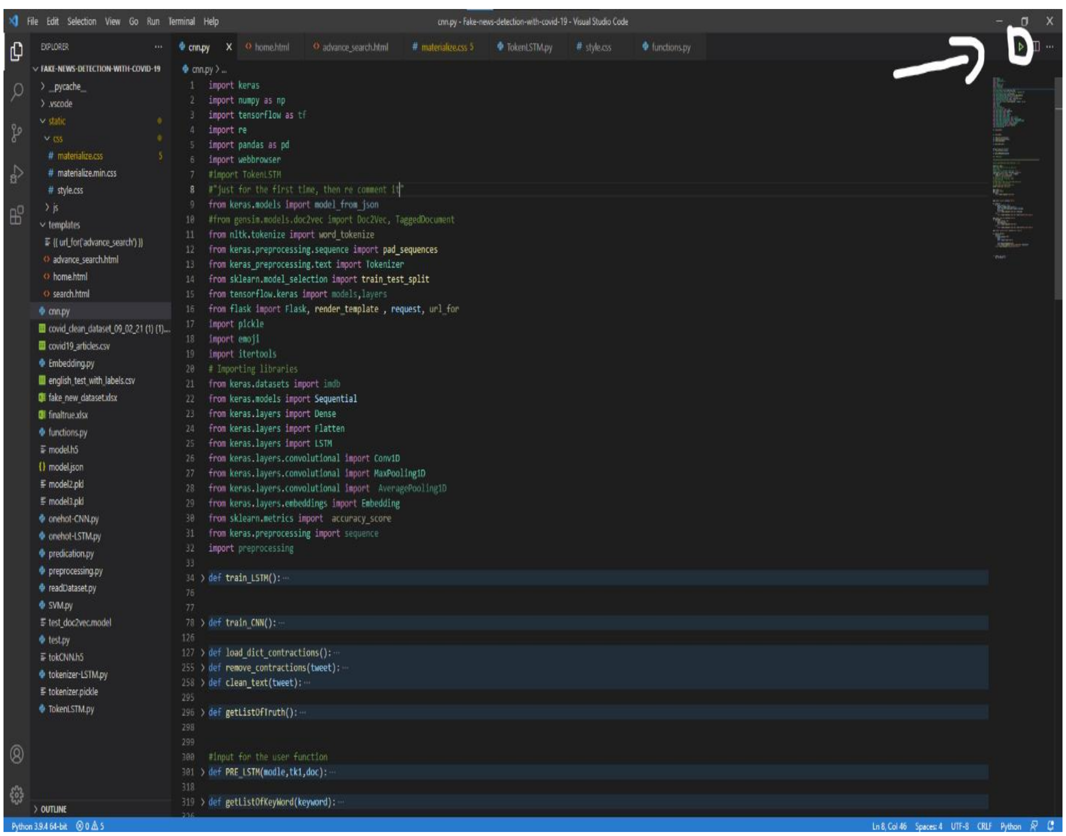


Figure 5.8 Run project step

7. You can try to open the website http://127.0.0.1:5000/ from here

    7.1. or you just click "ctrl+click" here like below from terminal

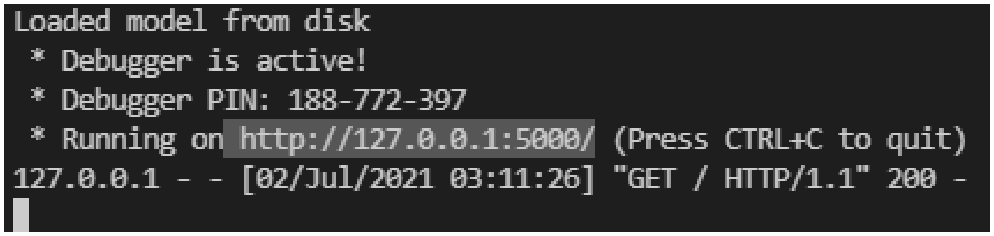


Figure 5.9 Project run

8. First is the homepage you are going to see with different functionalities''

Figure 5.10 Show front-end

9. First thing you will notice is the “news” area that has the main purpose of our project

