Hochschule Rhein-Waal
Rhine-Waal University of Applied Sciences
Faculty of Communication and Environment
Mrs. Prof. Dr. Margarita Spirova
Prof. Dr. Frank Zimmer

# Multi Languages Fake News Detection

Master Thesis

by
Md Monsur Ali

Hochschule Rhein-Waal
Rhine-Waal University of Applied Sciences
Faculty of Communication and Environment
Mrs. Prof. Dr. Margarita Spirova
Prof. Dr. Frank Zimmer

# Multi Languages Fake News Detection

A thesis submitted in
Partial fulfillment of the
Requirements for the degree of

Master of Science
in
Information Engineering and Computer Science

by
Md Monsur Ali
Matriculation Number: 24547

Submission Date:
17 October 2021

# Acknowledgement

A warm reception goes for me to acknowledge the faith in my actions as well as the person who encourages, appreciates, puts elegance to help me on the tasks of “Multi Languages Fake News Detection” that I have analyzed through my master’s thesis.

First of all, I want to take the opportunity to say thank Mrs. Prof. Dr. Margarita Spirova, my thesis’s principal supervisor, who has not only guided me through the process of choosing the appropriate topic but also provided his own valuable critiques. She has confronted me on several occasions as an approach to stimulate me too efficiently implements my exposure on this thesis and to conclude it promptly according to my plan. My special thanks go to Prof. Dr. Frank Zimmer, for his cordial trust and immediately agreeing to be my second supervisor.

Lastly, I might want to thank my family for their unending help and steady support over time of study and with the exploration and the most common way of writing this master’s thesis. It would not have been conceivable without them. This would not have been possible without them.

# Abstract

News is one type of information that has the potential to influence a large number of people. People have received news since the beginning of time through various birds, short letters, and other means. When newspapers were invented, information was available everywhere on paper. The news is no longer limited to paper-based platforms, thanks to the digitalization of online platforms. The online news platform is now available to read the news in a matter of seconds. As a result, news can easily connect people, and it is being used to spread fake news. Fake news is spread to gain attention for the wrong reasons. Different languages are used in our world to express our thoughts and feelings. There are specific materials for each language. The English language is the most studied topic when it comes to identifying fake news. Data and research resources are few in other languages, hence there is little research done. Of these, Bengali is one of the most widely spoken. Our ultimate goal is to create a tree that contains elements of both English and Bengali. Research on fake news and web scraping was used to get the language news data. Multilingual transformer models m-BERT and xlm-ROBERTa with long text or tokens are used to detect fake news (512 tokens or any token size). The two models were compared using two different datasets (with stop words and the other without) using three different fine-tuning freeze approaches (Freeze, No Freeze, and Freeze Embed). The results show that the dataset with stop words had a somewhat better performance than the dataset omitting stop words. The xlm-RoBERTa model outperforms the m-BERT model in terms of F1-score and accuracy.

Keywords: NLP, Fake News, Long text, m-BERT, Multilingual, xlm-RoBERTa

# List of Contents

# List of Acronyms

| | |
|---|---|
| AI | Artificial Intelligence |
| ANN | Artificial Neural Network |
| NLP | Natural Language Processing |
| NLI | Natural Language Inference |
| CNN | Convolutional Neural Network |
| RNN | Recurrent Neural Network |
| seq2seq | Sequence-to-Sequence |
| LSTM | Long Short-Term Memory |
| LD | Lexical Diversity |
| GRU | Gated Recurrent Unit |
| FF | Feed-Forward Layer |
| TF-IDF | Term Frequency-Inverse Document Frequency |
| HCI | Human Machine Interactions. |
| MLM | Masked Language Model |
| OSM | Online Social Media |
| PLM | Permutation Language Model |
| MTLD | Measure of Textual Lexical Diversity |
| NSP | Next Sentence Prediction |
| SOP | Sentence Order Prediction |
| SOTA | state-of-the-art |
| [MASK] | masking token |
| [UNK] | unknown token |
| [SEP] | separation token |
| [CLS] | classification token |
| GLUE | General Language Understanding Evaluation |

# List of Symbols

| | |
|---|---|
| $ | Dollar Sign |
| % | Percentage |
| √ | Root |
| √a | square root |
| f (x) | function of x |
| ∑ | sigma |

# List of Figures

# List of Tables

# 1

# Introduction

## 1.1 Motivation

Fake news, any false information intentionally completely and utterly false or misleading is spread through online social media, but sometimes seeks out the mainstream of the traditional print and broadcast media (Waweru, 2019). In the world, fake news or propaganda is used to mislead people. The difficult problem of controlling the fake news spreading remains same. So, it is a serious issue to understand the backend story of fake news. The method of getting the news is now changed and people generally consume information online now. Online media has turned into the most remarkable stage to get to news stories. Recent research has found that Facebook is one of the main wellsprings of information spending, particularly for the young aged people (Newman et al. 2019). The majority of social media users are younger. The younger generation is curious about any news and articles and it is easy to influence them. Other mainstream social media platforms include fake news sources such as WhatsApp, Twitter, Instagram, etc (Tandoc et al., 2017). This social media is used to spread false news[1] using clickbait headlines. The most interesting fact is that the user of these social media spread this news without knowing actual news or take any proper authentication. Harmful to the impact of fake news on American society.

As per a report, just 4% of adults accept local news can be valid and just 27% of adults accept national news is valid. The most leading fake news exposure country is turkey with 49% who say every previous week, people were faced entirely false news. Other countries like Mexico, Brazil, United States, and Germany with 43, 35, 31, and 9 percent respectively [Source: Forbes Statita (2018)]. The main idea is that emotions and human thoughts are skillfully believed to be things that are not true.

[1] See section 2.1 Fake News definition was described for this master's thesis.

The problem comes in a limited range that even the fake news cannot properly recognize by human eye. For example, evidence was found to show fake news, respondents discovered it to be "fairly 'or' exact 75% of the time", and one more followed that 80% of secondary grade understudies attempted to decide whether an article was fake (Ruchansky, 2017). To recognize the fact about fake news, several fact-checking websites (e.g., snopes.com, politicfact.com, jaachai.com) is developed where some are manually updating fake news stories published on online or offline platform with actual explanations and the reason behind the fake news. The website is critical in the propagation of false information. In response, different articles and online diaries have been made to build public mindfulness and give tips to recognizing truth from falsehood (McClure, 2017).

## 1.2 Problem Statement

The absence of fake news from the internet media is a blessing for the development of the digital world. Fake news is quite popular among readers. This may have an impact on the significance of serious news media. The problem is determining the trustworthiness of news and online content. Identifying the bots involved in the spread of false news is a critical concern as well.

Fake news has an impact on society, economy, and politics. Various unrealistic news sources divert society, and people react inappropriately to these unrealistic news sources. To make more money, the adviser creates eye-catching advertisements with false information. Politicians, on the other hand, influence the public in order to gain their support. It can sometimes go down the wrong path, resulting in violence among people. These are difficulties that the entire world is dealing with.

## 1.3 Methodology

The Bengali and English languages was used in this study report. Both the language dataset and past studies are available in Kaggle. Web scraping was used to acquire the additional dataset. The additional English language data came from the online newspapers **"The New York Times"** and **"The Guardian."** The Bengali data was gathered from earlier studies. The news was preprocessed independently in both languages. Both preprocessing data was combined for further analysis after preprocessing. The thesis came up with some ideas based on this information:

i. News data analysis and find pros and cons.
ii. Long token and output label handling approach
iii. Find the best fitted model for both languages Fake news.

This master's thesis assesses the two models m-BERT (Chang, 2019), Xlm-RoBERTa (Liu, 2019) which has recently been improved. Both models have learned a strong representation of the language in large corporations and can be made for the desired work. The main focus tuning is the determination of hyper parameters and general performance. Both models have multilingual advantages. Both models support more than 100 languages. Both models support 512 tokens, but this thesis attempted to implement token size approaches of any size. Further details about m-BERT chapter 6.2 BERT multilingual base model (uncased), xlm-Roberta model chapter 6.3 xlm-roberta-base, dataset details Chapter 5 Data Collection and Analysis and long token and output label handling chapter 6.5 Long Text Tokenizing.

## 1.4 Structure of the thesis

The thesis was structure by six different parts. Each part was also divided by its workflow. The structures are Data collection, Data preprocessing, Data Analysis, prepare the dataset for the model, fit the model and result. Below the details workflow graph are shown.

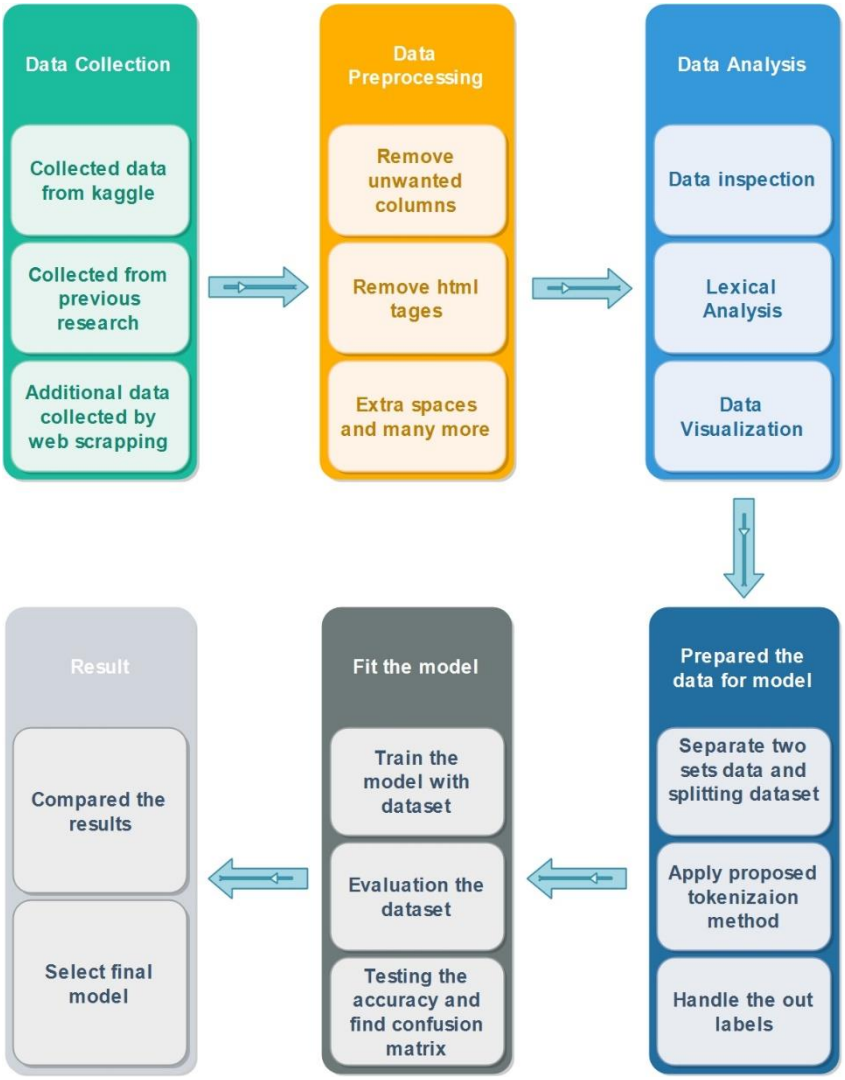



Figure 1: The thesis workflow structure (own illustrated)

In chapter 2, the thesis is summarized by first discussing the term Fake News and providing an overview of background history and related works to Fake News detection. The theoretical background chapter 3 was designed using machine learning natural, language processing, and model overview. In Chapter 4, it discussed the baseline model. Chapter 5 presents and explains the dataset collection and main pre-processing steps. The main chapter 6 begins with information on hyperparameter and finetuning, training model, long token and label handling approach, and evaluating metric. Chapters 7 and 8 went into great detail about the model's results and conclusion.

# 2

# Fake News Detection

## 2.1 Fake News

The term "fake news" refers to information that is false or deceptive but is disguised as genuine news. It undermines the actual reporting and propagation of the created media industry, creating it more credible for political purposes, mass-media, and residents (Qiao, n.d.). The term can also be used to express doubts about legitimate news from an opposition political point of view, known as false media. False information is often disturbing, dishonest, and directly fabricated, which is republished through social media. A broad definition of fake news can be false or misleading information published as truthful news, usually intentionally understood but perhaps accidentally. False news has no basis but has been presented as truthful (Waweru, 2019) The purpose of misinformation is to spread misleading information while ignoring the genuine facts. False information can be caused by incorrect labels, incorrect data checks, etc., and is effectively spread among readers who don't aware much about its authenticity whether they read or share (Nasir et al, 2021). In short, fake news embraces false, misleading, dubious, and media-driven stories that are widely circulated and corrupt or warm up the popular discourse (Qiao, n.d.).

## 2.2 Classes of Fake News

The distinct classes of fake news are as follows (Waweru, 2019):

i. Satire or parody – It's created for fun or to make people fool and no intention to cause harm to anyone.
ii. False connection – Mismatch. the headlines, main body, photo, and photo captions don't have a connection.
iii. Misleading content – Confusing information describe with details.
iv. False context – The real news is mixed with false contextual information.
v. Impostor content – When the actual source is false, the make-up is "disguised" with the source.
vi. Manipulated content - When actual information or imagery doctored "photos" like a used to deceive.
vii. Fabricated content - News content is designed to be 100% false, deceptive and harmful.

## 2.3 Details Historical Approach to Fake News

Fake news stories are not new to everyone. It is happening for several years. However, the process and techniques are different now. Disfigurement of realities, one-sided information, publicizing and data used to ruin convictions and qualities have consistently been important for society. What may be comparable in antiquity to a Roman time in 44 BC, even as a political move that is now called fake news, may be fulfilled as a political move. It happened when the rulers of Rome Mark Antony, fell in love with the Egyptian Queen Cleopatra. Octavian, the first emperor of ancient Rome, undertook a propaganda campaign against Antony to destroy his reputation. "Short, snappy phrases inscribed on coins in the vein of vintage Tweets" was one way to do it (Posetti et al, 2018). There is different proof throughout the long term that fake news may have consistently existed.

Sixteenth to twentieth centuries, every era was faced fake news propaganda. When Pietro Aretino saw a monument to a man named Pasquino outside the Piazza Navona in Rome in the 16th century, he attempted to influence the Pontifical election of 1522 by writing scathing sonnets about all the applicants and preventing them from receiving popular recognition. He explored wrong information and misrepresentation rumors to influence and dishonor people and politicians. Furthermore, in France, a newspaper named "Canard" sold the fake news of the French capital in the seventeenth century (Darnton, 2017).

In 1938, news radio became famous that was new at that time and listeners believed in live news broadcasting. News of Orson Welles' War of the Worlds was telecast and people were fright who believed the news. After a while, another news broadcast that previous news was false to capture people's reactions. On the other hand, yellow journalism was found that sending reporters abroad was expensive. So, the fake reporters created their own attractive stories to attract audiences. This circumstance is also happening today. Previous motivation is encouraging reporters to publish fake news and make their own benefit economically and financially.

The 21st century is dominated by Fake news too. In recent, fake news is viewed as probably the best danger to business, reporting, and nation everywhere in the world, with colossal accidental losses. The misfortune of 130 billion on the stock exchange was the immediate outcome of a false news responded that US President Barack Obama was damaged in an explosion (Nasir et al, 2021). Not only the business sector but also the political sector is popular with fake news and it is a normal issue to attack the opposition. The US political election in 2016 spread of purposeful false news did huge harm. The idea turned into an instrument or political weapon (Islam et al, 2020).

## 2.4 Related works

Several types of research have been developed to detect fake news. It's a Natural Language Processing (NLP) problem. Maximum NLP research has been done with the English language. So, other language resource is comparably low.

There are many varieties of analysis associated with detecting fake news like rumor detection and spam detection (Shen & al, 2017). From previous work, the definition of fake news is that an intentionally spreading miss-information news identified as (Ruchansky, 2017). Various algorithms try to detect fake news based on its characteristics, which can be drawn from the social background and content of the news. In the study of (Bal & al, 2019), The research by Arvinder et al. on seven machine learning methods for detecting false news performance.

The analysis from their research, Gradient Boosting with Ada boost has outperformed all other categories. They used a total of 18,962 false news pieces in their study, as well as 19,334 real news stories and 163 features gained from their categorization training. All these characteristics show that they have very high relevance scores in the target dataset.

Ahmed et al. (Ahmed & al, 2017) also analyzed the performance of six alternative algorithms that classify false and true information based on N-gram characteristics and TF-IDF scores. The dataset had product review comments and political news articles with the mixed of real and fake. They have developed a model that applies to two areas. Their research shows that using Unigram the most suitable for ranking news articles was found with TF and TF-IDF scores, and accuracy ranging from accurate to 89%. Although the results obtained in their research are promising and there are some good work paths, they only work on political articles, which limit the usefulness of their work to articles found during the 2016 US presidential election.

BanFakeNews research paper which is developed by Hossain et al. (Hossain et al, 2020) and published Bangla fake news dataset for further research. In that paper, they proposed two techniques to find Bengali fake news. The techniques were Human baseline and another traditional linguistic feature and neural network-based approaches. On Human baseline analysis, they hired five students to identify fake news between randomly mixed 60 fake and 90 real news. Concerning about news list, they also provided a questionnaire to answer. In addition, they analyze how to accurately detect human fake news. The F1-scores in the five-note counterfeit class were 58, 65, 70, 68 and 63 percent, respectively (Hossain et al, 2020). Creating the algorithm model, they used SVM, CNN, LSTM, and BERT (Courville, 2016) and the best result when incorporating all linguistic features with SVM. It scores 91 percentage F1-score. (Details Check page 67)

The multi-language fake news researches (Ermakova et al, 2019) are limited. Most of the research on automatic fake news detection is limited to documents in English, rather than comparing some tasks with other languages and evaluating language-independent characteristics.

A study on *"Multilingual Detection of Fake News and Vaccination Themed Satire"* studied three languages: US English, Brazilian Portuguese, and Spanish. They use online social media (OSM) to spot fake news. Online multilingual social media (OSM) have transformed the way news organizations written in American English, Brazilian Portuguese, and Spanish are written in order to better understand the hardships, stylistic, and psychological features of a text. Extracted features aid in the detection of fake, true, and humorous news. Detection model induced by the four machine learning algorithms (k-Nearest Neighbors (k-NN), Support Vector Machine (SVM), Mango Forest (RF), and Extreme Gradient Boosting (XGB) (Hastie, et al., 2009). The outcomes show that the defined language independence feature has been successful in three different languages, including false, ironic and legal news descriptions, with a mean radio frequency detection accuracy rate of 85.3%.

# 3

# Theoretical Background

## 3.1 Learning

'Learning' is a fundamental part of intelligence that helps our humans and animals adapt based on our past experiences. For example, in almost every situation, our decision is based on how we (previous output) have dealt with a similar function (previous information) and if it is effective (improvement). In other words, education is an estimate of actual output based on past experience. More succinctly as follows, (Knox, 2018) defines the problem of education as given below.