    9.1. You can either paste any news you saw from the source like this

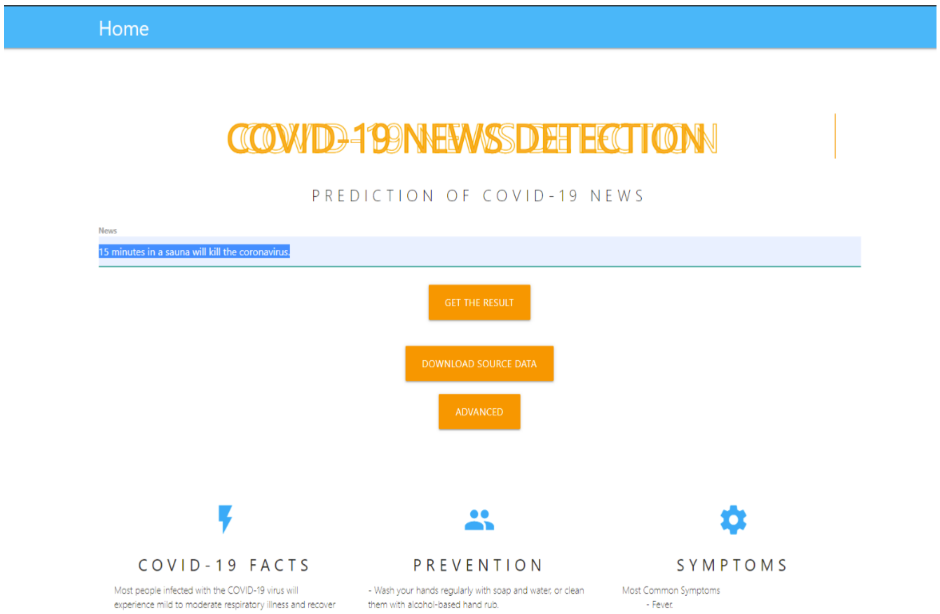
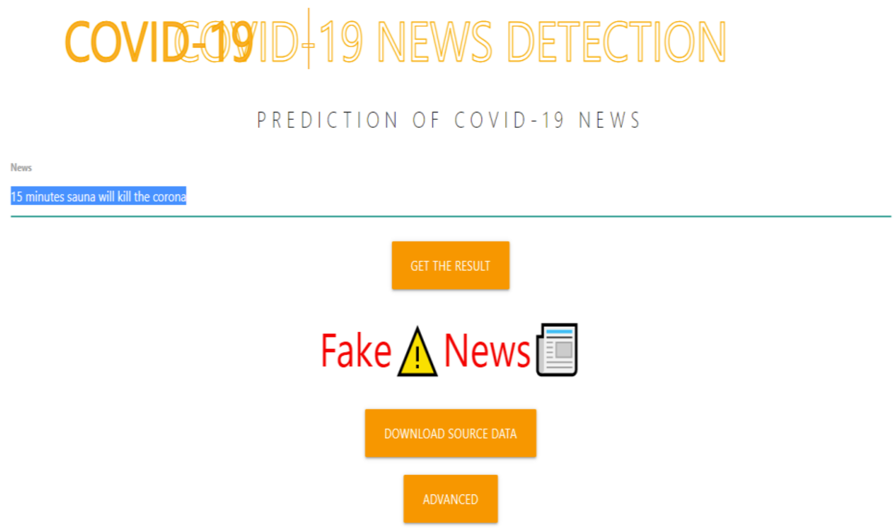


Figure 5.11 Demo screenshot (User input)

Then click on “Get the Result” button

Figure 5.12 Demo screenshot (Fake prediction result

9.2. Or you can write the news manually, no need for (copy&paste) with different form you want to write with "will get the same result !"