Definition 1 The problem of Learning:

"Suppose a known set of *x* and *f* is unknown functions of *x*. See the data, make a good guess *y* of *f* where it is called learning *f*."

The expected output is $y = f(x)$ for the problem and the target is $y \in f(x)$. $y$ is varied depending on the target range $f(x)$. If the range is finite then $Y$ is classification or $Y$ is regression if the range is a continuum. The probability distribution can be defined as data from the random draw $(x, y)$ from $x \times f(x)$. In the data visited $(x, y)$ the pair or only the $x$ may again separate the partitions as supervised and unsupervised, respectively. How good the approximate amount $f$ is compared with the $f$ id defined as the loss function.

## 3.2 Machine learning

Machine learning is a subfield of Artificial Intelligence (AI). Humans are researching, how computers or machines can learn by themselves or previous information and improve them by previous mistakes. For this, different types of algorithms and methods are using for solving the problems.

Creating algorithms to solve problems like learning from medical records, personal software as assistants, etc. So, it is helping humans in complex problems and situations.

## 3.3 Natural Language Processing

Natural Language Processing (NLP) is a branch of study that combines automated artificial intelligence computer algorithms with human (natural) language input procedures and human-machine interactions (HCI). It is also called "Computational Linguistics." (Kalyanathaya & Rajesh, 2019). The major NLP research on understanding the meaning of the text, translating from one language to another, creating and translating text, feeling of a subject, web search document segmentation, radiology report, etc. With the success of modern Artificial Intelligence and machine learning research, developers are building automated technology. Though NLP techniques are not easy and human connection is mandatory in every step to command them (Vedantam, et al., 2020). Natural language processing can be divided into two categories: "natural language understanding" and "natural language production," which refer to the duties of comprehending and producing text, respectively (Vedantam, et al., 2020).

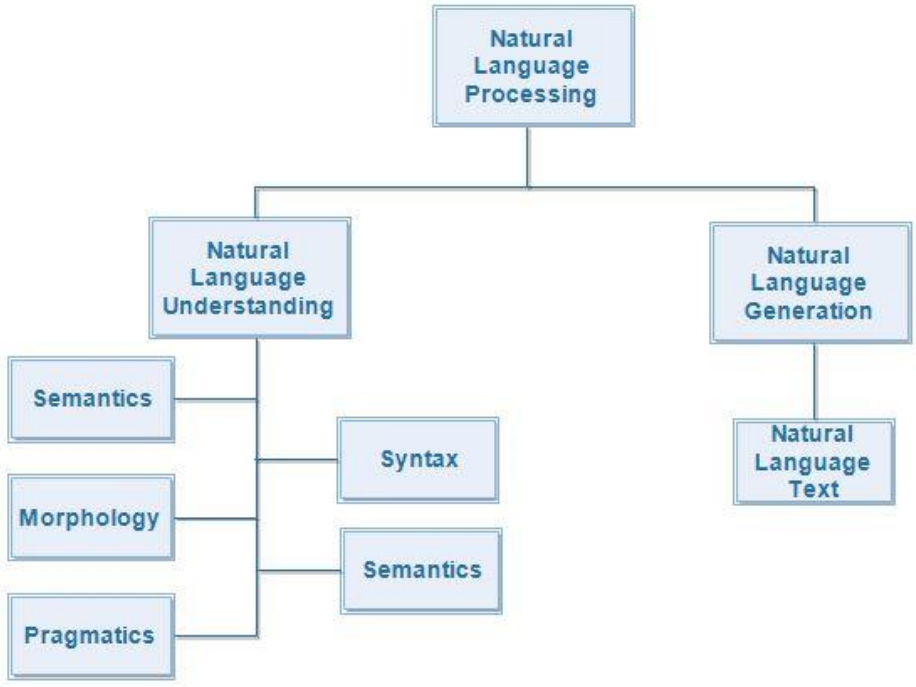



Figure 2: Broad Classification of NLP (e.g. by “Own illustration”, Source: (Vedantam, et al., 2020))

Linguistics includes phonetics, including words, external organs, word formation, sentence syntax, semantics and pragmatic syntax, that is, comprehension.

## 3.4 Brief history of NLP

The NLP did was trace all the way back to the 1950s with the advances in what is called the "Turing Test" and quality-based measure of design implemented in 1957. Improvements were delayed until 1990 because of less computational power and the system depended on complex arrangements of written by hand rules and limited vocabulary. There has been a growing interest in recent research and applications to introduce machine learning and regular growth in computational capabilities. NLP has recently been in the area of major landmarks in the application of voice recognition, dialogue systems, language processing, and deep learning technologies (Kalyanathaya & Rajesh, 2019). In the mid-1980s computational punctuation hypothesis turned into an extremely dynamic space of exploration associated with the law for the client's conviction and reason for importance and information and for the capacity to manage assignments like accentuation and subject.

NLP concepts and machine translation research work mostly done in 1990. Recent research work at NLP has leveraged the ability to use deep learning technology using statistical models, machine learning, and DAP-driven methods. Although NLP still faces a lot of obstacles (like human-computer interfaces), so many research attractions and it opens up plenty of opportunities for using robotics, automation, and digital conversion technology. Natural language processing research themes occasionally overlap with some artificial intelligence and deep running themes. This is usually the most recently adopted approach to performing NLP operations in the most efficient way.

## 3.5 Learning Model: Neural Network

Artificial neural network (ANN) biology consists of a set of 60 trillion bound neurons that are inspired to perform patterns within the human brain, the decision-making network. Based on this basic concept, the process of artificial neural networks begins to act as a single processor of neurons that are very common to each other. The concept of perceptron was demonstrated based on the neuron model of MacCulloch and Pitts (Negnevitsky, 2009). They have developed a simple and simple neural network for electrical circuits. To extract information from unstructured data, patterns and identify trends the highly efficient neural networks can be used that are unnoticed by humans and other computer technologies. I can. For this reason, neural networks are sometimes regarded as experts in given data and information (Maind & Wanka, n.d.).

### 3.5.1 Artificial Neural Network

An artificial neural network (ANN) is a framework that mimics the human brain's neural network while relying on the biological natural neural network.

Artificial neural networks have process components that is called neurons and it is the heart of artificial neural networks. Primary neurons build with five main functions: the input weighted synthesis function, the activation function and the output.

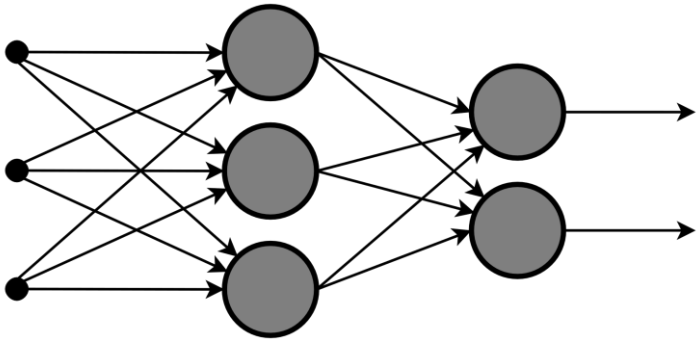


Figure 3: Simple Neural Network (Source: (Pngwing, 2021))

**Inputs ($X_1$, $X_2$, …, $X_n$):** This is a user-generated level with samples from the dataset.

**Weight ($W_1$, $W_2$, ..., Wn):** The weight decides how much input information will arrive at the output. For example; The weight $W_1$ indicates how much the output will be affected by input $X_1$. The weight values can change, this does not imply that the inputs are worthless.

**Summation function:** The full input of the cell is calculated by this function. For the calculation various function are available.

**Activation function:** The output value of this function is evaluated based on the input value. Some neural network models can develop activation functions.

The learning system of the network is important to determine the derivatives. As a result, the origin of a sigmoid function is the most widely employed function because it can be described in the function itself. All the cells do not need to have the same activation function. They may have different activation functions. The activation function is: Linear functions, sigmoid functions, hyperbolic tangent functions, sine functions, numeric functions (Çelik & Altunaydin, 2018). There are various valid functions such as Sigmoid, ReLU, tanh, and Softmax Function.

**Sigmoid:** Receive input of real values and scale between 0 and 1

Equation: $\sigma(x) = 1 / (1 + \exp(-x))$ …… (1)

**ReLu: "**Recorded Linear Unit" is the abbreviation for "Recorded Linear Unit." It multiplies a real-value input by a factor that is greater than zero (replacing negative values with zero)

Equation: $f(x) = \max(0, x)$ …… (2)

**Tanah:** Take an input of a real value and expand it to the limit [1,1]

Equation: $\tanh(x) = 2\sigma(2x) - 1$ …… (3)

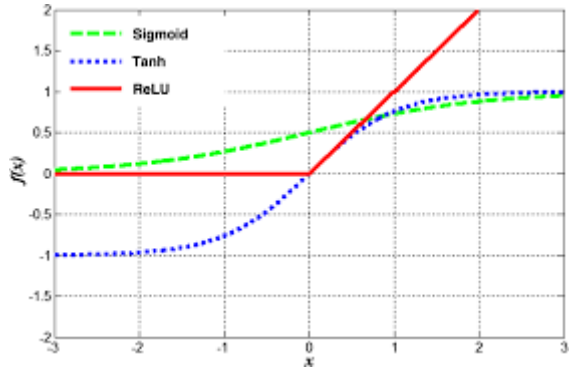



Figure 4: Different activation functions. (Source: (Wang, et al., 2017))

**Output:** The activation value determines the output. The last output generated can be moved to another cell or outside. For responses, the cell can use the output of the response as input (Çelik & Altunaydin, 2018).

**Importance of Bias:** The principal function of bias is that every node has a teachable steady who's worth will be provided (in addition to the common inputs received by the node).

### 3.5.2 Single Layer and Multilayer Artificial Neural Networks

In the first study, a single-layer artificial neural network begain. It is the most common feedforward nerve network and has no hidden layers. The classification of problems is the main feature of a network that can be selected linearly as a level. Problem inputs are multiplied by a weight and the values calculated after adding are classified as high or low according to their threshold values. As shown in Group-1 and 1 or 0 and 1, both weights and thresholds are updated during the learning process. The thresholds value is 1 for the output. Non-linear problems don't work for multi-layer artificial neural networks have been written as well as single-level artificial neural networks. Multi-layer artificial neural networks are the most common neural networks used today. A multi-layered network to investigate XOR problems during the investigation period. Multi-layer networks have three levels (Çelik & Altunaydin, 2018).

**Input level:** This level gets data from the rest of the world, yet there is no cycle at this level.

**Hidden layers:** Input level data is processed at this level. Most of the time, an insert could be enough to solve the problem. However, if the relationship between inputs and outputs is not linear or has some complexity, multiple layers can be used.

**Output level:** The data got from the interlayer is handled at this level and the yields identified with the information sources are distinguished. In the preparation of multi-layer neural organizations, the "delta rule" is utilized. Using multilayer network monitoring learning methods, inputs and outputs compatible with the input are exposed to the network. As indicated by the learning rules with reducing the margin of error, the margin for give and take between the output and the normal output is circulated across the network.

### 3.5.3 Feedforward and back propagation artificial neural networks

Artificial neural networks can be classified into two types based on the direction of connection between neurons: forward propagation and backward propagation. In a relay network, the signal passes from the input stage to the output level of the one-way link. Individual relay network cells' output values are sent to the next step as weighted inputs at the same time. The hidden level receives the input unchanged from the input level. This information is encrypted and processed in the output to determine the output from the network (Çelik & Altunaydin, 2018). Multi-layered detection and learning vector quantities can be an example of a predictive artificial network.

A key characteristic of error the output value of one cell is supplied either spontaneously or as input to another cell in backpropagation neural networks. Back propagation can be transformed into a barrier unit, a single layer of cells, or a second layer of cells. Because of this feature, backpropagation shows a dynamic behavior of artificial neural networks (Güneren, 2015). This network has earned its name with the ability to set inverse weights to reduce errors that occur at the output level.

## 3.6 Transformer

Before the introduction of translators, the problem of natural language processing was being used in methods or circuit networks of recurrent networks such as LSTM and GRU. Encoder-decoder structures with bidirectional LSTMs are a common model design in language modeling. RNNs have the advantage of being able to process any length sequence, however concurrent computing is not practical because of the nature of the input processing order. All things being equal, this implies that this sort of consecutive modeling is costly on schedule and memory resources.

In 2017 Vaswani et al. proposed another neural network architecture called Transformers. This advanced architecture is quickly changing the universe of natural language processing. In view of the design of this transformer, models like GPT and BERT have completely surpassed the previous sophisticated networks. It surpasses the previous approach that all current top models will be displayed based on this transformer-based architecture (Gillioz, 2019). Transducers have been developed to address sequence transformation or neural machine translation. Therefore, it is the task of converting the input sequence into the output sequence. This includes speech recognition, speech synthesis, text classification, and more.

Some memory is required to allow the model to perform sequence transformations. For example, translate the following sentences into another language (French):


Figure 5: The input is represented in rhombus shape, the model is represented in blue, and the output is represented in round shape, e.g. by “Own illustration”

*“This experimental rock band “Can” from Cologne is without a doubt one of the best bands in Germany. The band are responsible for creating the German Krautrock genre.”*

Above example, the word *"the band"* in the second sentence refers to the band *"this experimental rock band “can” "* entered in the first sentence. It can be important for translation of some sentences. There are many suggestions, this term refers to the word in the previous sentence.

### 3.6.1 Model Architecture

With an encoder-decoder structure the neural sequence models are built (Cho, et al., 2014). It is formed with the transformer encoder decoder architecture, however, using only the attention mechanism without repetition or convolution, resulting in simple parallelism. When the input sequence indicated by the encoder symbol ($x_1$, ..., $x_n$) is a continuous representation sequence z = ($z_1$, ..., $z_n$). Using a sequence of continuous representation z, the decoder produces the output sequence ($y_1$, ..., $y_m$) of the element in each case. (Graves., 2013).

Transformers follow this overall architecture using fully connected layers of self-focus and left and right of Figure 6, based on points stacked with encoder and decoder, separately.

#### 3.6.1.1 Encoder

The transformer made of the same stack of N = 6 layers of encoders. Each encoder layer is supported by two sublayers: a multi-head self-consideration layer and a fully associated feedforward layer.

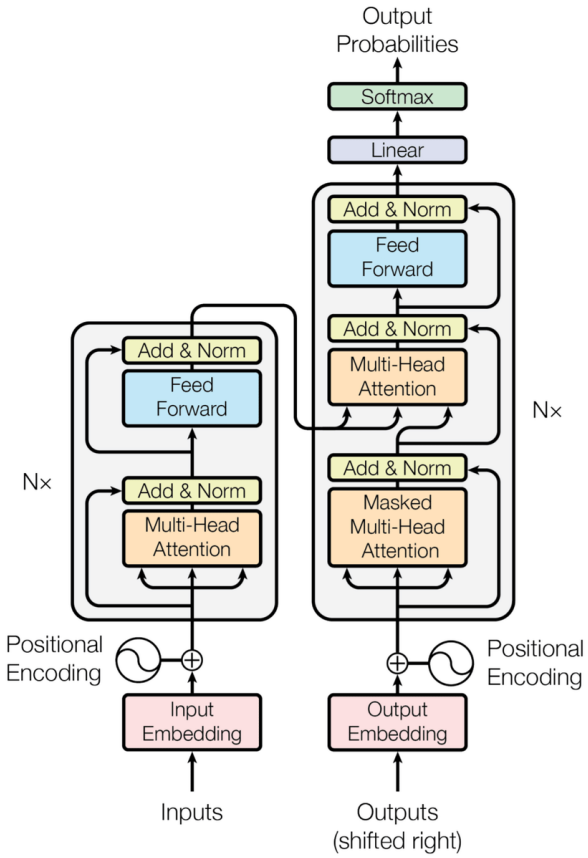



Figure 6: Architecture of Transformer (Source: Vaswani et al. 2017, p.3.)

Normalization of this layer continues with the remaining connections around every of the two sub-layers (Vaswani, et al., 2018). The LayerNorm (x + sublayer (x)) is normalized by the result of each sublayer. The sublayer (x) is a capacity that can be applied to the sublayer. The sublayer (x) is a function applicable to the sublayer itself. To facilitate these remaining connections, the containing layer of all sub-layers of the model produces the output of $d_{model}$ = 512 dimensions (Vaswani, et al., 2018).

### 3.6.1.2 Decoder

The decoder creates N = 6 stack layers, with each encoder layer consisting of two sub-layers. Also, the decoder is the encoder/decoder attention layer, there is a third sub-layer that uses the mask on the self-attention layer. Additionally, for encoders, the yield of each sublayer has the excess connections around each sublayer, trailed by layer standardization.

The decoder stack's self-attention sublayer prevents locations from being associated with locations near them. When combined with the fact that the output embedding is one place off, the expectation of position *i* can be based solely on known results in areas where *i* is not (Vaswani, et al., 2018).

Table 1: Original Transformer demonstration.

| Transformer | Layers | Hidden size | Embedding size | Attention heads |
|---|---|---|---|---|
| Vanilla | 6+6 | 512 | 512 | 8 |

### 3.6.1.3 Attention

The function of attention can be described as a set of maps with queries and outputs. In this case, query key values are vectors associated with those key values, and outputs are vectors related with those key values. The weight is the output value determined by the corresponding function of the query. The corresponding key is determined by each value defined as the sum of values (Vaswani, et al., 2018).

### 3.6.1.4 Scaled Dot-Product Attention

"Scaled Dot Product Attention" appears beneath the left-hand figure. Figure 7 is important to remember. Dimensions, $d_k$ keys, and $d_v$ values are used in input queries. Apply the softmax function to get the weight in the query dot product, each share by $\sqrt{d_k}$, and the price through all the keys. In reality, simultaneously calculate the attention function on a set of questions, grouping a matrix Q together (Vaswani, et al., 2018). Keys and values also group the K and V matrix together. Then the calculation the output matrix:

$$\text{Attention}(Q, K, V) = \text{SoftMax}\left(\frac{QK^T}{\sqrt{d_k}}\right) V \quad \ldots\ldots\ldots\ldots\ldots (4)$$

The attention and dot-product (multiplicative) attention are the two most regularly utilized attention functions (Dzmitry Bahdanau, 2014). The attention of the dot product without $1/\sqrt{d_k}$ scale factor is the same as our algorithm.

Single hidden layer of connector attention computes a steering function using a feed-forward network. Two similar theoretical complexities, but attention to the dot product, can be much faster and more space efficient with these highly optimized matrix quality codes. Although the two process $d_k$ work similarly for smaller values, the added attention $d_k$ (Britz, et al., 2017) for larger values, the product attention is covered on an unscaled scale. The evidence that $d_k$ dot products to greater quality, increasing product amplitude, push the Softmax function in an area with an extremely low gradient 4. In this sense, it is dealing with resizing $1/\sqrt{d_k}$ dot products (Vaswani, et al., 2018).

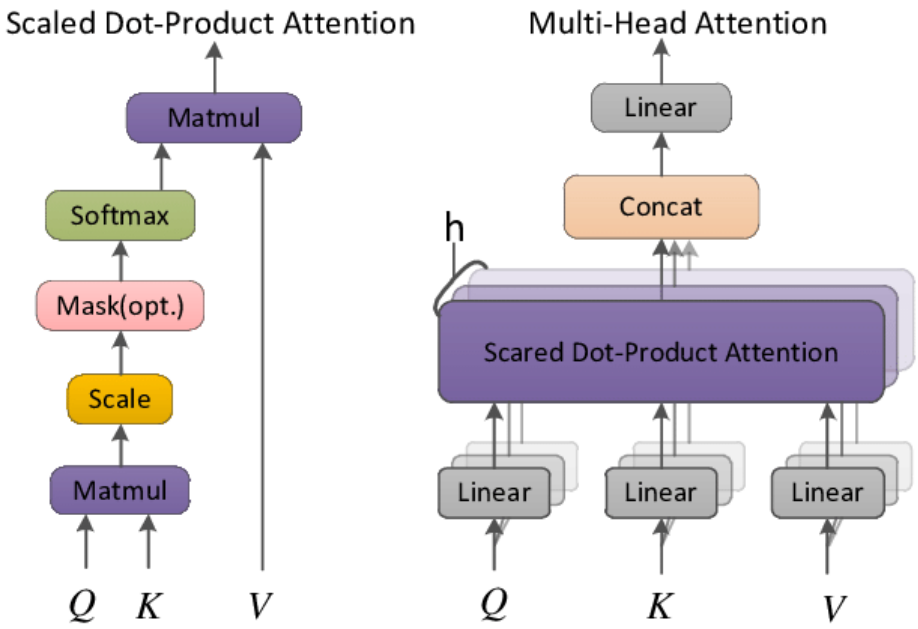



Figure 7: Scaled Dot-Product Attention and Multi-Head Attention, Source: (Vaswani, et al., 2018)

### 3.6.1.5 Multi-Head Attention

Draws multi-heads instead of summing the weights of a single attention. Capturing different aspects of the input determines the number of multiple attentions. It can get information about different sub-spaces through simple splicing of multiple independent attention (Vaswani, et al., 2018).

To know the odd representation, there is a unique linear transformation of the query, key, and value input representation as each major. Then the scale-dot draw is calculated parallel h times, it makes the so-called multi-Headed. The output is then condensed. Finally, a single linear transformation is applied, as shown in Figure 7. Multi-Head Self-Attracting will be able to learn related information from different presentation sub-spaces, because with this question, the key, each of the value sets is activated randomly.

Dimensions for questioning and what can always be different levels of value at the same time. However, the transformers, question, key, and value are all equal-sized vectors $d_k = d_v = 64$. The focal function output value vector is the accumulated circumference so far. The weight vector is interpreted as a consistency measure in the query with the corresponding key assigned to each item. In fact, it applies the focus function to the matrix $Q, K, V \in \mathbb{R}^{T*d_k}$ which contains a set of queries, keys and value vectors, respectively (Vaswani, et al., 2018).

### 3.6.1.6 Position-wise Feed-Forward Networks

Additionally, a completely associated feed-forward network is remembered for each layer between the encoder and the decoder, which is a sub-level of resale attention. Each word is applied in a distinct and transparent manner. It is made up of two linear transformations that are activated by ReLU.

$$FFN(x) = \max(0, xW_1 + b_1) W_2 + b_2 \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (5)$$

The linear transformation is the same elsewhere, but uses other parameters between the classes. Another way to describe this is the size of kernel 1. There are two convolutions $d_{model} = 512$ with input and output dimensions, and the inner layer has dimension $d_{ff} = 2048$ applied (Vaswani, et al., 2018).

### 3.6.1.7 Positional Encoding

Since there is no iteration and no modification for the model used to order our model sequence, the token's relative or outright area must be determined by the chain. In this way, add "Position Encoding" to the bottom of the encoder stack and decoder stack in the input integration. Position-encoded embeddings have the same dimension $d_{model}$ that can be compiled. Positional encoding has a choice, learned and modified (Kaiming He, 2016). Positional encoding is compiled up to the same dimension $d_{embed}$ = 512 tokens for encoding as embeddings and embedding. PE is defined as Positional encoding,

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{model}}) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (6)$$

$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{model}}) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (7)$$

The input vector encoding is the same as dividing by the first part of the sine. The cosine function determines the second part. It makes position $pos$ in this case and $i$ takes model dimensions, $d_{model}$ = 512.

Transformer design was initially presented for machine interpretation work. The authors have seen that their models perform shockingly well on different activities. Bert, Bert-based Modeling and Robert, XLNet also described the learning idea.

## 3.7 Bidirectional Encoder Representations from Transformers (BERT)

GoogleAI released a pre-trained model according to Transformers in October 2018. This model was able to obtain new and up-to-date results in 11 NLP benchmarking operations at publication times. Before publishing the BERT, the bidirectional encoder portrayals of transformers are short left and right, presented at the same time with regards to the chance of learning and the setting of the right word (Devlin, et al., 2018).

Transformer Bidirectional Encoder Representation (BERT) can be defined as a powerful NLP pre-training technique that builds on working in contextual representation. The main difference between BERT and other models is that it is the first deep bi-directional, untrained language representation (Chang, 2019).

Until then, bidirectionality had been accomplished by modeling two different networks for each of the presenter orientations, which would then merge. It emphasizes the idea of generating a good pre-trained model that can be used throughout the process. After a deep dive into the two-dimensional representation of unlabeled text, Bert can be utilized for “fine-tuning” or “feature extraction” after a profound plunge into the two-dimensional portrayal of unlabeled text. This model is only suitable for performing sentence-level operations at the symbol level, such as recognizing entity names, such as operations in that language (Devlin, et al., 2018).

The context-free model generates a word for embedding the presentation for each word vocabulary (Chang, 2019). For example, in models such as Word2vec and Glove "Test cricket" and "Turkish language test" would represent "Test" in the same way. BERT, on the other hand, takes into account the context of both directions. For instance, consider the following sentence, "I accessed the bank account", BERT represents "bank" using the preceding ("I am accessing it") and the latter ("account") references. This makes it very easy to fix and build various NLG applications with just one additional output layer (Hugging Face, 2021).

The most used Bart approach in this section:

### 3.7.1 Feature-based Approaches

In NLP feature-based training, a pretrained language model[2] is used to generate features based on some input tokens. These features are then used for training a small and possibly different model, like an LSTM or ConvNet for a specific language task.

The widely adopted representation of word learning has been a space of functional testing for quite some time, covering non-neural and neural strategies. Pre-trained word integration is a must-have in current NLP frameworks, providing major improvements to learned integrations throughout. When it comes to left and right, the correct extension of the word false has been used for the purpose of combining vector modeling objectives, left-to-right languages, as well as targets (Devlin, et al., 2018).

This method has been extended to more general aspects, such as sentence embeddings or paragraph embeddings (Rajpurkar, et al., 2016). To present the train sentence, previously used, the purpose of the sentence is to use the next rank of the sentence, the left-to-right previous sentence or denoising autoencoder derived purpose is a presentation of the next sentence word generation (Devlin, et al., 2018).

BERT, for example, can achieve this by taking the C token square output as a collective, sentence-level representation of all input tokens. Since Bert (like any Transformer) processes parallel tokens through his self-focus process, some information from the token overflows as well as C. For this reason, C can be a good sentence-level representation and can also be used to generate sentence-levels that have additional features used in different models (Devlin, et al., 2018).

[2] Although the grouping layer is included into the basic architecture of certain models, it is still insufficient to suit the final grouping and classification layer.

### 3.7.2 Fine-tuning Approaches

In a fine-tuned based approach to training an NLP model that mostly used with transformer architecture and the training consists of two steps:

1. Pre-train the model using a large unlabeled data set. This is achieved through the use of specific language actions example as language modelling, next sentence prediction and masked language modelling.
2. Finetune the model with a small to medium sized label dataset. The pre-training models that have generic language for this and understand the ability to effectively set own data set to tune model.

The advantage of these methods is that need to learn some parameters from the beginning. Because of this advantage, OpenAI GPT GLUE has achieved previously complex results on many sentence-level tasks in benchmark tests, at least to a certain extent (Devlin, et al., 2018).

### 3.7. 3 Model Architecture of BERT

The design of the Bert model is portrayed in a multistage bidirectional transformer Vaswani et (2017) and encoder based on the original implementation published on the tensor2tensor library. A detailed description of the structure and reference of the model is omitted because the use of the transformer is almost identical to the original implementation in the popular implementation. Refer readers to great tutorials like (Vaswani, et al., 2018) "The Transformer Annotations" (Hugging Face, 2021).

The biggest difference in Transformer Vanilla implementation is that Bert uses only encoder's stacks. Utilize 6 encoder levels rather than Bert Base 12. Bert additionally utilizes a bigger supplement size of 512 rather than 768.

Table 2 contains more information about BERT's model sizes. BERTBASE (L=12, H=768, A=12, Total Parameters=110M) and BERTLARGE (L=24, H=1024, A=16, Total Parameters=340M) are the two model sizes shown in the table (Devlin, et al., 2018). With self-attention, self-attention disguise, and encoder-decoder attention, Bert employs the transformer mechanism. These two masking processes serve a variety of purposes and should be approached in an unexpected manner.

Table 2: Available BERT versions (Devlin, et al., 2018)

| BERT | Layers | Hidden Size | Embedding Size | Attention Size |
|---|---|---|---|---|
| base | 12 | 768 | 768 | 12 |
| large | 24 | 1024 | 1024 | 16 |

Introduce some methods of autoregression using decoder masking transformers. With the translation of a target phrase source, the model is supposed to refer only to the locations that have been seen. Codes with masks are only defined on $-\infty$. The token [mask] is actually replaced by a token by masking the path in Bert, randomly replacing it with another token, or as it remains. When the transformer just masks its input process in different ways, the predictor Bert [Mask] is trained in T markers. Basically, Bert is the Transformer Encoder stack, which includes all of the important architectural features like interrupt, residual connection, leveling, and, of course, the care process (Devlin, et al., 2018).

For pre-training purposes, BERT uses two operations:

1. Masked Language Model (MLM)
2. Next Sentence Prediction (NSP)

### 3.7.4 Masked Language Model (MLM)

MLM involves optimizing the weight of the BERT to give the BERT a write and extract the same sentence. BERT to input so that one sentence and the same sentence output. However, before actually give that input sentence to BERT - the mask a few tokens (Devlin, et al., 2018).

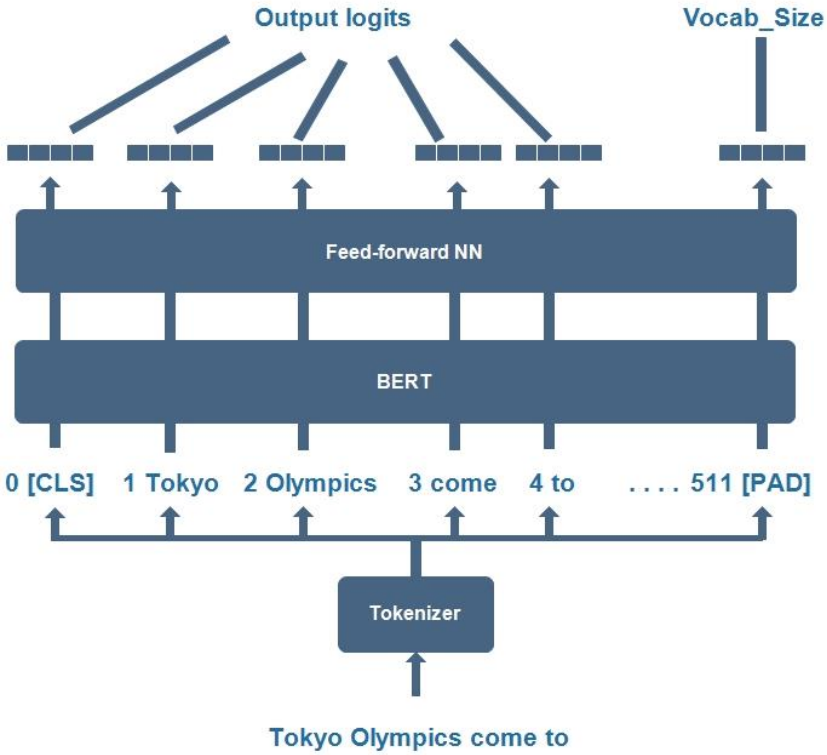



Figure 8: Before passing tokens into BERT — masked the token, replacing it with [MASK]. (e.g. by Own illustration)

So, actually inputting an incomplete sentence and asking BERT to complete it.

When pre-training to predict the original vocabulary ID according to context, the BERT token is 15% regardless of the unmasking chosen for potential masking. This is the last covering vector for the masked tokens passed using the vocabulary by the Softmax function. The original token is predicted using mask tokens and cross-entropy losses. Cross-entropy loss is a classification model that uses the observed phenomenon y = 1 to measure the performance of log probabilities that distribute probabilities to the output. (Devlin, et al., 2018).

Using this loss, calculated the required gradient change with Bert - and optimized the weight of the model.

### 3.7.5 Next Sentence Prediction (NSP)

The Next Sentence Prediction (NSP)[3] task allows BERT to derive relationships between various segments of a sequence that are not directly captured by language modeling (Devlin, et al., 2018). NSP was introduced to improve the performance of natural language reasoning and question answering tasks that need to understand the relationships between segments.

Pre-train the following sentence prediction tasks to train a model for understanding sentence relationships. A branch that can occur a bit from a single language corpus. Specifically, if sentence A and each pre-training example B are selected, 50% of time B will support the actual next sentence following A (labeled next sentence). The other half of the time corresponds to a corpus of arbitrary sentences (labeled as NotNext) (Devlin, et al., 2018).

### 3.7.6 Pretraining

The pre-training method basically follows the linguistic model of the pre-training literature. For the pre-trained corpus, the model used the book corpus (800 million words) (Chu et al., 2015) and English Wikipedia (2,500 M words) (Devlin, et al., 2018).

Using the WordPiece model is tokenized by the data (Wu et al. 2016). Then apply the masking method to each instance of the corpus. Bert uses duplicate versions of the dataset, and this redundant dataset impersonates a token for everyone. All additional, enter special tokens Sequences are limited to 512 tokens with a total length set to 512 or less. Sequences aren't too expensive, so the developer reduced processing expenses by using a sequence length of 128 tokens, which is just 90 percent of the pre-preparing stage.

[3] Instead of "segment," BERT uses the term "sentence." A "sentence" is defined in this paper as any chunk of text that isn't a sentence in the traditional sense. Any span of text is referred to as a "segment" or "order" in this context to prevent confusion.

The remaining 10% of the steps used 512 tokens to learn "full" sequence length inclusion of positions (Devlin, et al., 2018).

The BERT is predefined with a batch size of 256 and the sequence length is set to 512 to generate approximately 128,000 tokens / placements. This indicates that the remaining 384 positions were packed using the length of the 128 sequences. Adam is used in the optimizer along with learning. The ratio of $1e^{-4}$ is $ß_1 = 0.9$ and $ß_2 = 0.999$ (Kingma, 2014). In addition, an L2 weight loss of 0.01, a learning rate straight miscalculation with a learning rate preheated to the first 10,000 stages were used. A GELU (Gaussian Error Linear Unit) is implemented with a dropout probability of 0.1 at all levels similar to GPT as the activation function (Hendrycks, 2016).

### 3.7.7 Tokenization and embedding

For pre-training, the vocabulary that Bert uses WordPiece to embed is approximately 30,000 tokens. The size of the Bert model is 28,996 vocabularies and requires training in uppercase text. A vocabulary consists of 30,522 cards when using a model trained in non-capitalized or uncased text. In all the models reviewed, BERT is the model that can be used for pre-training with or without a case (Devlin, et al., 2018).

The WordPiece model is displayed on a sub-word basis rather than the entire word. Words are isolated into a limited arrangement of normal subwords, likewise called "word parts," making it simple to manage uncommon words (Wu, 2016). Using the WordPiece model, Bert looks at that whole word, which is more efficient based on the model that character and is very flexible and finds a balance between models. To separate the entire word by sub word, the following words are displayed in the hash.

So, the model can recover the order without ambiguity (Wu, 2016). As already mentioned, a special token for each order sequence [CLS] classification. This token [CLS] is an extra part, and will be further processed when it is hidden by other fine-tuning layers when performing sorting operations to adjust BERT. Since the token has been used for NSP pre-training purposes, the author claims that it is the conclusion of sequence presentation after fine-tuning (Devlin, et al., 2018).

Bert contains binary segments and can be easily identified and position embedded. This section can know that each token brand has a maximum of 512 tokens in the absolute position of each token (Devlin, et al., 2018).

## 3.8 XLM-Roberta

Another methodology is to prepare a multilingual model, which is a solitary model that can deal with different dialects at the same time. Recent results prove that multilingual models can have a better performance than monolingual models, especially for low-resource languages (Conneau, et al., 2020). In particular, XML-R, with a 100-language model was launched by Facebook AI researcher, Cross-Linguistic Transfer State-of-the-art results were achieved.

The multilingual model XLM-RoBERTa or XLM-R has greatly expanded the amount of multilingual training data that was used in the untrained MLM pre-training compared to previous work and has reached the latest state in both monolingual and multilingual benchmarks (Ou & Li, n.d.).

Multilingual mask language models (MLMs), such as mBERT (Devlin, et al., 2018) and XLM, have taken the place of taking on large-scale Transformer models of multiple languages (Vaswani, et al., 2018). This model enables effective cross-language transfer, as observed in several benchmarks, including cross-language natural language reasoning (Bowman, et al., 2015) and recognition of named entities. However, all these studies conducted pre-training on Wikipedia with a relatively limited scope, especially for languages with limited resources (Conneau, et al., 2020).

XLM-R (Conneau, et al., 2020), where the R stands for RoBERTa (Ruchansky, 2017). According to the name, it assumes that the model is combination of XLM and RoBERTa but that is wrong. XLM-R takes a step back from XLM, eschewing TLM purpose, and only gives Robert training on a huge, multilingual data set on a huge scale (Conneau, et al., 2020). The unlabeled texted in 100 languages is extracted from the Common Crawl dataset, a total of 2.5TB of text.

It is a Robert fashion, that is, trained to use MLM purposes only. In fact, Robert's only significant difference is vocabulary size: 250,000 tokens compared to Robert's 50,000 tokens.