9.2.1. Another sample test for different case

Figure 5.13 Demo screenshot (Real prediction result)

10. The second button is an extra feature, make it possible to the user to download the data for e.g. (studying purposes)

Figure 5.14 Demo screenshot (Download data)

10.1. And it is available for downloading

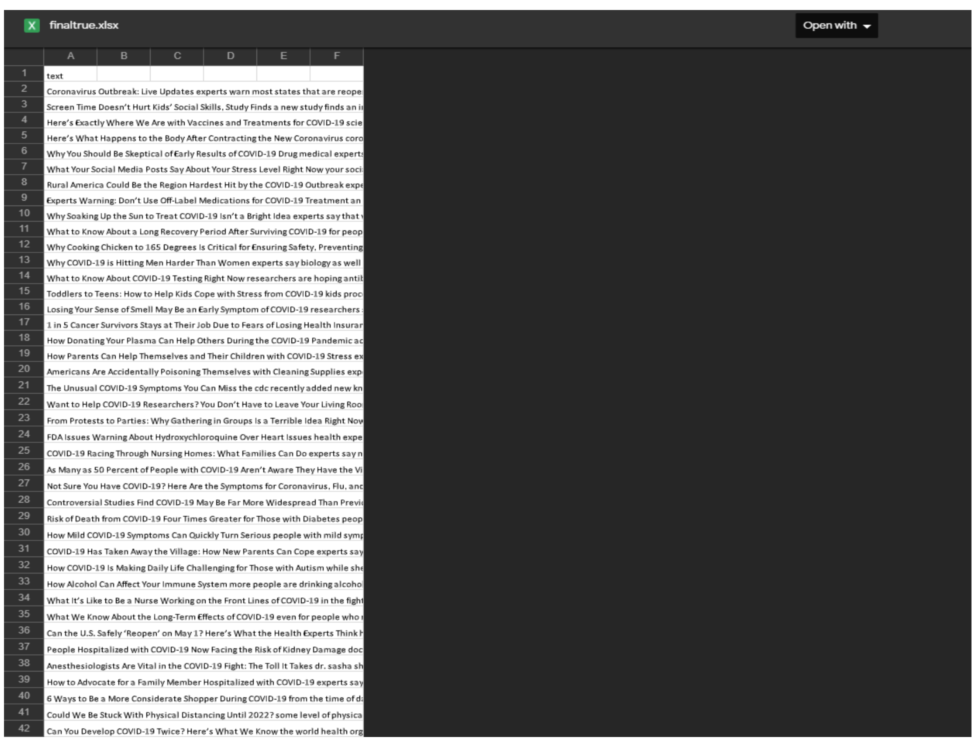


Figure 5.15 Demo screenshot (Download data 2)

11. The third button “Advanced” => advanced search to the user if he wants to know the records that came up with a certain “Word” he is searching about it

Figure 5.16 Demo screenshot (Advanced search)

11.1. Write any word that you want to search about, the hit “Enter” on your keyboard Or you can just click on “Search” button…

The result

Figure 5.17 Demo screenshot (Advanced search result)

12. Finally, if you need to go to the homepage just click on “Home” tap upper-left

Figure 5.18 Demo screenshot (Return to home)

Chapter 6

Conclusion and Future Work

6.1 Conclusions

The ongoing COVID-19 pandemic is a threat to human beings. Unlike other global challenges, such as global warming, containing and defeating COVID-19 will depend much on the quality and credibility of information shared amongst people. In this work, we suggested an approach for detecting fake news of COVID-19 in social media using natural language processing and machine learning techniques, A large number of fake and real news were collected in order to train the examined models.

The performance of the proposed approach is assessed while using three different evaluation metrics. Three different models are examined which are SVM, CNN, and LSTM. The performance of the LSTM was the best out of the three modules. The proposed approach obtained an accuracy of 94.65%, a precision of 93.99% and a recall of 92.13% in detecting

fake news of COVID-19.

6.2 Future Work

We acknowledge that there is room for improvement, part of which shall include the following.

- First, increasing the size of the dataset by adding more news, whether fake or real, and through it we will have a very large dictionary of words that we can use it to predict with better accuracy.
- Second, use other techniques than that we used in the prediction process.
- Finally, rely on other forms than text only. For example, the user may only publish an image, and this image may contain false news, so we hope to use some NLP to extract the text from within the image and then enter it on our model.