The cross-linguistic approach it uses with XLM and Robert, increasing the number of training examples for languages and new models, training self-inspected publicly available CommonCrawl data from two more terabytes that have been cleared and built on cross-linguistic representations. There are no new labels for this low-resource language produced by Corpora, which includes scaling the amount of training available to data for those languages by two orders of magnitude (Conneau, et al., 2020).

During fine-tuning, the capabilities of multilingual models have been used to use multi-language tagged data to get better the performance of low-flow tasks. This model has been possible to achieve primary results at linguistic cross-reference points and has greatly enhanced performance by the language of the monolingual BERT model (Conneau, et al., 2020).

XLMR improves mBERT in cross-language classification with an accuracy of up to 23% in the author Wong language. With an average accuracy of 5.1% in XNLI, an average F1score of 2.42% in recognition of specified entities, and an average of F1score of 9.1% in response to cross-language questions and answers, it surpasses previous sophistication. XLMR Adhesives and XNLI benchmarks evaluated by Fine Tuning using this result are competitive with the latest pre-models including Robert. (Liu, et al., 2019).

# 4

# Baseline and participator's models

The multi-language with English and Bengali research haven't done yet. The combination of other languages is popular and used traditional model like SVM, Naïve Bayes, CNN, RNN and LSTM. A baseline model was set where language combination was French and English. A multi-language research paper was published under the name as "Multilingual Fake News Detection with Satire". They have used French and English languages (Guibon, et al., 2020).

In this paper, the researcher used various automated methods for detecting fake news form of statistical text analysis of the vaccine dataset provided by Storyzy. Although the performance of CNNs was excellent in the identification of larger classes (both fakes and reliable), gradient-boosting decision trees using feature stacking methods gave excellent results in satiation detection. Efficient sarcasm detection, a combined source, results in a higher cost of classes to be achieved using certain models, other findings. Predict fake news and satire news from reliable news that merges unnecessary information.

Storyzy Company provides this hackathon dataset in (.tsv) format. The corpus has massages, comments from different websites in English and French, as well as annotations. These annotations are automated transcripts of YouTube videos in French, covering widely publicized fake news topics. The task is divided the text into three classes: fake news, trusted or satire. The researchers compared the performances with hackathons best performances in identifying fake news.

The main job of the hackathon is to get the best results from the two classes. Though the satire class seems the challenging and competing. So, they decided to classify all texts with satire class. To do this, they focus on the best way to classify text in English, copy French, and French it using an experimental method, closer to the effect of data presentation.

They use the chatNoir API to find news headlines in web snapshots and compare the news text to the top ten results (Guibon, et al., 2020). After using NLTK to segment English text and French text and restore morphology, use a tokenizer to preprocess the text. The tokenizer splits the text with spaces and apostrophes and removes words. After that, the text was classified by NLTK FrenchStemmer. Then for each question / article pair get a "similarity".

The ChatNoir API is used to find instant news headlines from the net, and if they receive and test the top ten domain names that belong to satirical, fake or credible websites. The predefined list they trust is used as a list of famous websites associated with forgery or satire. Establishing lists, the used domains on training corpus that collected from online.

This feature has three classes: fake sites, satirical sites, and trusted sites. Each parameter starts at 0. If the domain is in the list, it will be 1. The first result is that 74% of fake news headlines are searched for fake news websites, 4% are satirical websites, and 22% are links to trusted websites. However, fake sites contain reliable news with 50% frequency and 9% satire being the biggest problem. The rate of imbalance.

To obtain classifiers that can generalize predictions more efficiently, TFIDF (tf) FastText, Word2Vec (wv) and Hashing Trick (hv) tried to present a variety of data.

CNN achieved the best micro F1 score. However, the optimized d LGBM (Light Gradient Boosting Machine) system works much better with a macro F1 score. All results show how to improve the overall macro F1 score by estimating the performance differences of each class and the good ironic rare cases of the LGBM classifier. So, CNN can better manage subtle classes like tree satire of encouraged decisions, while looking better to distinguish larger classes (fake news vs. trustworthy). increase. Good ironic detection can be gained by mixing Decision Tree Boosting with gradient type merge embedding.

# 5

# Data Collection and Analysis

## 5.1 Data Collection

The dataset identified, in order to implement this task is the BanFakeNews dataset for Bengali Language (Dataset., 2020) and Fake News for English Language (Dataset., 2019). Additional data was collected by web scrapping. For data collection, I selected two the most well-known news broadcasting media as truthful medium. These two news broadcasting media are "*The New York Times*" and *"The Guardian".* The two-news media are offering their Application Programming Interface (API) for developer. I used their API to collect their news and stored our local storage.

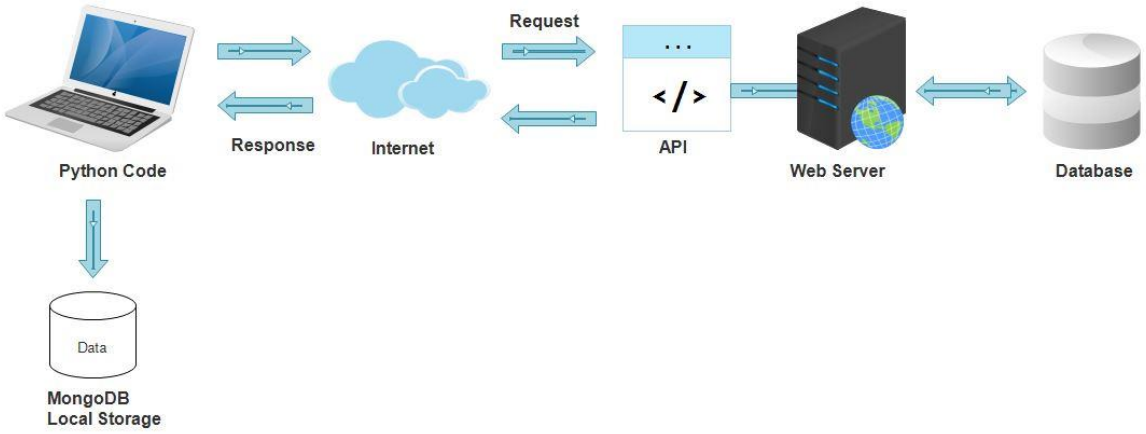



Figure 9: Data collection by API. (e.g. by Own illustration)

An API is a set of programming code that allows the transfer of knowledge between one software package and another. It also contains the terms of this knowledge exchange.

The way to offer data/functionality by occupation the API of computer code that needs access to info from another computer code (e.g., rate of X flight for a particular date) or practicality (e.g., a route from purpose A to purpose B) different computer code provides data/functionality requested by the previous application. There are two types of public APIs, free and commercially available. The Open API definition suggests that the options for each of these types of APIs are universal and may be used with limited Terms of Service. For example, you can create an application that uses the API without the explicit approval of the API provider or the required license fee. The any definition stipulates that the API will be used smoothly to create and test applications if the documentation associated with the API description must be forced.

I used the public API of "The New York Times" and "The Guardian". Public APIs are called developers or external, and these APIs are available to all third-party developers. Generic API programs increase brand awareness and enable additional revenue sources when executed properly.

I create an account on the developer portals "New York Times" and "Guardian" and request an API key. After collecting the keys, I write the Python code using the BeautifulSoup library. I also create a MongoDB database on the laptop as local storage.

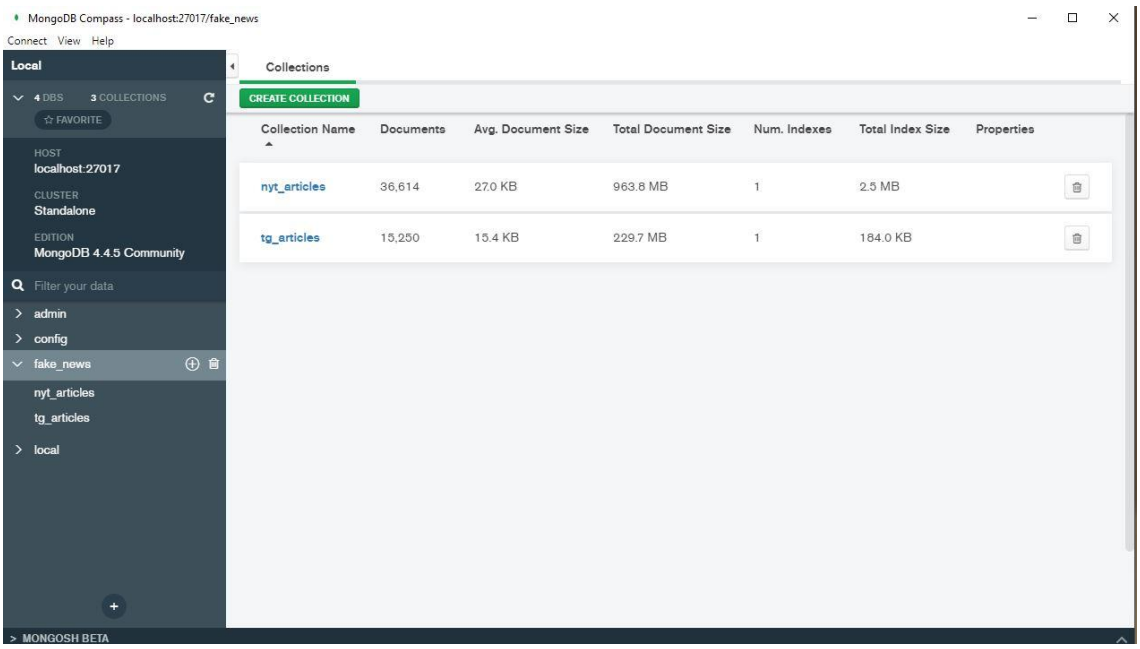



Figure 10: Web Scraping data store in MongoDB

With BeautifulSoup library and API Key, I sent requested to the webserver of respective news media. The webserver checked our request and verify our API Key. When the verification was successful and it returned responded with data. I have collected “The New York Times” news data from 2021-04-16 to 2001-06-14 and save the data on the MongoDB database. In our collecting dataset, I had the author’s name, the title of news, the main body of the news, news date, URL, country name, source, and word count. The total dataset size is 242.74 MB with 36.6K rows.

I have collected “The Guardian” news data from 2018-01-01 to 2018-11-01 and save the data on the MongoDB database. In our collecting dataset, I had the author’s name, the title of news, the main body of the news, news date, URL, country name, source, and word count. The total dataset size is 80.55 MB with 15K rows.

Both web scraping datasets were merged and labeled as real news. The datasets weren’t structured way to analyze. The datasets were polluted with HTML symbols, unusual coma, space, no column, etc. I cleaned the datasets separately then merged them. After merging the dataset, the total dataset size is 320.84 MB with 52K rows and saved as CSV format.

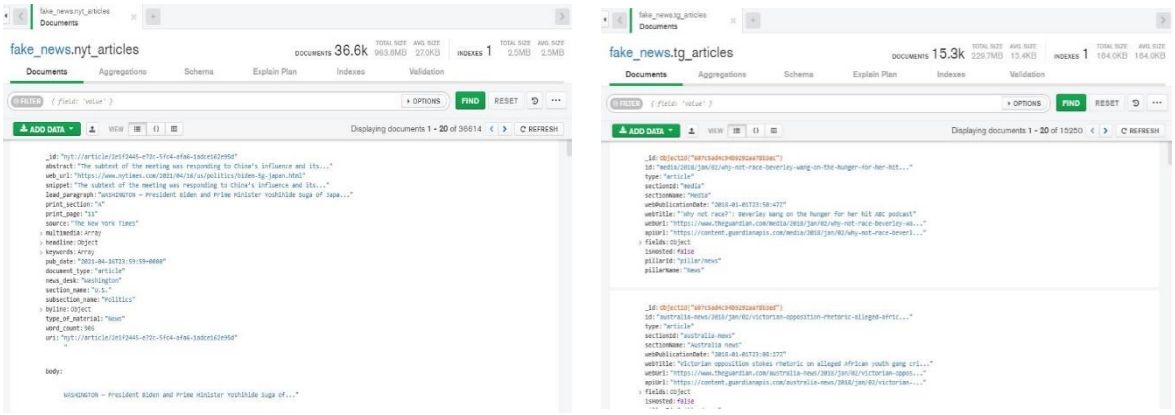



Figure 11: Web Scraping sample news data

Statistics of both datasets are given below:

**English Language:**

Total attributes: 9

Total Data: 52K (real news) & (18K)

Size of Dataset: 320.84 MB & 118.18 MB

Fake and Real News Ratio: 52K:18K

**Bengali Language:**

Total attributes: 9

Total Data: 60K

Fake and Real News Ratio: 4537: 54723

Size of Dataset: 277.76 MB

## 5.2 Data Preprocessing

Before evaluating the overall function of the model and analyzing the manual hyperparameters, the raw data must be processed. Data preprocessing steps are as short as possible.

Analyzing, applying machine learning algorithms on text data, it needs some preprocessing. The raw text datasets are mixed noise. Several techniques are applied to transfer raw text into a prepared-modeling form (Shah & Virendra, 2020). Below I discuss about prepressing steps on “headline” and “content” columns:

- **Stop Word Removal:** Stop words are that words used as a common word and doesn't have any context. These words have no meaningful information, so I remove these words from both headline and content. For English language, stop words are like preposition ("of", "in", "on"), articles ("a", "an", "the") and conjunctions ("and", "or"). On the other side, Bengali languages have also same common word. These are *('অতএব', 'অথচ', 'অথবা', 'অনুযায়ী', 'অনেক', 'অনেকে', 'অনেকেই', 'অন্তত', 'অন্য', 'অবধি')*

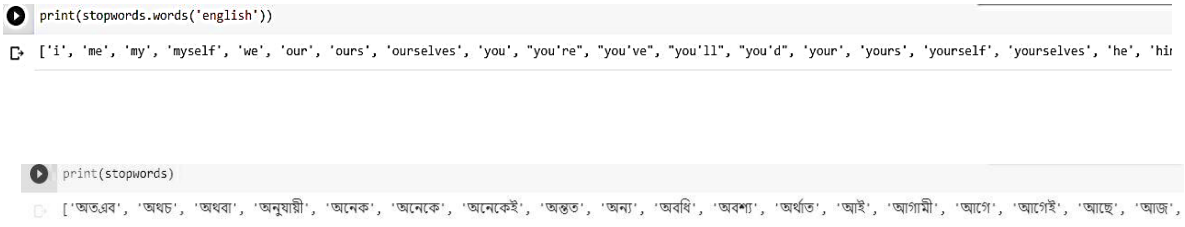



Figure 12: List of stop words form own data

Processing these less important stop words can take significant time, and removing stop words as part of data pre-processing and it is a first step in natural language processing.

- **Punctuation Removal**: Punctuation in natural language provides the grammatical context of the sentence. Punctuation like a comma that cannot add much value to the understanding of the sentence (Shah & Virendra, 2020). The figure illustrates the method of eliminating punctuation.

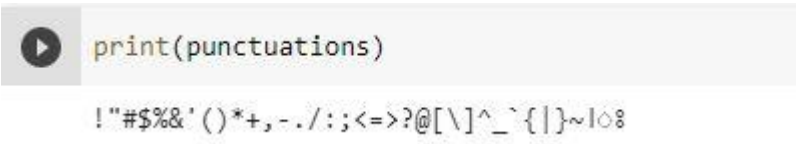



Figure 13: List of punctuation form own data

- **Lemmatization and Stemming:** Lemmatization and steaming both process the word to reach into its root. Stemming remove prefixes and suffixes from the word to reach the root. So, sometimes the steaming word has no meaning and no actual word. On the other hand, lemmatization build the actual word. Example:

  Lemmatizing: considered, considering, consider → "consider"
  Stemming: considered, considering, consider → "consid"

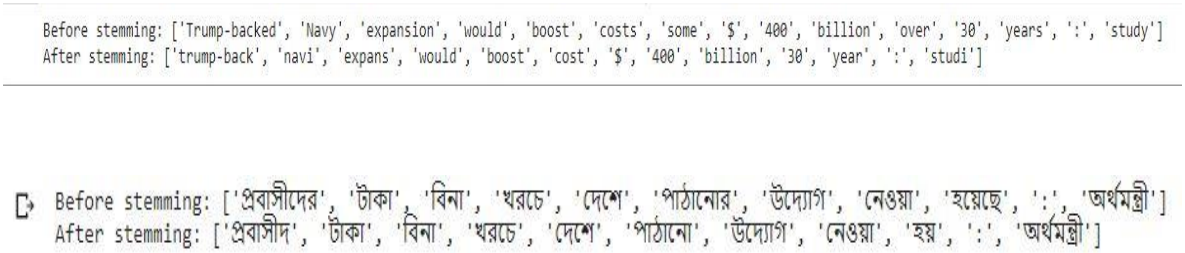



Figure 14: Example of English and Bengali Steaming from own data

So, when to use steam or lemma? It depends on application. Lemmatization takes much time to process. It uses WordNet corpus. it works slower than stemming when it produces lemma with stop words. Parts-of-speech also must be defined to obtain the correct lemma. Besides, steaming is faster and it copy an algorithm with the right path to build the words. I used steaming for Bengali dataset and lemmatization for English dataset because I don't need actual word for our model.

- **Tokenization:** Tokenization is a method of dividing a piece of text into smaller units. Here, the tags can be words, letters, or subworlds. Therefore, tokenization is roughly divided into 3 types of tokenization of words, letters, and subworlds (Ngram letters) (Guderlei, et al., 2020).

For example, a sentence: “I love my country.”

The most common way to create tokens is based on space. Assuming the space as a boundary, the tokenization result of the sentence is 4 tokens. “I”, “love”, “my”, “country”. Here each token represents a word, so it’s an example of word tokenization.

Same as Bengali sentence: *"আমি বাংলায় গান গাই।"*

After tokenization it shows: *"আমি", "বাংলায়", "গান", "গাই", " ।"*

- **Duplicate and missing data:** The data may contain duplicate observations. I have to analyze the data set to make a decision on whether to drop duplicates or not. In our data set, there is a chance of two observations. Duplicates occur due to some data entry errors. These types of duplicates must be dropped. Similarly, missing or null data were also dropped.

- **Remove HTML tags:** The data was collected from website. So, it may have some html tags. By using python BeautifulSoup libraries, it can easily remove the tags.

- **Expanding Contractions:** Contraction’s word or abbreviated version of the word. In the English case, contraction is often created by removing the vowel from the word. In its expanded, original form each compression transformation helps with text standardization. Examples don’t, isn’t, I’m to convert into full form as do not, is not, I am.

- **Removing accented characters:** It makes the conversion and accented character values between ASCII characters. A good example is to convert é to e.

- **Remove extra whitespace tabs:** Sometimes paragraph have extra whitespace and long tabs. It may cause data noise. So, it was removed.

## 5.3 Text Analyzing

**Dataset ratio:** The fake and true news data are labeled as “0” and “1” where “0” indicates fake news and “1” indicates true news.