References

[1] Kaliyar, R.K., Goswami, A. and Narang, P., 2021. FakeBERT: Fake news detection in social media with a BERT-based deep learning approach. *Multimedia Tools and Applications*, *80*(8), pp.11765-11788.

[2] Ράπτη, M., 2019. Fake News in the era of online intentional misinformation; a review of existing approaches.

[3] Shu, K., Wang, S., Lee, D. and Liu, H., 2020. Mining disinformation and fake news: Concepts, methods, and recent advancements. ArXiv, 1-18.

[4] Umer, M., Imtiaz, Z., Ullah, S., Mehmood, A., Choi, G.S. and On, B.W., 2020. Fake news stance detection using deep learning architecture (CNN-LSTM). *IEEE Access*, *8*, pp.156695-156706.

[3] Patwa, P., Sharma, S., Pykl, S., Guptha, V., Kumari, G., Akhtar, M.S., Ekbal, A., Das, A. and Chakraborty, T., 2020. Fighting an infodemic: Covid-19 fake news dataset. *arXiv preprint arXiv:2011.03327*.

[4] *Abdelminaam, D.S., Ismail, F.H., Taha, M., Taha, A., Houssein, E.H. and Nabil, A., 2021. Coaid-deep: An optimized intelligent framework for automated detecting covid-19 misleading information on twitter. IEEE Access, 9, pp.27840-27867.*

[5] Al-Rakhami, M.S. and Al-Amri, A.M., 2020. Lies kill, facts save: detecting COVID-19 misinformation in twitter. Ieee Access, 8, pp.155961-155970.

[6] Elhadad, M.K., Li, K.F. and Gebali, F., 2020, August. COVID-19-FAKES: A Twitter (Arabic/English) dataset for detecting misleading information on COVID-19. In International Conference on Intelligent Networking and Collaborative Systems (pp. 256-268). Springer, Cham.

[7] Gundapu, S. and Mamidi, R., 2021. Transformer based Automatic COVID-19 Fake News Detection System. arXiv preprint arXiv:2101.00180.

[8] Li, X., Xia, Y., Long, X., Li, Z. and Li, S., 2021. Exploring text-transformers in aaai 2021 shared task: Covid-19 fake news detection in english. arXiv preprint arXiv:2101.02359.

[9] Hamid, A., Shiekh, N., Said, N., Ahmad, K., Gul, A., Hassan, L. and Al-Fuqaha, A., 2020. Fake news detection in social media using graph neural networks and NLP Techniques: A COVID-19 use-case. arXiv preprint arXiv:2012.07517.

[10] Gupta, A., Sukumaran, R., John, K. and Teki, S., 2021. Hostility detection and covid-19 fake news detection in social media. arXiv preprint arXiv:2101.05953.

[11] Raha, T., Indurthi, V., Upadhyaya, A., Kataria, J., Bommakanti, P., Keswani, V. and Varma, V., 2021. Identifying COVID-19 Fake News in Social Media. arXiv preprint arXiv:2101.11954.

[12] Glazkova, A., Glazkov, M. and Trifonov, T., 2020. g2tmn at Constraint@ AAAI2021: exploiting CT-BERT and ensembling learning for COVID-19 fake news detection. arXiv preprint arXiv:2012.11967.

[13] Cervantes, J., Garcia-Lamont, F., Rodríguez-Mazahua, L. and Lopez, A., 2020. A comprehensive survey on support vector machine classification: Applications, challenges and trends. Neurocomputing, 408, pp.189-215.

[14] Ce, P. and Tie, B., 2020. An Analysis Method for Interpretability of CNN Text Classification Model. Future Internet, 12(12), p.228.

[15] Banda, J.M.; Tekumalla, R.; Wang, G.; Yu, J.; Liu, T.; Ding, Y.; Chowell, G.;, 2021. A large-scale COVID-19 Twitter chatter dataset for open scientific research - an international collaboration. [Online] Available at: http://doi.org/10.5281/zenodo.4484079.

[16] Cui, L. and Lee, D., 2020. Coaid: Covid-19 healthcare misinformation dataset. arXiv preprint arXiv:2006.00885.