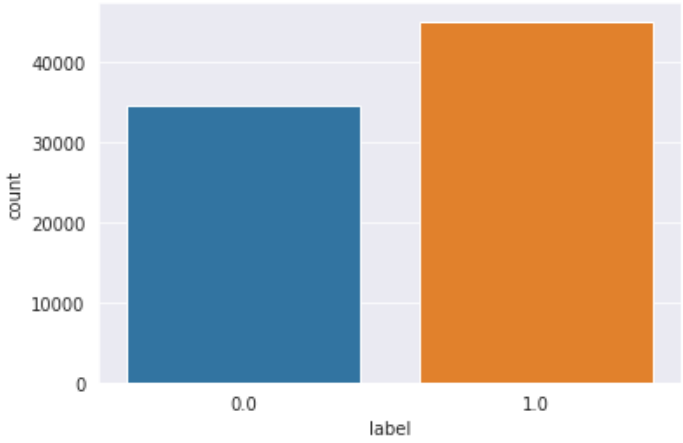



Figure 15: Fake and true data ratio

Datasets have 45000 true data and 34613 fake data where English and Bengali both are combined here.

**Length of news**: The length news means how many letters any available in the news. Example: “I have a cat”, The length of this sentence is 12 including the space.

The raw datasets of the English and Bengali were merged. The average length of the headline is 63.49 and content is 3657.50.

Table 3: Average news length of news

| Dataset | Headline | Content |
|---|---|---|
| English | 69 | 4075.92 |
| Bengali | 44.28 | 1783.95 |
| English and Bengali | 63.49 | 3657.5 |

I also counted the length of headline for each news. Below a bar graph is shown:

The maximum length of headline news was 113 and low 3. Other side, the maximum length of content news length was 109925 and low 28. So, the length of headline and content is imbalanced.

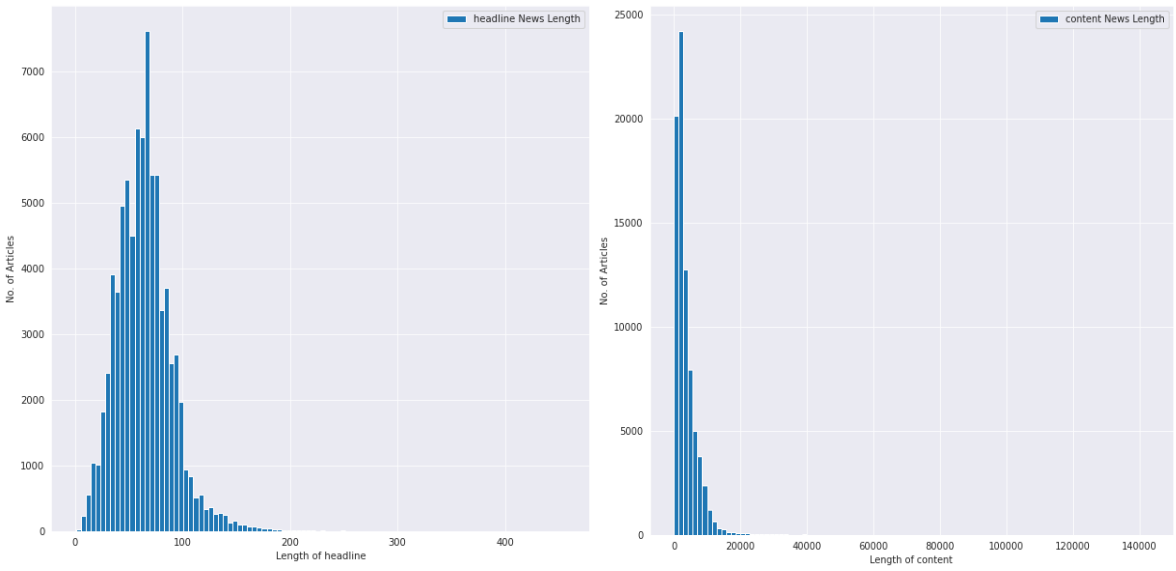



Figure 16: Headline and Content length of all news

**Word cloud:** The word cloud is a simple words visual presentation for text processing where most frequent word are showed with bigger and bolder format, and different colors. The less frequent words show smaller and smaller words are less importance.

I created word cloud for our dataset. I divided our dataset into true news and fake news. Therefore, I can find most and less frequent words in true and fake news. Below is showed true news word cloud where “Trump”, “Russia”, “China”, “Say”, “New” are most frequent words and “bid”, “job”, “killed” etc are less frequent words.


Figure 17: True news word cloud, e.g. by Own illustration

Below is showed fake news word cloud where “Video”, “Trump”, “Obama”, “Hilary etc are most frequent words and “List”, “Gun”, “put” etc are less frequent words.


Figure 18: Fake news word cloud, e.g. by Own illustration

**Top word:** Using python libraries, I found top 30 words from real news dataset. So, I can check that real news have much ‘said’, ‘mr’, ‘year’ etc words. These top 30 words used 30000 to 210000.

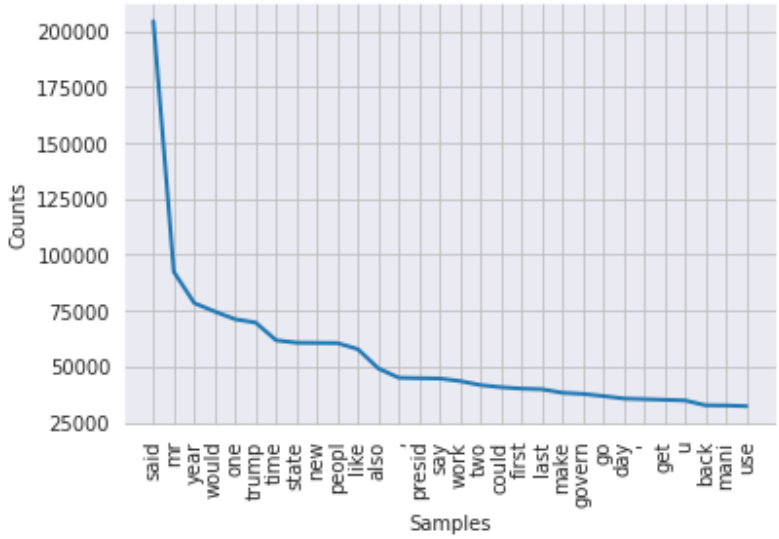



Figure 19: Top words in real and fake news, e.g. by Own illustration

## 5.4 Linguistic Analysis

The difficult task is to find out what people are saying in written language. The bottom line is that computers don't really understand human language. Yet, computers can do quite well in what I am doing later: gaining ideas and sensations from the text. The term linguistic analysis covers several areas. I'll use it in the narrow sense of trying to extort money from computer text. Linguistic analysis is the theory behind what computers do. I say that when a computer performs analysis based on theory, it does Natural Language Processing (NLP). Linguistic analysis is the basis of lexical analysis.

The linguistic analysis consists of steps that are used in almost all attempts by computers to understand the text. It is good to know some of these terms.

- Sentence detection
- Tokenization
- Lemmatization and cleaning
- Part of speech tagging

The linguistic analysis is a complex and continuously developing NLP science. There are several techniques and terms for linguistic analysis that have been developed but each has its own strength and weakness.

### 5.4.1 Lexical Diversity

The study of lexical types of frequencies and their distribution in the text has been a focus of natural language processing and linguistic research in recent decades (Coats, n.d.).

Although the terms lexical variation, and logical flexibility can be utilized more or less interchangeably, lexical variation is defined as the degree of vocabulary inequality in the language samples given here (Johnson, 1944).

As imagined, a lexical variation is the totally different from vocabulary density. The more frequent the ratio of linguastic sample words, the more pronounced the disparity, the higher and lower the color density change, respectively. The term change in vocabulary is generally favored by the language community as it is considered an indicator of dictionary quality and language fluency: There is an overall basic idea that a higher shading variety is 'something worth being thankful for' Ability." (D, et al., 2004). Indeed, research has shown that if listeners fully read and listen the speaker's message, their perceptions of reliability, efficiency, refinement, socioeconomic status, and effectiveness of communication have a positive relationship with the diversity of the speaker's vocabulary. (JP & M, 2002).

### 5.4.1.1 What is Lexical Diversity?

Lexical diversity is a measure of how many individual words appear in a text and can be calculated in different ways (Nicolas Be´rube´, n.d.).

As all know, the Lexical Diversity (LD) measure is sensitive to text length, and many revised LD measures have been proposed.

The next section summarizes the four LD systems in terms of characteristics.

i. **Type-Token Ratio:** It is a proportion of the quantity of various words and the all-out number of words (Nicolas Be´rube´, n.d.). So,

TTR = Type Count/Token Count

Take an example of a sentence: "*He is a boy and he is nice*."

The above sentence, which have eight words tokens and six opposite words: 'He', 'is', 'a', 'boy', 'and', 'nice'. Except for "a", Each sort of word begins with the usage of two-word labels once displayed in this sentence. It is imbalanced among markers and types that makes it possible to quantify vocabulary diversity: the more redundancies of words in a language test, the more noteworthy the awkwardness and accordingly the more prominent the vocabulary diversity.

which means that its TTR score is 6/8 = 0.75.

ii. **Guiraud:** The whole number of words separated by the all-out number of different words is the square root proportion (Koizumi, 2012). So,

$$\text{Guiraud} = \text{Type Count} / \sqrt{\text{Token Count}}$$

From above example, the Guiraud is 2.12 (i.e., $6/\sqrt{8}$).

iii. **voc-d:** This method finds a newly constructed type-optimal curve by taking a subsample of random numbers 35,36, ..., 49, and 50 tokens from the data and calculating the average token ratio of each of these lengths. Token rate curves (from a series of curves generated by equations that differ only in the values of a single parameter). Diversity measurements give the values of the parameters corresponding to the optimal curves. The entire procedure can be repeated several times to reveal.

Suppose we have 50-word tokens with 25 different or type count words. From 50 words, randomly selected 35 tokens. The selected word cannot be changed. The selected 35 token TTR is in the form of 0.57 to 20 and 35 tokens. Then again, we randomly selected 35 tokens and calculated the TTR. This should be repeated 100 times to calculate the average TTR of 100 segments of 35 tokens (0.57). Similarly, we keep the average TTR of 100 segments of 36 tokens (0.56), the average TTR of 100 segments of 37 tokens (0.54), etc. up to the average TTR of 50 100 segments generate tokens (0.50). The 16 averages for every token part and displays the TTR curve. Determine the best D incentive for the TTR curve from which the theoretical curve is derived empirically. The D for 50 token text is 12.00 (Koizumi, 2012).

iv. **MTLD (Measure of Textual Lexical Diversity):** Text requests from left to right and text requests from right to left are both used to estimate in MTLD. Each run gives a weighted average (variance), the 2 means are averaged again to retrieve the final report value (the 2 variances are also averaged).

This property specifies whether the reported mean should also be calculated based on the number of cases that may differ between the two analyzes ("inside_and_between" value) ("inside_ only" value). The default is "inside_only" to match McCarthy and Jarvis (2010), although the author of this implementation thinks that choosing "inside_and_between" is more consistent (Koizumi, 2012).

### 5.4.1.2 Lexical Diversity of Experimental Dataset

Let's define a measure for lexical diversity to find out how many unique vocabs are used in Fake News articles. For that I separated the dataset into two parts, one is real news and the other is fake news. Alongside, the dataset has two sections headline and content. So, I compared our lexical diversity separately for both headline and content.

Let's define the lexical diversity measure as $= \frac{\text{number of unique words in one (target)category}}{number\ of\ words\ in\ both\ (target)categories}$

Table 4: Lexical Diversity for headline

| Lexical Diversity of headline | Real news | Fake news |
|---|---|---|
| | 0.056 | 0.057 |
| Lexical Diversity of content | 0.014 | 0.0096 |

Interestingly, it seems that the headline of fake news has more lexical diversity than real news. Their authors seem to have a wide vocabulary. But the content has a different view. The content of real news has almost 1.5 times greater lexical diversity than fake news.

### 5.4.2 Punctuation Analysis

In this section, I focus on punctuation and try to find out how much punctuation is available in our dataset. Six punctuations were searching and these were: Period (.), Bengali period or full stop ( ।), comma (,), Question mark (?), Exclamation mark (!), and Semi-colon (;). Similarly, I separated the dataset into two parts, one is real news and the others is fake news. Alongside, the dataset has two sections headline and content. So, I compared punctuation for both real and fake news.

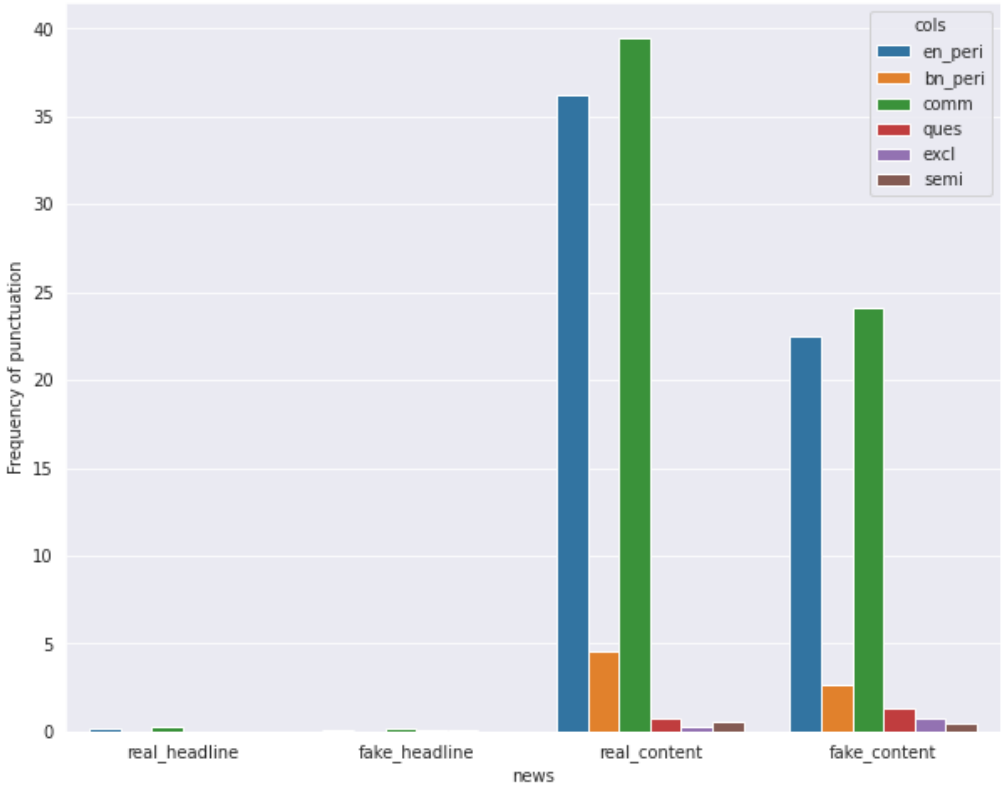



Figure 20: Average punctuation frequency from own data

The above bar graph shows the average punctuation frequency both real and fake news. According to the graph, the fake headline has maximum Question mark (?), Exclamation mark (!), and Semi-colon (;) than real headline news. So, writers used this punctuation comparatively more than real news.

Besides, same things are repeating on fake news content. Real and fake content have all types of punctuations but real content news has a bigger collection of period, Bengali period, and comma. In conclusion, both fake headlines and content news have Question mark (?), Exclamation mark (!), and Semi-colon (;).

### 3.4.3 Average Text Length

In this section, I checked the text length of both real and fake news. Both real and fake news have headline and content news. I compared between real headlines and fake headlines and also the same in content news.

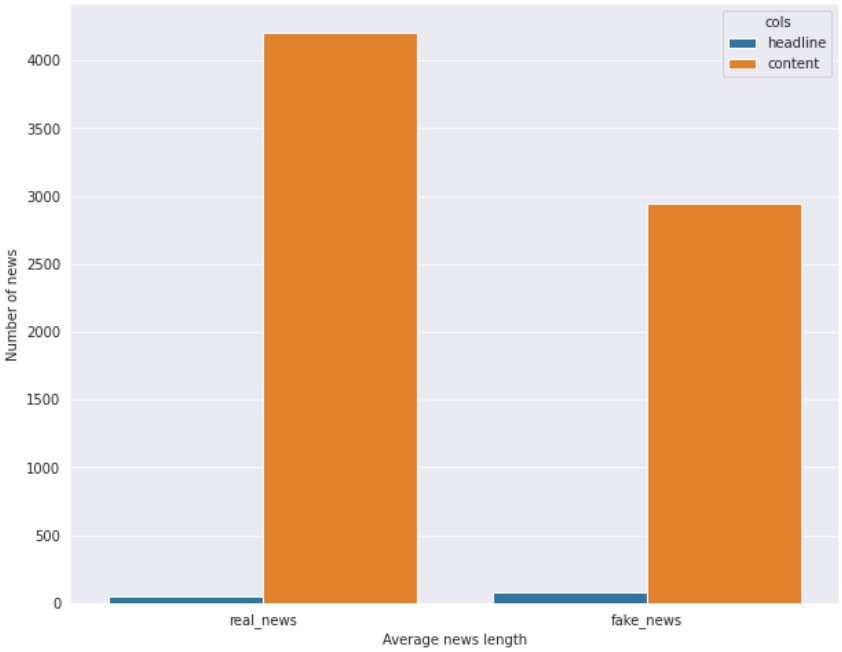



Figure 21: Average length of news from own data

The bar graph shows that the average length of fake headline has more length than real headline. In contrast, real content news has bigger length which is almost 1.5 times.

### 5.4.4 Number of Words in the headline and content

How many words are used in real and fake news? Is there any conflict number between real and fake news? So, I counted

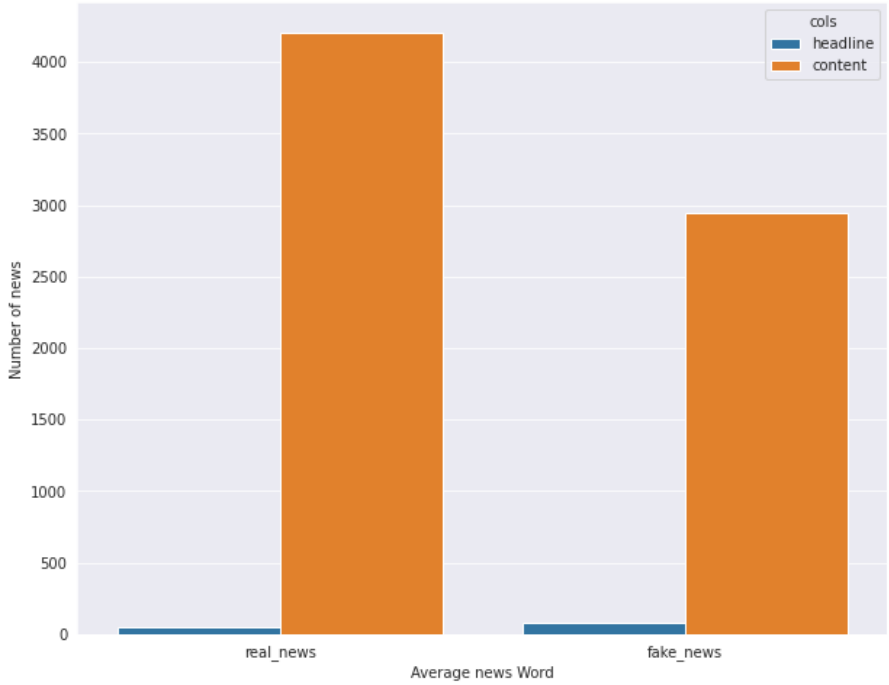



Figure 22: Average news word from own data

Below the bar graph are shown the result. Fake headline news has a bigger word count than real headline news. Though the dataset has less real news, the word count of real content has higher. The interesting fact is that real news has fewer headline words and high content words. At the same time, fake news has higher headline words and fewer content words.

## 5.5 Data split and Concatenation

The dataset is imbalance. The ratio of false and true and English and Bengali news ratio is also imbalance. To overcome these problems, the data was split correctly. The English and Bengali dataset was split as train, validation, and test dataset must have the equal ratio of Bengali, English news, and also false, true news ratio.

Table 5: Data distribution before splitting

| **Language** | **Instant** | **False (0)** | **True (1)** |
|---|---|---|---|
| **English** | 65,076 | 30,076 | 35,000 |
| **Bengali** | 14,537 | 4,537 | 10,000 |

First, the Bengali and English news dataset was split by using ***sklearn*** and its ***train_test_split*** library and used stratify parameter which maintains the ratio of y. The training set had a 70 percent splitting ratio, while the validation set had a 30 percent splitting ratio. However, the validation 30% was again split into 70% and 30% where 70% was for validation and 30% was the test dataset. Similarly, The English dataset was split. Finally, two English and Bengali splitting dataset was merged.

Table 6: Data distribution after splitting

| **Class** | **Training** | **Validation** | **Test** |
|---|---|---|---|
| **False (0)** | 24226 | 9444 | 3115 |
| **True (1)** | 31484 | 7269 | 4049 |

After slitting the dataset, it looked like as Table 6. The ratio of splitting was equal to every set.

## 5.6 Two datasets method

As the result found that, the news has lots of stop words and it has effective role on the news. Using lexical analysis, I found that the real news has more punctuation than fake news and also stop words have also same issues both news. The fake news reporter uses more stop words and has it an important role to create the model? So, I create two datasets where one dataset was created with stop words and another was created without stop words.

# 6

# Hyperparameter and Finetuning tuning Model

Fine-tuning is a process in which the model has been trained for a certain task, and then the adjustment or adjustment of the model is performed on a second similar task (Deeplizard, 2021). At first, it is preferable to train on a big amount of data with a pre-training model. Train the model with smaller data set, and this method is called model fine-tunning (Analyticsvidhya, 2021).

Different Fine-Tuning Techniques have different mechanism. Below three mechanisms are described:

Train the whole architecture - Also, using the provided dataset, train the entire pre-training model and deliver softmax level output. With this, the error is optimized throughout the architecture and the model is updated based on a new data set with pre-trained weights. Some level ships are logged by others, another way is to train them using a pre-training model. (Deeplizard, 2021).

Partially train some layers while freezing others - Pre-training your model can be accomplished in a variety of ways. It has the potential to lighten the model's original frozen layer, while just the upper layer is recycled. In the following step, try to figure out the number of frozen levels vs those that will be trained.

Freeze all the architecture - The model layers are freeze and connect a few layers of its own neural network and build this new model. Note that only the attached diaper weight model is updated during training.

## 6.1 General setup

The pre-trained model is applied to deep learning structures using Python (Rosam 1995) and PyTorch. The huggingface library (Wolf, et al., 2019), which provides a number of SOTA models and prepares them for usage by NLPs, is at the center of the evaluation. Huggingface's run_glue.py script is built by the code (Hugging Face, 2021). The implementation uses a pre-trained model and can load this model and apply it to specific tasks. For architecture, the default implementation is used. mBERT and XLMRoberta come with a variety of pre-trained model variants. Table 7 shows the *huggingface* versions used for each model.

Table 7: Used transformer model from the huggingface pipeline

| Model | Pretrained Version | Task Version |
|---|---|---|
| BERT | bert-base-multilingual-uncased | BertModel |
| XLMRoberta | xlm-roberta-base | XLMRobertaModel |

English data, lowercase model, case sensitive text corpus selection pre-learned. The evaluating of sequence classification is used for all models. It means that an additional softmax classification hierarchy has been added to perform the classification process extra to the model's architecture itself.

## 6.2 BERT multilingual base model (uncased)

Pretrained top 102 language models with the largest Wikipedia using a camouflage language modeling (MLM) purpose. This paper has introduced the model and first released in this repository. This model is uncased: There is no distinction between english (lower case) and English (upper case). (Devlin, et al., 2018).

### 6.2.1 Model description

BERT is a transformer model that is pre-trained in a large corpus of multilingual data in a self-monitoring method. This means that it is an automated process of generating input and labels from texts that have only been pre-trained in raw texts without human labeling in any way (that's why it can use a lot of publicly available data) (Devlin, et al., 2018). More specifically, it was pre-trained with two goals:

### 6.2.2 Masked Language Modeling (MLM)

The model chooses a sentence at random from the input that masks 15% of the words, then runs the complete masked sentence and predicts the masked words. This differs from the traditional endogenous neural network (RNN), which usually sees one word at a time, or from self-contained models such as the GPT, which masks tokens internally in the future. This enables the model to learn the dual representation of the sentence (Devlin, et al., 2018).

### 6.2.3 Next sentence prediction (NSP)

Two masked sentences as input being pre-trained connected according to the model. Sometimes it matches adjacent sentences in the original text, sometimes it doesn't. Then the model should predict whether two sentences continue with each other.

In this way, the model has learned the internal representation of languages in the training organization, which can be used to extract functions useful for downstream work: for example, if you have a data set with labeled sentences, you can train using functions from BERT-models generated as input. Standard classifier (Devlin, et al., 2018).

## 6.3 xlm-roberta-base

The XLM Roberta model (Conneau, et al., 2020) was proposed in "Unsupervised Cross-Lingual Representation Learning at Scale" by Alexis Konyu, Karthikeya Khandelwal, Naman Goel, Visrav Choudhury, Guillaume Wenzac, Francisco Guzman, Edouard Grave, Myel Ott, Luc Zettelmoyer and Veselin Stoyanov. It is based on the RoBERta model published by Facebook in 2019. It makes the filter general crawl data 2.5 TB training a large, multilingual language model. Details described at 3.8 XLM-Roberta

## 6.4 Training the Model

The training program for models is described in this section.

### 6.4.1 Training Data and Batching

The training, validation, testing data loader was developed by tensor tokenized data. The batch size was set 8 for training and validation.

### 6.4.2 Hardware and Schedule

The whole code was run in Google Colab pro with GPU. Taking advantages from Colab, the model was trained with `Tesla V100-SXM2-16GB`. It took roughly 0.4 seconds to complete each training phase. The models were trained for a total of 551 steps, or 10 hours.

### 6.4.3 Optimizer

Adam optimizer (Rajpurkar, et al., 2016) was used with β1 = 0.9, β2 = 0.98 and = 10-9.

lrate = 2e-5 model, min (step_num-0.5, step_num, warmup_steps- 0)

### 6.4.4 Early-stopping

Overcome the overfitting issue, the early-stopping technique was used. The early-stopping used to reduce overfitting problem. The parameter of early-stopping "patience" set as 2 that means if the validation loss decreases continuously 2 times the model training will be stopped and save the model at that point.

### 6.4.5 Freeze the model

This work was fine-tuned using three different versions. All layers except the last projection and classification layer are frozen during the first freeze run. The second pass of No Freeze does not use any freezing, so all parameters are updated during fine-tuning, and the final pass of Freeze Embed is done with the embedding layer frozen. This enables models to be evaluated in terms of both language and semantic capabilities.

## 6.5 Long Text Tokenizing

The Bert and RoBerta NLP model only accept 512 tokens. It is a downstream for any text classification problem. How do we solve this issue? We have two methods: 1. Use specific token size for entire dataset, 2. Use model parameter truncate which allow to accept first 512 tokens. Except these two methods, we can use chunking for every sentence.

Example, A paragraph with 1436 tokens could be converted into three chunks, first and second chunk have 510 tokens and third chunk have rest of the 416 tokens. With this chunking method can handle any length size of paragraph.

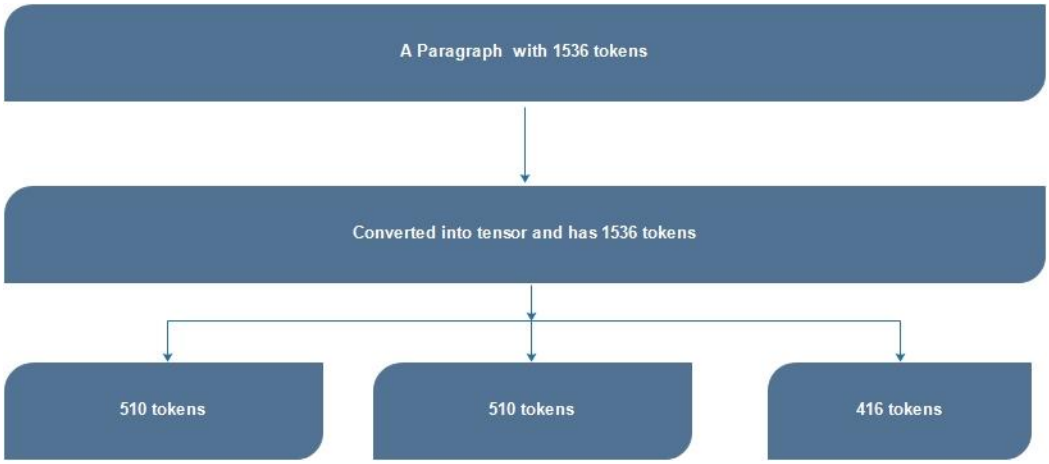



Figure 23: Chunking of long text, e.g. by Own illustration

### 6.5.1 Long text and target variable

Above it is shown how do we handle long text but the problem is about target variable. Our fake news detection dataset has label as target variable where it has two form “1” it refers the news is true and “0” refers the news is false. So, it is a classification problem. With long text chunking, I have one main news that have multiple chunk news. So, I decided that the main news target variable will also be shared with multiple chunk news. Example:

Suppose, we have a long text and using tokenizer, we create ABC tensor and find 1536 token. Besides we have target variable “0” which indicates the text is false. Now split the text token into three chunks (A, B, C) because each chunk doesn’t have more than 510 tokens. Additionally, we add special token [CLS], [SEP] and padding. So, each token size must be 512.

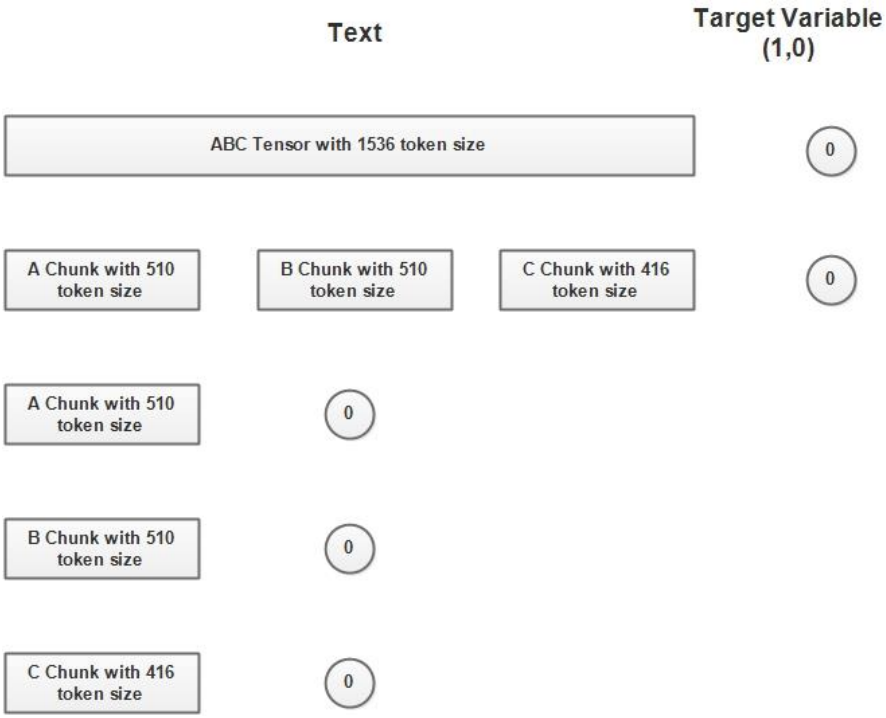



Figure 24: Chunking with targeted variable. (e.g. by Own illustration)

Finally, I share the targeted variable to each token. In describe, A, B chunk have 512 tokens which have two special tokens [CLS], [SEP] and same targeted variable 0. The last chunk C has also 512 tokens but it has extra padding to fill 512 tokens and also same targeted variable 0. Normally, it has 416 tokens.

### 6.5.2 Long text and target variable handling architecture

Below, I have a flow chart of the long text and target variable handling architecture:

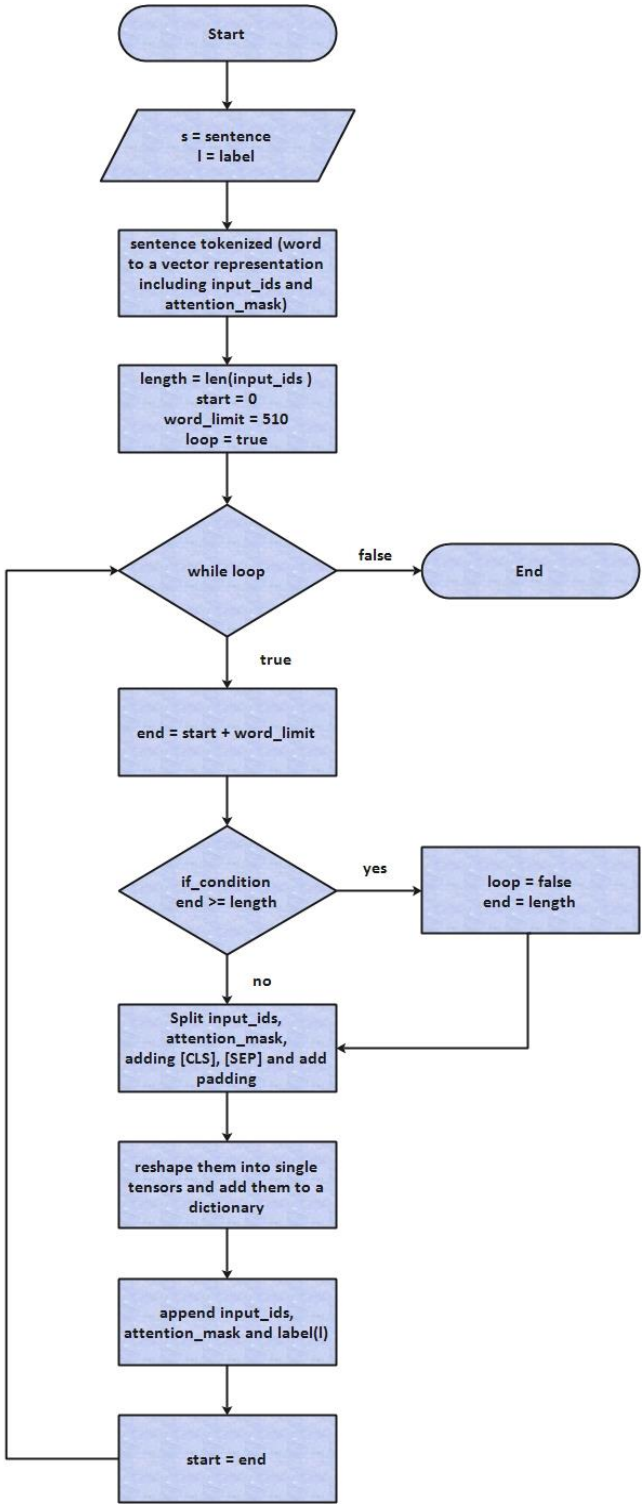



Figure 25: Flow chart of the long text and target variable handling architecture. (e.g. by Own illustration)

First, I aim for a long text as a sentence (s) and variable as a label (l). The next thing I need to do is launch the tokenizer. All in all, I am going to use the PyTorch and HuggingFace Transformer libraries.

1. Tokenization is the process of converting a text into a list of locations where a word should be placed in an embedding array as a vector representation.

2. I am utilizing the tokenizers encode_plus strategy to make tokens from "s" strings however I change the encode_plus technique to not play out any special token, truncation or padding. This generates a dictionary with three key values, input_id, token_type_id, and attention_mask, although token_type_id is ignored.

3. I build a loop where it controls the token chunking and label(l). Before starting the *while* loop, I set loop as true and three variable where length have total input_ids token, start = 0 and word_limit=510 which means I set the token split size 510.

4. The beginning of the *while* loop set as true, so it automatically executes to next steps. The next step, end variable is sum of start and word_limit. So, end= 510

5. Here I use *if condition* where the end is greater or equal to length.

If the condition is false then,

a) Now I need to break the tensor into piece of close to 510 tokens. I pick 510 as opposed to 512 because remain two spots extra. The remain two spot is filled by the token [CLS] and the [SEP] tokens. Both input_ids and attention_mask tensors are split using the split approach. Note that I should add padding if any piece won't fulfill the tensor size of 512
b) To construct an embedding array dictionary, modify the datatype of input_ids and attention_mask. The input_ids converted into long datatype and attention_mask into int datatype.
c) From dictionary, I separately create input_ids and attention_mask where these two are appended from reshape dictionary and also convert them into tensor. Besides the label is appended to a new label variable where every chunk has same label.
d) Now change the value of variable start with end and again the loop starts with new value variable until the *while* loop get false

e) If the condition is yes then,

First change the loop as false. The variable end updated with length where end have now total token size. Then loop as false and updated variable pass to the token splitting part and continue from a to *d*. This time loop set as false, so the *while* loop will stop.

## 6.6 Evaluation metric

During the classification training, evaluation matrices play an important role in achieving the best classifier. Therefore, an appropriate evaluation index is an important key to obtain an optimal and biased selective classifier. (Hossin, et al., 2015).

Threshold, probability, and ranking metrics are the three categories of evaluation metrics (Mizil & A, 2004). Each of these matrix's type evaluates classifiers with different purposes. In addition, the matrices are a method of clustering in all of these types of scales, where a single point value of the overall performance is used.

To decide the speculation limit of the prepared classifier is utilized by the evaluation metrics. For this situation, the evaluation metric is utilized to measure and summarize the value of the training classifier when testing with unknown data. The norm of precision or error rate is one of the most well-known features practically speaking to decide the capacity of the classifier speculation used by many researchers (Mizil & A, 2004).

The fake news dataset are a classification problem and dataset are classified as Unrelated. When making irrelevant predictions, the evaluation measure that basically provides genuine positive forecasts on the underlying slanting of the illusion matrix can be the model of choice because these predictions are accurate in most cases.

With 56.54% of training instances when irradiated with only two related classes are classified as true news or "1" and 43.46.7% are classified as fake news or "0" label. In reality, the percentage ratio between two classes is big because of sentence chunking. Although many news articles may reflect reality come from different sources and a claim-off argument, it is vital to correctly predict “true” and “false”, classes. If these unequal distributions are not considered in the evaluation metrics, the model can only be predicted immediately after determining the corresponding part.

Therefore, it is proposed to consider the macro-mean F1 score as an evaluation metric Hanselowski et al. It is defined as

$$F_1 - m = \frac{1}{K}\sum_{k=1}^{K} 2 * \frac{PR.RE}{PR+RE} = \frac{1}{K}\sum_{k=1}^{K} \frac{2TP}{2TP+FP+FN} \text{…….. …... (8)}$$

Here, *K* is number of classes. For our cases *K = 2*. Every class's true positive (TP), false positive (FP), and false negative (FN) values are determined separately where true positive *(TP)* alludes to when it is, in fact true, how frequently it predicts true, false positive *(FP)* alludes to when it is, in fact false, how frequently it predicts true and false negative *(FN)* refers to when it is actually true, how often it predicts false. Precision *(PR)* is defined as $PR = \frac{TP}{TP+FP}$ where it informs us how many of the accurately predicted scenarios were in fact true and recall *(RE)* as $RE = \frac{TP}{TP+FN}$ where it tells us how many of the actual true cases, I was able to predict correctly with our model. The F1-score is a calculated average precision and recall score, and so it gives a combined idea of the value of these two features. It makes the most when the exact recall is equal.

Another metric is to find accuracy,

$$ACC = \frac{TP+TN}{TP+TN+FP+FN} \text{ …….. (9)}$$

Here TN refers true negative where it's actually false, how often does it predict false and others are same as $F_1$ metric (Goodfellow, 2018).

## 6.7 Hyperparameter Tuning

The finetuning model was evaluated with several parameter to get better result. All parameter tuning was set manually and checked all results. Such as batch size, freeze the layers, learning rate and drop out. The hyperparameter tuning approaches are described below and Annex section.

Both model m-BERT and xlm-RoBERTa model perfectly run with batch size 8. The bigger batch size needs bigger GPU system. With various learning rates, the accuracy was tested (1e-05, 2e-05, 3e-05, 4e-05). The m-BERT model with 2e-3 learning rate has good accuracy with 92.54% and 3e-3 learning has also pretty good accuracy 91.84%. Other learning rates have the accuracy below 70%. So, the m-BERT model trained with learning rate 2e-3 and batch size 8.

Table 8: Overview the results of different learning rate (LR) and batch size.

| | m-BERT | xlm-RoBERTa | Batch Size |
|---|---|---|---|
| Learning rate | Accuracy | Accuracy | 8 |
| 1e-3 | 63.69 | 63.58 | 8 |
| 2e-3 | 92.54 | 94 | 8 |
| 3e-3 | 91.84 | 69.38 | 8 |
| 4e-3 | 69 | 77.36 | 8 |

The xlm-RoBERTa model has also same better accuracy 94% with learning rate 2e-3 and batch size 8. The second-best accuracy found with learning rate 4e-3 and accuracy is 77.36%.

# 7

# Model Result

The results of m-BERT and XLM-RoBERTa models are shown below Table 9. Each model was trained and evaluated by two sets of datasets. One was considering the word stop and the other was not considering using the word stop and even the word lemmatization.

From the Table 9, the results of m-BERT with stop words, Freeze Embed technique has good $F_1$-score with 90% where the model overall accuracy is 93.42%. On the other side, No Freeze has the lowest $F_1$-score of both score and overall accuracy.

Table 9: m-BERT Performance with different freeze techniques

| Dataset | With stop words | | |
|---|---|---|---|
| m-BERT | Freezing technique | | |
| Metric | Freeze | No Freeze | Freeze Embed |
| Accuracy | 92.21 | 91.32 | 93.42 |
| $F_1$-score | 89 | 87 | 90 |

| Dataset | Without stop words | | |
|---|---|---|---|
| m-BERT | Freezing technique | | |
| Metric | Freeze | No Freeze | Freeze Embed |
| Accuracy | 92.69 | 92.23 | 92.4 |
| $F_1$-score | 90 | 89 | 89 |

From Table 9, the results without stop words and word lemmatization, the Freezing technique of Freeze has the best accuracy as F1-score and overall accuracy with 90% and 92.69% respectively. The No Freeze and Freeze Embed techniques have the same accuracy of $F_1$-score 89%.

So, overall, with stop words and the Freeze Embed technique, the m-BERT model results are slightly better than with other freezing techniques. This model was trained with three epochs and batch size 8. Three epochs and a batch size of eight were used to train this model. 6.4.4 Early-stopping approaches computed a three-epoch epoch size.


| epoch | Training Loss | Valid. Loss | Valid. Accur. | training time | validation time |
|---|---|---|---|---|---|
| 1 | 0.24 | 0.18 | 93.10 | 1:47:04 | 0:08:39 |
| 2 | 0.19 | 0.19 | 93.43 | 1:46:53 | 0:08:39 |
| 3 | 0.20 | 0.20 | 93.57 | 1:47:01 | 0:08:39 |

Figure 26: m-BERT Data by epoch for training and validation loss graphs

The training and validation loss are depicted in the diagram above, with the training loss starting at 0.24 and decreasing by 0.19 at epoch 2, but increasing at the third epoch iteration. Besides, the validation loss was 0.18 at the first epoch and it continuously increased through the third epochs. The model took 1 hour and 45 minutes to run one epoch and 8.39 minutes to evaluate.

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.95 | 0.86 | 0.90 | 8864 |
| 1 | 0.93 | 0.98 | 0.95 | 16095 |
| accuracy | | | 0.93 | 24959 |
| macro avg | 0.94 | 0.92 | 0.93 | 24959 |
| weighted avg | 0.94 | 0.93 | 0.93 | 24959 |

Figure 27: m-BERT classification Report

The classification report was introduced for the model to get more accurate results. The model has a better F1 score for true news than for false news.

According to the results of xlm-RoBERTa with stop words, the Freeze Embed technique has an accuracy of 94.55% and an F1-score of 92% Table 10. It is the best combined result between all freeze techniques. On the other hand, no freeze and Freeze Embed techniques have the same accuracy with 94.15% and F1-score 91.35 and 91.45% respectively.

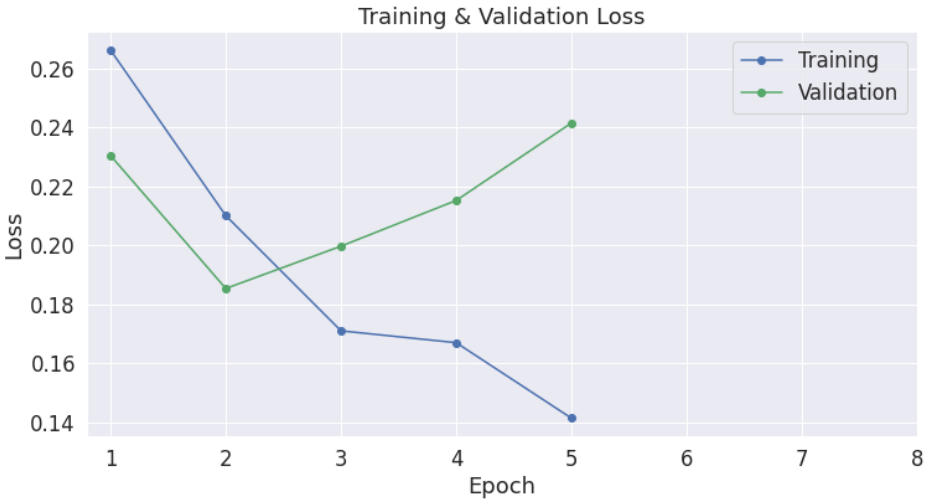
Table 10: xlm-RoBERTa Performance with different freeze techniques

| Dataset | With stop words | | |
|---|---|---|---|
| xlm-RoBERTa | Freezing technique | | |
| Metric | Freeze | No Freeze | Freeze Embed |
| Accuracy | 94.55 | 94.15 | 94.15 |
| $F_1$-score | 92.12 | 91.35 | 91.45 |

| Dataset | Without stop words | | |
|---|---|---|---|
| xlm-RoBERTa | Freezing technique | | |
| Metric | Freeze | No Freeze | Freeze Embed |
| Accuracy | 92.36 | 53.45 | 92.36 |
| $F_1$-score | 89.25 | 37.88 | 89.65 |

According to Table 10, the freezing techniques of freeze and freeze embed have the same accuracy with 92.36 percent, F1-score of 89.25 percent, and 89.65 percent, respectively. The No Freeze technique has the lowest overall accuracy and F1-score, and it outperforms both the m-BERT and xlm-RoBERTa models.

Overall, the xlm-RoBERTa model performs slightly better than other freezing techniques when using the stop words dataset and the freeze technique. The model was trained using five epochs and a batch size of eight. The model was saved, however, when the training model had the lowest validation loss. As a result, the training model had the lowest validation loss at epoch 3.




| epoch | Training Loss | Valid. Loss | Valid. Accur. | training time | validation time |
|---|---|---|---|---|---|
| 1 | 0.27 | 0.23 | 93.88 | 1:58:22 | 0:10:55 |
| 2 | 0.21 | 0.19 | 94.65 | 1:58:15 | 0:10:55 |
| 3 | 0.17 | 0.20 | 94.99 | 1:58:12 | 0:10:56 |
| 4 | 0.17 | 0.22 | 95.16 | 1:58:08 | 0:10:55 |
| 5 | 0.14 | 0.24 | 95.34 | 1:58:02 | 0:10:55 |

Figure 28: xlm-RoBERTa Training loss and validation loss graph, data by each epoch

The above figure depicts the training and validation loss, with the training loss starting at 0.27 and decreasing by 0.21 at epoch 2 and continuing to decrease with each epoch iteration. Furthermore, the validation loss was 0.23 at the first epoch and decreased steadily through the third epochs. Each epoch took nearly two hours to train and eleven minutes to validate.

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.96 | 0.89 | 0.92 | 5614 |
| 1 | 0.94 | 0.98 | 0.96 | 9577 |
| accuracy | | | 0.95 | 15191 |
| macro avg | 0.95 | 0.93 | 0.94 | 15191 |
| weighted avg | 0.95 | 0.95 | 0.95 | 15191 |

Figure 29: xlm-RoBERTa classification Report

The classification report was introduced for the model to get more accurate results. The model has a better F1 score for true news than for false news.

# 8

# Conclusion

The goal of the thesis is to examine and demonstrate unaided learning in the area of spotting fake news. As a result, fake news is characterized as clearly false text that is disseminated for malevolent objectives in the hopes of being useful. Confirmation bias, chamber of commerce echoes, and general social needs all contribute to people's sensitivity to fake news. The expectation theory explains how people perceive the data ecosystem. In expectation theory, publishers and customers aim to get the most utility out of their decisions. Fake news may be particularly successful when publishers are affected by psychological and sociological factors rather than by focusing on fair information about short-term benefits to increase the usefulness of readers and consumers.

Many online social media platforms, including YouTube, are now used for journalism. It is happening as a result of people bumming the internet all over the world. Almost everyone owns a smartphone these days. According to Statista, global smartphone subscriptions have surpassed 6 billion and are expected to rise by hundreds of millions in the coming years. As a result, some yellow journalism is used on these online platforms to publish news that has not been verified. Even those journalists do not have a background in journalism. The yellow journalists are always on the lookout for trending and viral news. They do not publish any content-based news. Users or viewers are more interested in this type of news than others. Researchers are experimenting with new technology to detect false news automatically rather than manually checking every piece of news, and they have already had great success.

The resulting stance identification job was evaluated using two common pretrained NLP models, m-BERT and xlm-RoBERTa. With the release of m-BERT in 2019, another time of commonly effective exchange learning in NLP starts. The main pre-prepared NLP model that used the Transformer concept and added bidirectional settings without repeat was the m-BERT. The xlm-RoBERTa has also add bidirectional context. Each model is effective in multilingual approach. The final results found that the transformer m-BERT and xlm-RoBERTa models are slightly better performed with stopwords, it happens because of language lexical diversity. The news authors have word diversity that may affect news classification. Fine-tuning approach also works almost same for both models. The performance between two model doesn't have that much difference.

I had a small dataset for this experiment. The proportion of English and Bengali datasets was also skewed. Datasets are essential for producing better results. The false/true ratio was unbalanced, as were the effects on the training model. More balance data will be collected in the future. Another work will be added as more different languages are supported by the mBERT and xlm-RoBERTa transformers, which support over 100 languages, it could be effective to research with other languages like German, French, Spanish etc.

Finally, different types of techniques and modeling approaches utilized to combat fake news will only be effective if they adapt to the actual word process. Though multilingual model approach research is complicated but it has achieved tremendous success current times. Artificial intelligence (AI) tools for detecting false news may only be viewed as a brick in the wall to lessen the actual negative impacts of fake news.

# References

Ahmed & al, e., 2017. Detecting opinion spams and fake news using text classification.

Analyticsvidhya, 2021. *Transfer Learning for NLP: Fine-Tuning BERT for Text Classification.* [Online]
Available at: https://www.analyticsvidhya.com/blog/2020/07/transfer-learning-for-nlp-fine-tuning-bert-for-text-classification/

Bal, A. & al, e., 2019. Comparative Performance of Machine Learning Algorithms for Fake News Detection. *Advances in Computing and Data Sciences. ICACDS,* p. vol 1046.

Bowman, S. R., Angeli, G., Potts, C. & Manning, C. D., 2015. A large annotated corpus for learning natural language inference. In: s.l.:Association for Computational Linguistics, p. 632–642.

Britz, D., Goldie, A. & Le, M.-T. L. a. Q. V., 2017. Massive exploration of neural machine translation architectures.. pp. CoRR, abs/1703.03906,.

Çelik, Ö. & Altunaydin, S. S., 2018. A Research on Machine Learning Methods and Its Applications. *Journal of Educational Technology & Online Learning,* p. Volume 1 │Issue 3│.

Chang, J. D. a., 2019. Open Sourcing BERT: State-of-the-Art Pretraining for Natural Language Processing. *Google AI Blog.*

Cho, K. et al., 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. pp. CoRR, abs/1406.1078,.

Coats, S., n.d. Comparing Word Frequencies and Lexical Diversity with the ZipfExplorer Tool.

Conneau, A., Khandelwal, K. & Goyal, N., 2020. Unsupervised Cross-lingual Representation Learning at Scale. *arXiv:1911.02116v2 [cs.CL].*

Courville, I. G. a. Y. B. a. A., 2016. Deep Learning. In: s.l.:MIT Press, pp. 321,397.

Darnton, R., 2017. The True History of Fake News.. pp. https://www.nybooks.com/daily/2017/02/13/the-true-history-of-fake-news/ .

Dataset., B. F. n., 2020. *Kaggle.* [Online]
Available at: https://www.kaggle.com/cryptexcode/banfakenews

Dataset., E. F. n., 2019. *Kaggle.* [Online]
Available at: https://www.kaggle.com/c/fake-news/data

Deeplizard, 2021. *Fine-Tuning A Neural Network Explained.* [Online]
Available at: https://deeplizard.com/learn/video/5T-iXNNiwIs

Devlin, J., Chang, M.-W., Lee, K. & Toutanova, K., 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. p. arXiv:1810.04805.

D, M., B, R., N, C. & P, D., 2004. Lexical diversity and language development: Quantification and assessment.. *New York: Palgrave Macmillan;*.

Dzmitry Bahdanau, K. C. a. Y. B., 2014. Neural machine translation by jointly learning to align and translate. pp. CoRR, abs/1409.0473,.

Ermakova et al, 2019. Multilingual Fake News Detection with Satire on Vaccination Topic.

Gillioz, A., 2019. Overview of the Transformer-based Models for NLP Tasks" Proceedings of the Federated Conference on Computer Science and Information Systems. *University of Neuchâtel Neuchâtel, Switzerland.*, pp. pp. 179–183 DOI: 10.15439/2020F20 ISSN 2300-5.

Goodfellow, I. Y. B. a. A. C., 2018. Deep Learning. MIT Press. Hanselowski, Andreas et al. (2018). "A Retrospective Analysis of the Fake News Challenge Stance Detection Task. *In: CoRR abs/1806.05180.*

Graves., A., 2013. Generating sequences with recurrent neural networks. pp. .arXiv preprint arXiv:1308.0850,.

Guderlei, M., Heumann, P. D. C. & Aßenmacher, M., 2020. Fake News Detection, Evaluating Unsupervised Representation Learning for Detecting Stances of Fake News. *Ludwig-Maximilians Universität München.*

Guibon, G., Ermakova, L. & Seffih, H., 2020. Multilingual Fake News Detection with Satire.

Güneren, H., 2015. Destek Vektör Makineleri Kullanarak Gömülü Sistem Üzerinde Yüz Tanıma Uygulaması. *Yüksek lisans tezi, Yıldız Teknik Üniversitesi, İstanbul..*

Hastie, T., Tibshirani, R. & Friedman, J., 2009. The Element Statistical Learning. In: s.l.:Springer, pp. 417, 463.

Hendrycks, D. a. K. G., 2016. Bridging Nonlinearities and Stochastic Regularizers with Gaussian Error Linear Units. *In: CoRR abs/1606.08415.*

Hossain et al, 2020. BanFakeNews: A Dataset for Detecting Fake News in Bangla.

Hossin, Sulaiman, M. a. & M.N, 2015. A Review on Evaluation Metrics for Data Classification Evaluations.. *IJDKP,* pp. Vol.5, No.2.

Hugging Face, 2021. *BERT base model (uncased).* [Online]
Available at: https://huggingface.co/bert-base-uncased

Islam et al, 2020. Bengali Fake News Detection. *IEEE 10th International Conference on Intelligent Systems (IS).*

Johnson, W., 1944. Studies in language behavior: 1. A program of research. Psychological Monographs. p. 56:1–15.

JP, D. & M, P., 2002. The persuasion handbook: Develppments in theory and practice. *Thousand Oaks, CA: Sage;*.

Kaiming He, X. Z. S. R. a. J. S., 2016. Deep residual learning for image recognition.. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,*, pp. pages 770–778, .

Kalyanathaya, K. P. & Rajesh, D. A. a. P., 2019. Advances in Natural Language Processing – A Survey of Current Research Trends, Development Tools and Industry Applications.

Kingma, D. P. a. J. B., 2014. Adam: A Method for Stochastic Optimization. *In: ArXiv 1412.6980.*

Knox, S. W., 2018. Machine Learning a Concise Introduction. In: s.l.:Wiley.

Koizumi, R., 2012. Relationships Between Text Length and Lexical Diversity Measures: Can We Use Short Texts of Less than 100 Tokens?. pp. Volume 1, Issue 1.

Liu, Y., 2019. Roberta: A robustly optimized BERT pretraining approach.. p. arXiv preprint arXiv:1907.11692..

Liu, Y. et al., 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach.

Maind, M. S. B. & Wanka, M. P., n.d. Research Paper on Basic of Artificial Neural Network. *International Journal on Recent and Innovation Trends in Computing (Communication ISSN: 2321-8169 96 – 100)*, p. Volume: 2 Issue: 1.

McClure, L., 2017. How to tell fake news from real news. January.pp. blog.ed.ted.com/2017/01/12/how-to-tell-fake-news-from-real-news/.

Mizil, N. & A, R. C. a., 2004. Data mining in metric space: an empirical analysis of supervised learning performance criteria. *in Proc. of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '04)*, pp. 69-78.

Nasir et al, 2021. Fake news detection: A hybrid CNN-RNN based deep learning approach. *International Journal of Information Management Data Insights.*

Negnevitsky, M., 2009. Artificial Intelligence: A Guide to Intelligent Systems.. *Addison Wesley: Longman Publishing.*.

Nicolas Be´rube´, M. S.-M. P. M. ,. V. L., n.d. Words by the tail: Assessing lexical diversity in scholarly titles using frequency-rank distribution tail fits.

Ou, X. & Li, H., n.d. XLM-RoBERTa for Multi-language Sentiment Analysis. *School of Information Science and Engineering, Yunnan University, Kunming, 650500, Yunnan, P.R. China.*

Pngwing, 2021. *Pngwing.* [Online]
Available at: https://www.pngwing.com/en/free-png-yzjfj
[Accessed 03 07 2021].

Posetti et al, 2018. A Short Guide to the History of'fake News' and Disinformation.. *International Center for Journalists.,* pp. https://www.icfj.org/sites/default/files/2018-07/A% .

Qiao, Y., n.d. A Language-Based Approach to Fake News Detection Through Interpretable Features and BRNN.

Rajpurkar, P., Zhang, J., Lopyrev, K. & Liang, a. P., 2016. Squad: 100,000+ questions for machine comprehension of text.. *In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing,,* p. pages 2383–2392.

Ruchansky, N., 2017. A Hybrid Deep Model for Fake News Detection.

Shah & Virendra, P., 2020. Multimodal fake news detection using a Cultural Algorithm with situational and normative knowledge. *University of Windsor Published.*

Shen & al, H. e., 2017. Discovering social spammers from multiple views. *Neurocomputing,* p. 225:49–57.

Tandoc et al., 2017. DEFINING "FAKE NEWS" A typology of scholarly definitions. *Digital Journalism,* pp. Vol. 2, Issue 2, pp. 137-153,.

Vaswani, A. et al., 2018. Attention Is All You Need.

Vedantam, V. K., Analytics, A., Mahindra, T. & Copenhagen, 2020. The Survey: Advances in Natural Language Processing using Deep Learning.

Wang, X. et al., 2017. Robust visual tracking via multiscale deep sparse networks. *Spie digital library.*

Waweru, J., 2019. "Understanding Fake News". *Article in International Journal of Scientific and Research Publications (IJSRP).*

Wolf, T., Sanh, V., Chaumond, J. & Delangue, C., 2019. TransferTransfo: A Transfer Learning Approach for Neural Network Based Conversational Agents.

Wu, Y. e. a., 2016. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *In: CoRR abs/1609.08144.*.

# Annex

## A. Additional Evaluation Results

m-BERT model Finetuning with various Freeze techniques:

With Stop words and Freeze Technique:

| epoch | Training Loss | Valid. Loss | Valid. Accur. | training time | validation time |
|---|---|---|---|---|---|
| 1 | 0.28 | 0.21 | 92.46 | 3:45:50 | 0:22:21 |
| 2 | 0.23 | 0.22 | 92.73 | 3:45:32 | 0:21:55 |
| 3 | 0.23 | 0.28 | 92.72 | 3:43:10 | 0:21:58 |

Figure A. 1: m-BERT with Stop words and Freeze Technique each epoch details


Figure A. 2: m-BERT with Stop words and Freeze Technique training and validation loss details

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.94 | 0.85 | 0.89 | 5469 |
| 1 | 0.92 | 0.97 | 0.94 | 9359 |
| accuracy | | | 0.92 | 14828 |
| macro avg | 0.93 | 0.91 | 0.92 | 14828 |
| weighted avg | 0.93 | 0.92 | 0.92 | 14828 |

Figure A. 3: m-BERT with Stop words and Freeze Technique Classification report

With Stop words and No Freeze Technique:

| epoch | Training Loss | Valid. Loss | Valid. Accur. | training time | validation time |
|---|---|---|---|---|---|
| 1 | 0.24 | 0.18 | 92.58 | 1:50:47 | 0:10:37 |
| 2 | 0.19 | 0.21 | 92.76 | 1:50:41 | 0:10:37 |
| 3 | 0.18 | 0.21 | 91.45 | 1:50:36 | 0:10:37 |

Figure A. 4: m-BERT with Stop words and No Freeze Technique each epoch details


Figure A. 5: m-BERT with Stop words and No Freeze Technique training and validation loss details

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.95 | 0.80 | 0.87 | 5469 |
| 1 | 0.90 | 0.98 | 0.93 | 9359 |
| accuracy | | | 0.91 | 14828 |
| macro avg | 0.93 | 0.89 | 0.90 | 14828 |
| weighted avg | 0.92 | 0.91 | 0.91 | 14828 |

Figure A. 6: m-BERT with Stop words and No Freeze Technique Classification report

Without Stop words and Freeze Technique:

| epoch | Training Loss | Valid. Loss | Valid. Accur. | training time | validation time |
|---|---|---|---|---|---|
| 1 | 0.29 | 0.23 | 91.75 | 2:47:54 | 0:16:57 |
| 2 | 0.21 | 0.22 | 92.32 | 2:47:30 | 0:16:50 |
| 3 | 0.18 | 0.23 | 92.32 | 2:47:26 | 0:16:47 |
| 4 | 0.19 | 0.34 | 92.57 | 2:47:29 | 0:16:41 |

Figure A. 7: m-BERT without Stop words and Freeze Technique each epoch details


Figure A. 8: m-BERT without Stop words and Freeze Technique training and validation loss details

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.96 | 0.85 | 0.90 | 4297 |
| 1 | 0.91 | 0.98 | 0.94 | 6927 |
| accuracy | | | 0.93 | 11224 |
| macro avg | 0.93 | 0.91 | 0.92 | 11224 |
| weighted avg | 0.93 | 0.93 | 0.93 | 11224 |

Figure A. 9: m-BERT without Stop words and Freeze Technique Classification report

Without Stop words and No Freeze Technique:

| epoch | Training Loss | Valid. Loss | Valid. Accur. | training time | validation time |
|---|---|---|---|---|---|
| **1** | 0.30 | 0.29 | 91.81 | 1:24:17 | 0:08:09 |
| **2** | 0.23 | 0.23 | 92.14 | 1:24:06 | 0:08:09 |
| **3** | 0.24 | 0.29 | 92.22 | 1:24:11 | 0:08:09 |
| **4** | 0.23 | 0.31 | 92.52 | 1:24:12 | 0:08:09 |

Figure A. 10: m-BERT without Stop words and No Freeze Technique each epoch details

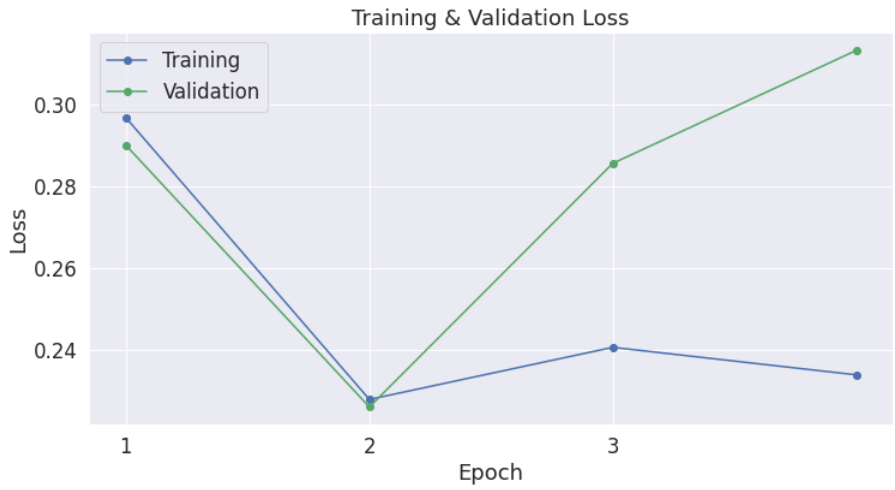



Figure A. 11: m-BERT without Stop words and No Freeze Technique training and validation loss details

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.97 | 0.82 | 0.89 | 4297 |
| 1 | 0.90 | 0.99 | 0.94 | 6927 |
| accuracy | | | 0.92 | 11224 |
| macro avg | 0.94 | 0.91 | 0.92 | 11224 |
| weighted avg | 0.93 | 0.92 | 0.92 | 11224 |

Figure A. 12: m-BERT without Stop words and No Freeze Technique Classification report

Without Stop words and Freeze Embed Technique:

| epoch | Training Loss | Valid. Loss | Valid. Accur. | training time | validation time |
|---|---|---|---|---|---|
| **1** | 0.30 | 0.29 | 91.81 | 1:24:29 | 0:08:09 |
| **2** | 0.23 | 0.23 | 92.14 | 1:24:16 | 0:08:09 |
| **3** | 0.24 | 0.29 | 92.22 | 1:24:20 | 0:08:09 |
| **4** | 0.23 | 0.31 | 92.52 | 1:24:24 | 0:08:09 |

Figure A. 13: m-BERT without Stop words and Freeze Embed Technique each epoch details


Figure A. 14: m-BERT without Stop words and Freeze Embed Technique training and validation loss details

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.97 | 0.82 | 0.89 | 4297 |
| 1 | 0.90 | 0.99 | 0.94 | 6927 |
| accuracy | | | 0.92 | 11224 |
| macro avg | 0.94 | 0.91 | 0.92 | 11224 |
| weighted avg | 0.93 | 0.92 | 0.92 | 11224 |

Figure A. 15: m-BERT without Stop words and Freeze Technique Embed Classification report

xlm-RoBERTa model Finetuning with various Freeze techniques:

With Stop words and No Freeze Technique:

| epoch | Training Loss | Valid. Loss | Valid. Accur. | training time | validation time |
|---|---|---|---|---|---|
| 1 | 0.27 | 0.20 | 93.49 | 1:58:27 | 0:10:57 |
| 2 | 0.19 | 0.20 | 93.79 | 1:58:07 | 0:10:58 |
| 3 | 0.15 | 0.18 | 93.98 | 1:58:04 | 0:10:57 |
| 4 | 0.13 | 0.18 | 94.29 | 1:58:02 | 0:10:57 |

Figure A. 16: xlm-RoBERTa with Stop words and No Freeze each epoch details


Figure A. 17: xlm-RoBERTa with Stop words and No Freeze training and validation loss details graph

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.99 | 0.85 | 0.91 | 5614 |
| 1 | 0.92 | 1.00 | 0.96 | 9577 |
| accuracy | | | 0.94 | 15191 |
| macro avg | 0.96 | 0.92 | 0.94 | 15191 |
| weighted avg | 0.95 | 0.94 | 0.94 | 15191 |

Figure A. 18: xlm-RoBERTa with Stop words and No Freeze Technique Classification report

With Stop words and Freeze Embed Technique

| epoch | Training Loss | Valid. Loss | Valid. Accur. | training time | validation time |
|---|---|---|---|---|---|
| 1 | 0.27 | 0.20 | 93.49 | 1:59:42 | 0:10:56 |
| 2 | 0.19 | 0.20 | 93.79 | 1:59:24 | 0:10:56 |
| 3 | 0.15 | 0.18 | 93.98 | 1:59:19 | 0:10:56 |
| 4 | 0.13 | 0.18 | 94.29 | 1:59:17 | 0:10:56 |

Figure A. 19: xlm-RoBERTa with Stop words and Freeze Embed each epoch details


Figure A. 20: xlm-RoBERTa with Stop words and Freeze Embed training and validation loss details graph

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.99 | 0.85 | 0.91 | 5614 |
| 1 | 0.92 | 1.00 | 0.96 | 9577 |
| accuracy | | | 0.94 | 15191 |
| macro avg | 0.96 | 0.92 | 0.94 | 15191 |
| weighted avg | 0.95 | 0.94 | 0.94 | 15191 |

Figure A. 21: xlm-RoBERTa with Stop words and Freeze Embed Technique Classification report

Without Stop words and No Freeze Technique

| epoch | Training Loss | Valid. Loss | Valid. Accur. | training time | validation time |
|---|---|---|---|---|---|
| **1** | 0.38 | 0.29 | 90.81 | 1:29:53 | 0:08:18 |
| **2** | 0.26 | 0.23 | 91.89 | 1:29:37 | 0:08:18 |
| **3** | 0.21 | 0.22 | 92.38 | 1:29:34 | 0:08:18 |
| **4** | 0.20 | 0.26 | 92.48 | 1:29:37 | 0:08:18 |

Figure A. 22: xlm-RoBERTa without Stop words and No Freeze each epoch details


Figure A. 23: xlm-RoBERTa without Stop words and No Freeze training and validation loss details graph

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.41 | 0.35 | 0.37 | 4577 |
| 1 | 0.60 | 0.66 | 0.63 | 6777 |
| accuracy | | | 0.53 | 11354 |
| macro avg | 0.50 | 0.50 | 0.50 | 11354 |
| weighted avg | 0.52 | 0.53 | 0.53 | 11354 |

Figure A. 24: xlm-RoBERTa without Stop words and No Freeze Technique Classification report

Without Stop words and Freeze Embed Technique

| epoch | Training Loss | Valid. Loss | Valid. Accur. | training time | validation time |
|---|---|---|---|---|---|
| 1 | 0.38 | 0.29 | 90.81 | 1:29:19 | 0:08:15 |
| 2 | 0.26 | 0.23 | 91.89 | 1:29:04 | 0:08:15 |
| 3 | 0.21 | 0.22 | 92.38 | 1:29:00 | 0:08:15 |
| 4 | 0.20 | 0.26 | 92.48 | 1:29:03 | 0:08:15 |

Figure A. 25: xlm-RoBERTa without Stop words and Freeze Embed each epoch details


Figure A. 26: xlm-RoBERTa without Stop words and Freeze Embed training and validation loss details graph

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.95 | 0.84 | 0.89 | 4378 |
| 1 | 0.91 | 0.97 | 0.94 | 6976 |
| accuracy | | | 0.92 | 11354 |
| macro avg | 0.93 | 0.91 | 0.92 | 11354 |
| weighted avg | 0.93 | 0.92 | 0.92 | 11354 |

Figure A. 27: xlm-RoBERTa without Stop words and Freeze Embed Technique Classification report

Without Stop words and Freeze Technique

| epoch | Training Loss | Valid. Loss | Valid. Accur. | training time | validation time |
|---|---|---|---|---|---|
| 1 | 0.38 | 0.29 | 90.81 | 1:29:16 | 0:08:17 |
| 2 | 0.26 | 0.23 | 91.89 | 1:29:02 | 0:08:16 |
| 3 | 0.21 | 0.22 | 92.38 | 1:28:53 | 0:08:16 |
| 4 | 0.20 | 0.26 | 92.48 | 1:28:59 | 0:08:16 |

Figure A. 28: xlm-RoBERTa without Stop words and freeze each epoch details


Figure A. 29: xlm-RoBERTa without Stop words and Freeze training and validation loss details graph

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.95 | 0.84 | 0.89 | 4378 |
| 1 | 0.91 | 0.97 | 0.94 | 6976 |
| accuracy | | | 0.92 | 11354 |
| macro avg | 0.93 | 0.91 | 0.92 | 11354 |
| weighted avg | 0.93 | 0.92 | 0.92 | 11354 |

Figure A. 30: xlm-RoBERTa without Stop words and Freeze Technique Classification report

# B. Dataset and Code links

**Web scrapping code:**

New York Times news web scrapping:

https://colab.research.google.com/drive/1jsrZHPhQsvg41CJVr8fPxfOCOW1Cnvhc#scrollTo=vBnNV7nTxxiA

The Guardian news web scrapping:

https://colab.research.google.com/drive/16zRJWuCRzf_eZ94RTSmuy4PbQGM2eFxV#scrollTo=cQrTqvwmyfzD

English Dataset: https://drive.google.com/file/d/1KY5zaZJpF-CsImc1y4w5Pa4_W8qK7pVb/view?usp=sharing

Bengali Dataset: https://drive.google.com/file/d/1sfmcjKdiMpqryd--k5lb3EcTBTtVlcDj/view?usp=sharing

**Data Cleaning Code:**

https://colab.research.google.com/drive/1RqQhZtWP7TwXS5c5vTKTtSp2cMlmaGcH

**Linguistic Analysis:**

**Bengali dataset:**

https://colab.research.google.com/drive/1X1L1Pillbj7E3xBbRm6qOHfyUvHGRMSz#scrollTo=ncXhVBx8gv5d

**English Dataset:**

https://colab.research.google.com/drive/1VKlJ_IIrWULU-zSGH4uSn7H4x6yHQyAH

**Final Model training**

**m-BERT model**:

https://colab.research.google.com/drive/1-kkiDBjsKYJr9i15yYBeb59_FHkYTKEY

**xlm-RoBERTa Model:**

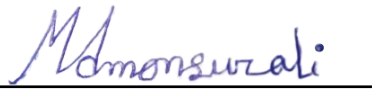
https://colab.research.google.com/drive/1CEPTkpe-hLRtGE2mcbRO-5gJ-ofqlzJE#scrollTo=HSInM1TVw4oF

# Declaration of Authenticity

I, Md Monsur Ali, thusly proclaim that the work given above is altogether my own work, achieved without the guides determined. Some other sources of data or endeavors done by others have been recognized and recorded in the reference segment. The recognizable proof of different references with connection to the assertion and extent of the work is cited; lines or parcels of sentences referred to in a real sense are set apart as citations. The work depicted here has never been distributed or submitted for assessment in the equivalent or a comparable way somewhere else. I'll keep a duplicate of this task until the Board of Examiners delivers the outcomes, which I'll make accessible on request.

Oberhausen, 17.10.2021

Place and Date



Signature: