OPINION ZERO: BIAS NEWS DETECTION MOBILE APPLICATION

PITH LAOHAVIROJANA
MAYLIN CATHERINE CERF

A SENIOR PROJECT SUBMITTED IN
PARTIAL FULFILMENT
OF THE REQUIREMENTS FOR
THE DEGREE OF BACHELOR OF SCIENCE
(COMPUTER SCIENCE)
MAHIDOL UNIVERSITY INTERNATIONAL COLLEGE
MAHIDOL UNIVERSITY
2022

Senior Project

entitled

OPINION ZERO: BIAS NEWS DETECTION MOBILE APPLICATION

was submitted to the Mahidol University International College, Mahidol University
for the degree of Bachelor of Science (Computer Science)
on
24th July 2022

.....
Pith Laohavirojana
Candidate

.....
Maylin Catherine Cerf
Candidate

.....
Dr.Sunsern Cheamanunkul
Advisor

.....
Dr. Brian J. Phillips
Chair of Science Division
Mahidol University International College
Mahidol University

.....
Dr. Boonyanit Mathayomchan
Program Director
Advisor
Bachelor of Science in Computer Science
Mahidol University International College
Mahidol University

OPINION ZERO: BIAS NEWS DETECTION MOBILE APPLICATION.

PITH LAOHAVIROJANA 6180048 1 ICCS/B

MAYLIN CATHERINE CERF 6180039 2 ICCS/B

B.Sc. (COMPUTER SCIENCE)

SENIOR PROJECT ADVISORS : DR.SUNSERN CHEAMANUNKUL, (COMPUTER SCIENCE)

ABSTRACT

In a contemporary world, man kinds has lived through the age of modern civilization defined by emerging of intelligence technologies such as super computer, machine learning and especially deep learning. The application of deep learning has been assisted and integrated in human's daily life, for instance, audio recognition, bio informatics, natural language processing and many mores. Although deep learning has contributed great benefits to the society; however, we still can extend the power of deep learning by utilizing it to tackle social issue like media bias. Media bias is the a type of bias occurring in the media. It can be done by the subjective view of journalists or news outlet. Media bias has been used to manipulate numerous significant incidents, for example, it can be used to manipulate national election or can be used to promote fake news. This has caused major spreading of false information. Therefore, In this project, we aimed to produce a deep learning learning model detecting a bias type: Red shirt and yellow shirt in the news and created a mobile application out of it.

KEY WORDS : DEEP LEARNING, MOBILE APPLICATION, BIAS DETECTION

51 pages

CONTENTS

ABSTRACT (ENGLISH) **iii**

1 Introduction **1**
1.1 Problem Summary 1
1.2 Media bias with its impacts 1
1.3 Bias News Detection Mobile Application Mitigates Media Bias 3
1.4 Overview of the application 3

2 Literature Review **4**
2.1 Political News Bias Detection using Machine learning 4
2.2 Exploration of classifying sentence bias in news articles with deep learning models 7
2.2.1 Data 9
2.2.2 Machine Learning Models 10

3 Solution and Design **11**
3.1 Overview Functionality of Our Opinion Zero Mobile Application 11
3.2 Features of Our Opinion Zero Mobile Application 11
3.2.1 Mobile Application: 11
3.2.2 Bias Detection model: 13
3.3 Prioritize features with justifications 15

4 Implementation **16**
4.1 Data Collection 16
4.2 Criteria of Bias Classification 17
4.3 Text Classification with NLP 18
4.3.1 TF-IDF 18
4.3.2 Fasttext 20
4.3.3 BERT 20
4.3.4 BERT with Siamese Model 24
4.3.5 BERT with Softmax activation function 26
4.3.6 BERT with Tanh activation function 27
4.3.7 BERT with Triplet Network 28
4.4 Result 30
4.5 Result Comparison 33
4.5.1 Performance of TF-IDF Model 34

4.5.2 Performance of FastText Model 34
4.5.3 Performance of BERT Model 34
4.5.4 Performance of BERT with Siamese Model 34
4.5.5 Performance of BERT with Softmax 35
4.5.6 Performance of BERT with Triplet Network 35
4.5.7 Potential issues 36
4.6 Tools in Building a Model 37
4.7 Mobile Application 37
4.7.1 Application Draft using Ionic and Vue-JS 38
4.8 Finished Prototype 39
4.8.1 Frontend Implementation 40
4.8.2 Backend Implementation 44

5 Conclusion and Future Work **48**
5.1 Future work and improvement on Model 48
5.2 Future work and improvement on Mobile Application 48
5.3 Conclusion 49

REFERENCES **50**

CHAPTER 1

INTRODUCTION

1.1 Problem Summary

Mass media is an innovation software medium varying from technologies to reach people on a global scale through mass communication. Various technologies such as computers, laptops, mobile phones, and tablets are now the main source to enable mass communication. In today's world where mass media are easily accessed by humans around the globe, it is undeniable that media bias is widely disputed and seen as a significant issue. The bias of the media and journalists all is when they are delivered in a way of their own interest whether it comes to choosing events to publish or even how they are reported or covered is called Media Bias. Rather than publishing from the true perspective of a reporter, the implication of persuasive or widespread bias contravenes with the authenticity of the article. The gravity of how the media bias occurs varies from country to country and is profusely altercation [9].

1.2 Media bias with its impacts

In our social construction, the media play a crucial role in shaping public perception and intuition of fundamental political and social issues. When the media characterizes the numerous events and furnishes trustworthy and reliable information regarding a variety of topics such as technology, environment and venture, it has a significant influence toward the public. Various research have revealed that the public gains information and their knowledge via the mass media.

As a result, it is critical to investigate the distortion and bias of significant topics in the media. Mass media's influence has taken the world on a global scale, it has taken various forms from real blatant injustices via News networks, internet websites and other forms of publications. People are drawn to these media because it allows them to be updated with what the world is going through. People create their own opinions and discussions thus expanding the political aspect of the modern world. The media's influence has a crucial role in political matters, as it is used as a medium to spread the power of influence or even win over an opposing political member. They use the media to mold an image and a reputation to the general public to benefit from their support. For instance, The Royal Thai Army led by the former chief, Gen Sonthi Boonyaratglin, staged a coup against the government of Thanksin Chinnawat in 2006. Thenceforward, Thai protests are deeply divided into two extreme political factions known as reds shirt and yellow shirts. Countless protests and riots have erupted throughout the years, and appear to persist. However, the political unrest is not solely rooted from the military coup, but biased news and tv stations are as much the cause. According to Duncan McCargo [7], political scientist of Leeds university, it suggests that each political polar operates their own tv station broadcasting subjective point of view of the opposite party. Additionally, McCargo argues that terrestrial TV stations like channel 3, 5, 7, 9 and 11 are formally and informally controlled by the Thai government. It is impossible for the people to consume objective media or facts while surrounded by blatantly distortion of news from the mainstream channels. Indubitably, media bias is a censorious matter that should be taken into account and properly handled.

1.3 Bias News Detection Mobile Application Mitigates Media Bias

Posterior to the consultation with our supervisor and the research that we have conducted regarding mass media and media bias. We have come across MUIC CDP exhibition, specifically Kaw Na Kai, which proposes the idea of creating Bias News Detection Mobile application in order to attenuate media bias and provide the knowledge for people to understand the drawback and impact of consuming media bias. As computer science students, see the importance of the particular issue, and ought to turn Kaw Na Kai which is an intangible proposal originated by Thai Thanyawong to be a usable and practical mobile application.

1.4 Overview of the application

Our mobile application called Opinion Zero, where the meaning represents or tries to encourage the non-bias opinion toward political news or information that the public consume from a computer program point of view. Opinion Zero focuses on two political biases occurring in Thailand as detection criteria, which are red shirt and yellow shirt. The application would evaluate, detect, and analyze each news by giving the percentage of identifying as red shirt favour or yellow shirt favour and those news will be accessible and displayed on the feed to the users. Our project is mainly focused on two important aspects: Mobile Application and Bias Detection model.

CHAPTER 2

LITERATURE REVIEW

This section of the paper will provide the background, previous work and literature review that will support and explain about an aspect of bias detection with machine learning models. We have come across three different research papers that will help us amplify the knowledge, background and the use of bias detection: Political News Bias Detection using Machine learning, and Exploration of classifying sentence bias in news articles with machine learning models.

2.1 Political News Bias Detection using Machine learning

This research paper is conducted by Minh Vu from the department of Computer Science from Earlham college. The paper is focused on political favoritism's impact on online news and social media networks specifically about the 2016 United States presidential election. The purpose of the paper is to attempt an unconventional approach to detect political bias. According to the paper, the conventional technique on ascertaining the political parity is Recurrent Neural Network. Although Recurrent Neural Network promises with high accuracy results; however, recently, Multilayer Perceptron and Convolutional Neural Networks have been implemented in various deep learning models, and they appear to exhibit outstanding results as well. Therefore, Minh Vu decided to corporate Multilayer Perceptron to build a political bias detection model. The criteria of the bias detection are categorized into three types: conservative, liberal or neutral. In each component of articles, the percentage of one of the three bias types will be determined. [15] For the training data set, the paper utilizes Ideological Book Corpus which is composed of 4,062 sentences annotated for political ideology at sub-sentential level. [13]

It consists of 600 neutral sentences, 1701 conservative sentences, and 2025 liberal sentences. In each sentence, it is represented by a parse tree in which each node is annotated in one of the three criterion (liberal, conservative, neutral. For the word representation matrix of MLP models, the paper uses fast Text word vector representation model, which is an unsupervised learning algorithm that returns a vector representation of words that is developed by Facebook's AI research lab. After the word embedding matrix is initialized, the matrix will be fed to the training session. During the training session, a Multi-layer Perceptron model by Python's machine learning library scikit-learn takes a word representation matrix as an input and outputs the predicted political bias. 75 percent of Ideological Book Corpus datasets will be randomly selected using scikit-learn's train_test_split() function to be a train data set, and the rest will be used as a test data set. To evaluate the training result, the paper uses three following measurements which are F1 score, Precision and Recall. However, the F1 score will be the main metric for evaluating the classifier. The result are recorded three times shown in the figure 2.1

	Experiment 1	Experiment 2	Experiment 3
hidden_layer_size	10,10,10,10	20,20,20,20	500,20,20,20
max_iter	200	1000	1000
batch_size	None	None	32
warm_start	False	False	True
early_stopping	False	False	True
F1 score	**68%**	**72%**	**81%**

Figure 2.1: Experimental Summary

Afterwards, the process will emerge to the Classifier module, where the trained model that is obtained from the training session will be taken in the form of the URL of the news article. In order to extract the word from an article URL, the paper uses Python's goose3

library to transform raw text, metadata and probable image to a vector representation and then feed them to the classifiers. The classifier will then determine the percentage of one of the three political factions. To construct a practical usage of the model in daily life, the paper employs the model as the google chrome extension in order to be used with real news articles. The extension will query the URL of the news article and then send this to the Flask application. The Flask application is being used to facilitate the communication between the google chrome extension and the MLP classifier where the flask application acts as a server to handle HTTP POST from the extension and return back the result from the classifier. There are a total of 20 online news articles being experimented on with the use of Chrome extension as show in 2.2. The result reveals that the majority of the articles are determined to be as close to 100 percent neutral although the article is not truly neutral as the model claims to be. From this, the results of the model are not very much promising.

According to the paper, it reveals several explanations why the outcomes are not favourable. First of all, the news article can neutrally structure all the sentences; however, the overall elements of the article can still be considered opinionated. From this, the classier where it detects the bias sentences by sentences and words by words are undoubtedly not capable. Secondly, negative and metaphor phrases are also missed out from the consideration by the classifier. Finally, there is also the connection between each sentence or phrase when being written. If the classifier is taken into consideration only at sentence/word-level, it is impossible to detect the connection between two or more different sentences. Hence, by all above reasons, the classifier yields undesirable outcomes [15].

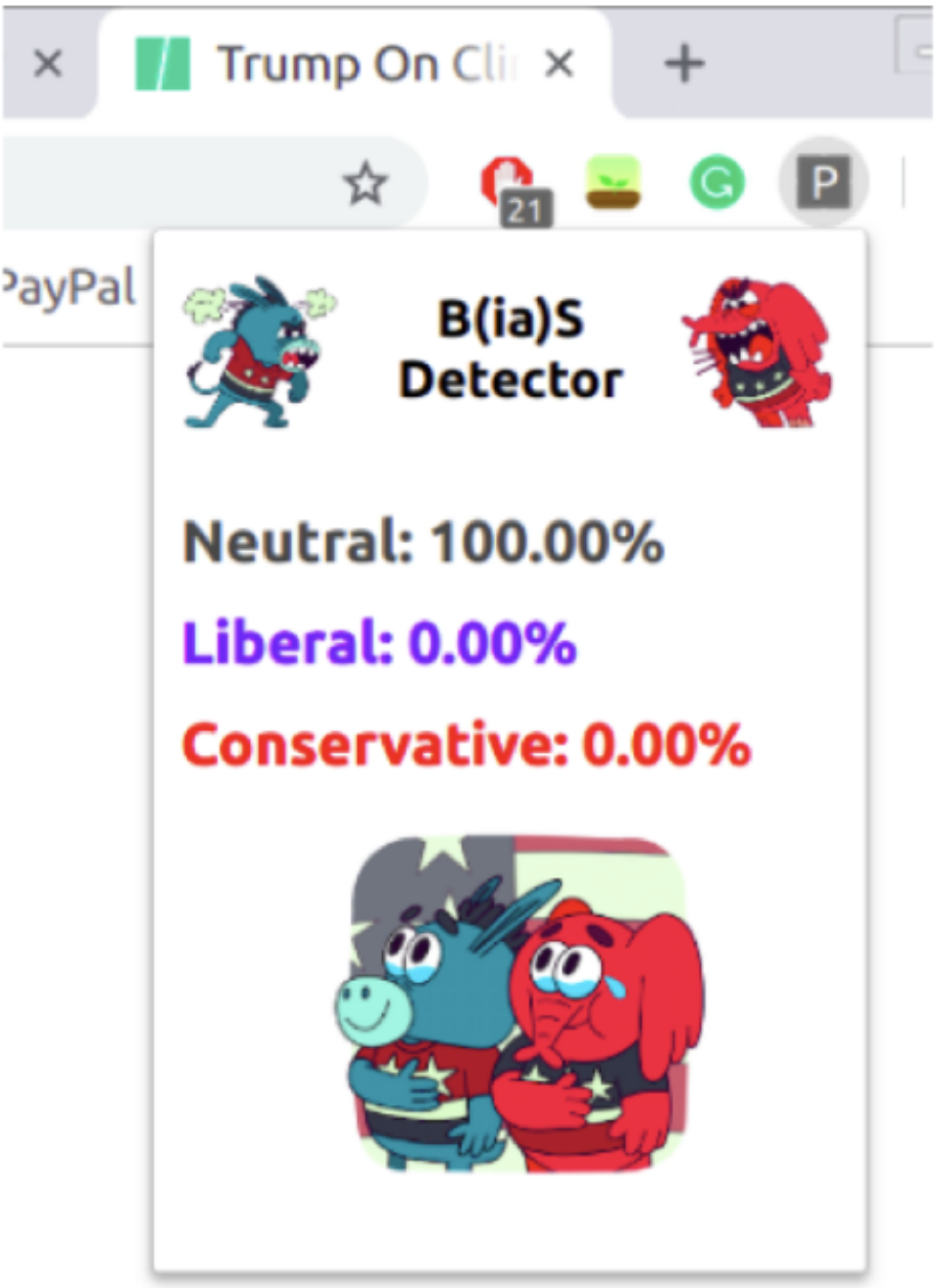

Figure 2.2: User Interface of Chrome extension

2.2 Exploration of classifying sentence bias in news articles with deep learning models

This research paper is conducted by Martha Bellows, University of Rhode Island. The paper presents an overview of what is currently being done in the topic, all in one spot.

Table 6: Online news articles classification

Article	Liberal	Neutral	Conservative
Huffington Post article #1	0%	99.54%	0.46%
Huffington Post article #2	0%	100%	0%
Bloomberg article	0%	100%	0%
CNN article #1	0%	100%	0%
CNN article #2	0%	100%	0%
Fox News article #1	0%	100%	0%
Fox News article #2	0%	100%	0%
Breitbart article #1	5.41%	94.59%	0%
Breitbart article #2	0%	100%	0%
The Economist article	0%	100%	0%
NYTimes article	0.34%	99.66%	0%
Wall Street Journal article	0%	100%	0%
The Blaze article	0%	100%	0%
Slate article #1	0%	100%	0%
Slate article #2	0%	99.91%	0.09%
NPR article #1	0%	100%	0%
NPR article #2	0%	99.69%	0.31%
BBC article #1	0%	100%	0%
BBC article #2	0%	100%	0%
Medium article	0%	100%	0%

Figure 2.3: Online news articles classification

Focused on exploring and investigating various embedding strategies and techniques along with the training of several machine learning models in order to classify or label if the sentences from the news are ”Unbiased” or ”Biased”.

2.2.1 Data

The proportion of annotated data available is a significant challenge towards several Natural Language Processing projects. Categorizing fresh data takes time and money. Since there are few annotated datasets available, therefore, many researchers rely on curating and annotating their own data where diverse methods can be used (Table 1 provides the summary about each researcher's method and their data)

- Using data that has already been labeled is a simple technique to access labeled data. For example, Some researchers use articles from http://www.bitterlemons.org (reflect a combined Palestinian-Israeli effort to encourage a calm discussion of the Israel-Arab issue) where the editors identified each article as Palestinian or Israeli, or categorized as right-ideology or left-ideology. Some researchers that are more into fake news detection employed statements that were human-labeled and truthfulness-tested by politifact.com or some used MPQA dataset which is derived from a range of international news documents. [2]

- Some researchers build up or form the labeled data by generating assumptions about the dataset's source and mapping the resulting emotion or bias to each sentence. For example, personally verifying for a substantial portion that the perspectives stated in papers were as predicted from an apprehend data resulting from the online sites. Moreover, data such as general coverage, debates, and speeches were used and mapped together into ideology, for instance, "conservative or liberal", or "democratic or republican" etc. Another mapping method that was utilized by several researchers is Wikipedia data dumps, where the metadata weather modification (accompanied by a tag indicating the rationale for the update) or not in each page from entire Wikipedia pages were copied. If the modification were made the researcher tends to focus more on the neutral point of view discussion. Afterward, any sentences that had been altered with a "weasel" tag were extracted by them. Those highlighted data then were subsequently utilized as the biased or

non-factual dataset [2] .

- Manually annotating data is one of the most difficult methods to have annotated data. Researchers amplified manual annotations to expand the number of labeled data. According to their research, 100 of the same sentences were annotated by 4 annotators. There were other researchers employed as annotators in person, they believed that in-person annotators do not calibrate well which led to several researchers using crowdsourcing services. Amazon's Mechanical Turk was used by a couple other employees to have their data annotated while additionally others used CrowdFlower [2] .

2.2.2 Machine Learning Models

Machine learning models can be used by fundamentally transforming text or strings into a numerical format by first implanting a document into a vector space. TF-IDF (Term Frequency-Inverse Document Frequency) or Word Frequencies are standard count based vectorizers are relied by researchers to implement various embedding strategies, however an alternative and updates approach is utilizing word2vec or GloVe which are predictive word embeddings that are even capable of creating feature space for data. Researchers would test their theories through variations of models once the data is formatted for the machine to interpret the data. The standard models used by researchers are Bidirectional Long Short-Term Memory (Bi-LSTM), Naive Bayes Recurrent (RNN), Convolutional Neural Networks, and Linear Discriminant Analysis (LDA). Logistic Regression and Support Vector Machines (SVMs) are both commonly used by researchers.

CHAPTER 3

SOLUTION AND DESIGN

3.1 Overview Functionality of Our Opinion Zero Mobile Application

Opinion Zero is a mobile application aiming to tackle two political biases, red shirt and yellow shirt which are arising throughout various media platforms resulting in subjective media consumption. Our application would provide an evaluation, detection, and analysis of the biases within the political news article where those news will be accessible and displayed on the feed to the users as well as the percentage of identifying as red shirt favour or yellow shirt favour.

3.2 Features of Our Opinion Zero Mobile Application

Our project is mainly focused on two important aspects: Mobile Application and Bias Detection model.

3.2.1 Mobile Application:

This section will display the user flow of our mobile application where the user flow is shown in figure 3.1

- Registration Page

  Users register to our application as a member in order to login into our mobile application.

- Login Page

  After login in the application, the page will be redirected to the Profile Page

- Home Page

  Home page consist of article, opinion and profile page, where user can choose to see as they desire.

- Articles

  Purpose of the Articles page is to display the list of news according to their category which are all, red, yellow, and neutral. Where each news will contain their specific information as well as their percentage of political bias position.

- Analysis

  The analysis of political biases of an article will be performed by our bias detection model and display the percentage of each bias type.

- Bias Detection

  Detect political biases toward red shirt and yellow shirt perspectives obtained from our Machine learning and deep learning model.

- Add Opinion Page

  Add Opinion Page provided news articles that have not been voted yet. User can as well read those article and decided their option. Add Opinion Page are also connected to vote page and history page.

- Vote Page

  The user can engaging in rating or voting the news according to their opinion or judgment in order for us to utilize those results and reinforce it into our model training.

- History Page

  The voted articles will be store into voted History page. Where it will display the list of the voted news along with it information.

- Profile Page

  Provide all the information about the user as well as connected with log out button.

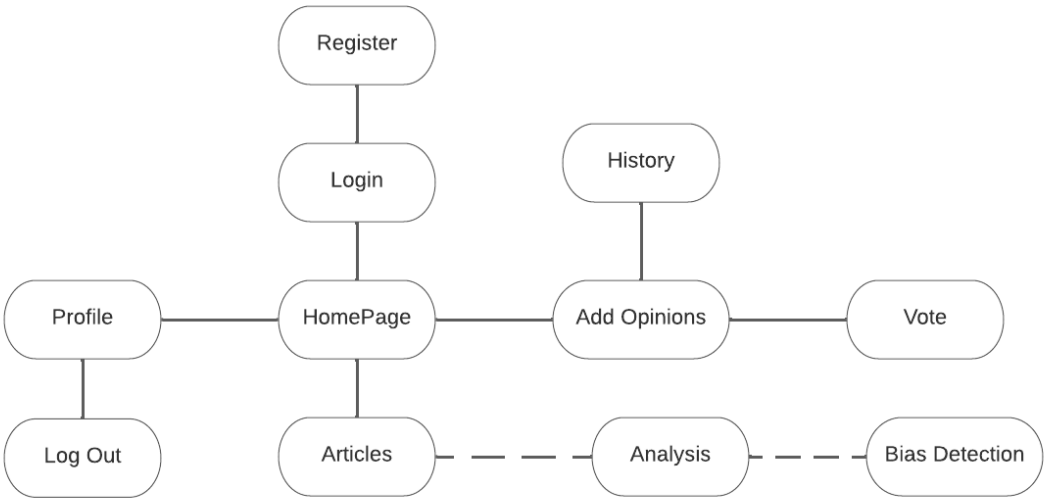

Figure 3.1: User Flow for Mobile Application

3.2.2 Bias Detection model:

For the bias detection model part, we have performed an experiment and investigation following the research paper that was conducted by Minh Vu from the department of Computer Science from Earlham college (Stated in the previous report - background draft) . According to the paper, the conventional technique on ascertaining the political parity is Recurrent Neural Network. Although the paper implemented the classifier by using skit-learn library, in our experiment, we have conducted the classifier by using Torch. Since there was an obstacle toward gathering the dataset from Ideological Book Corpus, we managed to find another dataset from Jerry Wei, where the dataset contains 200,000+ sentences about Donald Trump labeled by news source and political bias. We have tried experiments for a total of 4 times each time varying parameters such as numbers of epoch, learning rate and adding more layers. We establish an experiment by reading and cleaning the test data and adding them into the new data frame where there's a label of classes and the sentence data. Since 10 classes were labeling based on the news sources where they have gathered the data such as NewsdaysLiberal, New York PostConservative, etc, we then modify the data frame by adding the new label with 3

classes: conservative 1, Liberal 2, Neutral 0. We then continue the process by using FastText, to convert the sentences into a vector representation output. Afterwards, we retrieve each dataset in order to feed the classifier by using Dataloader. Once the dataset is ready to be trained, we feed to the classifier. Finally, we repeat the same process with the test dataset to test accuracy of our model.

- The first model was constructing based on the research paper which resulting :

  Hidden_layer_sizes=(500, 20, 20, 20) with learning rate 0.001

  Precision 0.6037849155504788

  Recall 0.5964545454545455

- We attempt to improve the second model by using

  Hidden_layer_sizes=(1000,800,450,800) with learning rate 0.001, randomly increasing the number of neurons because our dataset contains a greater amount of sentences compared to the paper that bases the neuron model on 4096 sentences.

  Precision 0.6097126056827492

  Recall 0.6061818181818182

- Third Model with Hidden_layer_sizes=(1200,400,400,400) with learning rate 0.001

  Precision 0.6046258362357615

  Recall 0.6011818181818182

As you can see, in all three trials of the experimentation, the model's performance is not robust. Based on our observations, the issues might be coming from the fact that the data does not contain much predictive value or the loss isn't plateau yet. What we can do to improve the precision and recall is perhap trying to use the different learning rate and add the complication of the model. However, we have faced another obstacle in training the data as we are limited to using the GPU on google colab, we are not

able to experiment on different models in order to achieve efficient performance classifiers.

3.3 Prioritize features with justifications

The prioritize features are undoubtedly the Articles, Analysis and Bias detection section of the application since they are the core of the application. In short of the mentioned sections, the user would not be able to examine the new articles as well as acknowledge the subjectivity regarding political prejudice occurring in each new article. With the bias detection section, we need to gather the dataset in order to acquire an efficient classifier. The possible challenge that we might encounter is gathering a sufficient as well as high quality dataset. This is because we are primarily focusing on major political polarities in Thailand where there is an inadequate amount of datasets to reach for. Furthermore, although collecting biased news regarding Thai political bias might be possible, so far from our observation and acknowledgement, there are still no datasets that are labelled in those two parties yet. The analysis section of our application, it will be based on bias detection as resulting in the percentage of each bias type.

CHAPTER 4

IMPLEMENTATION

4.1 Data Collection

Data was collected for 473 news articles, of which 101 were classified as red shirt, 105 as yellow shirt, and 267 as neutral. Beautifulsoup and Scrappy, which are libraries particularly designed for scraping the webpage, are the tools we use to scrape the news from the website (the process is shown in figure 4.1) . We have gathered information from a variety of Thai news outlets, including Voice Tv, NationTv, mgr-online, Thiarath, and others. However, there is no data available online that categorizes Thai news as red, yellow, or neutral. As a result, we must assess the sort of prejudice ourselves. Nevertheless, we have specific criteria or standards that we use to categorize each news item.

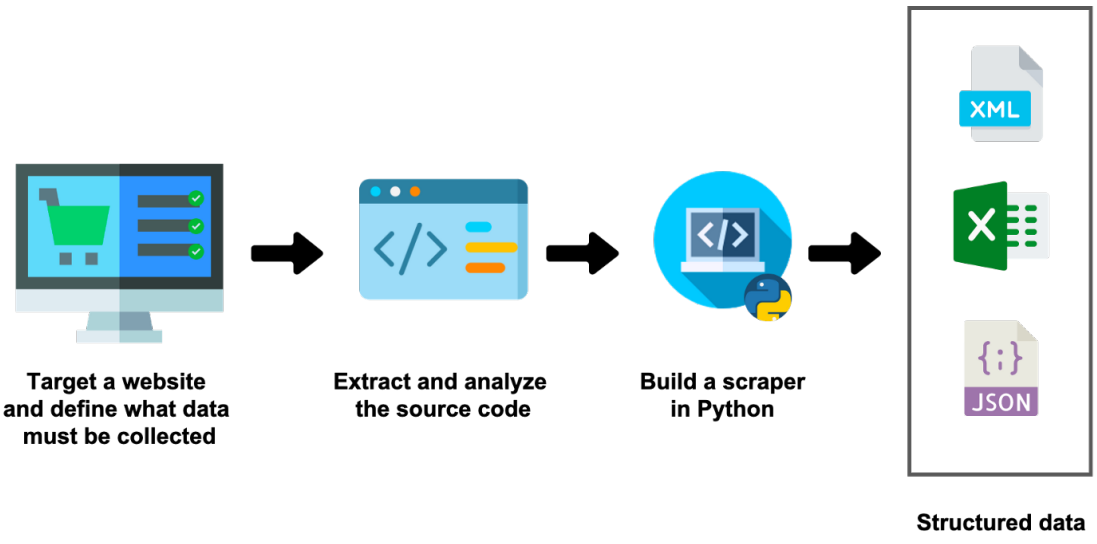

Figure 4.1: Process of web scraper

How to evaluate who we mean by Yellow shirt, Neutral shirt, and Red shirt. For yellow shirts, we consider the current government, Pracharat Party, Prayut chan-o-cha,

Prawit Wongsuwan and Anutin Charnvirakul. For neutral, the emphasis is on explaining what happened in the event, with no subjective comments included. The red shirts are represented by Pheu Thai party and its alliances, the Future Forward Party, and Thaksin Shinawatra.

4.2 Criteria of Bias Classification

We divided it into two categories: praise and criticism. We define an activity as Praise when there is unwarranted adulation for a certain political organization, and Attack when an article targets the opposing political group. For example,

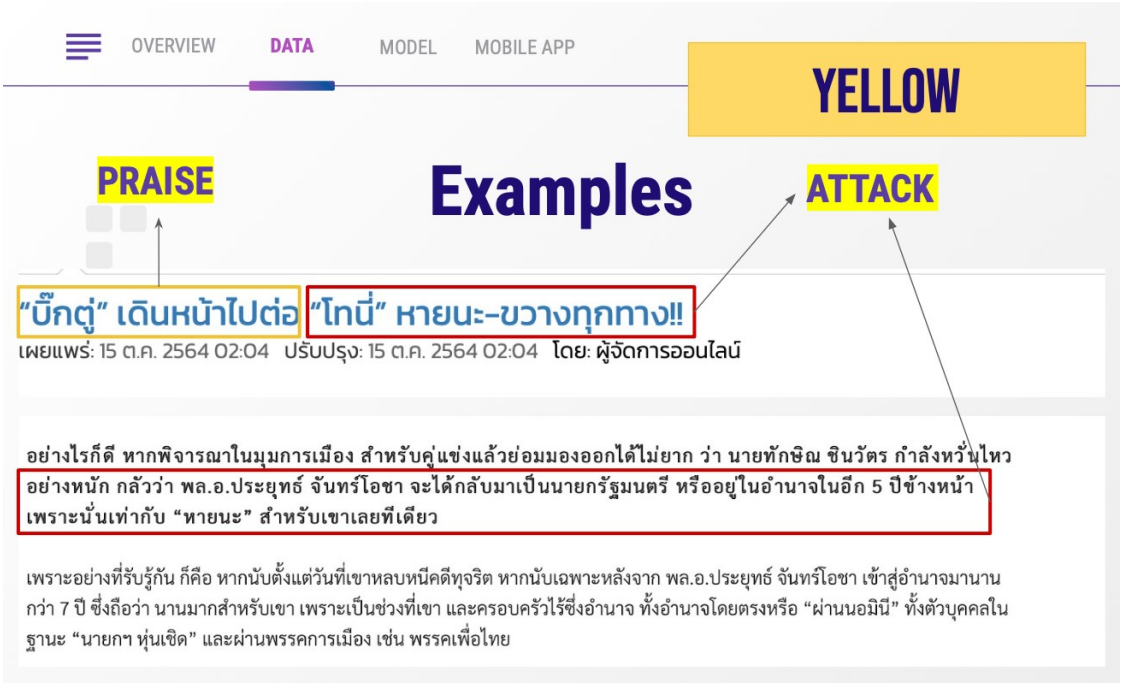

Figure 4.2: An Example of evaluating a news

The figure 4.2 is the example of news we classified as Yellow. Firstly, the heading purposely praises BigTu which refers to Prayut chan-o-cha, the current prime minister of Thailand. Secondly, the second part of the headline and the description of the news attack the opposition party, which is Thaksin Shinawatra. The news attack him by saying

he is the disaster blocking the way for Prayut chan-o-cha. Hence, the news is labeled as yellow shirt.

4.3 Text Classification with NLP

There are 3 text classification approaches were experimented to select the best performance model. Those methods are TF-IDF, FastText, and BERT. The workflows are demonstrated in figure 4.3 which include input sentences, embedding and model.

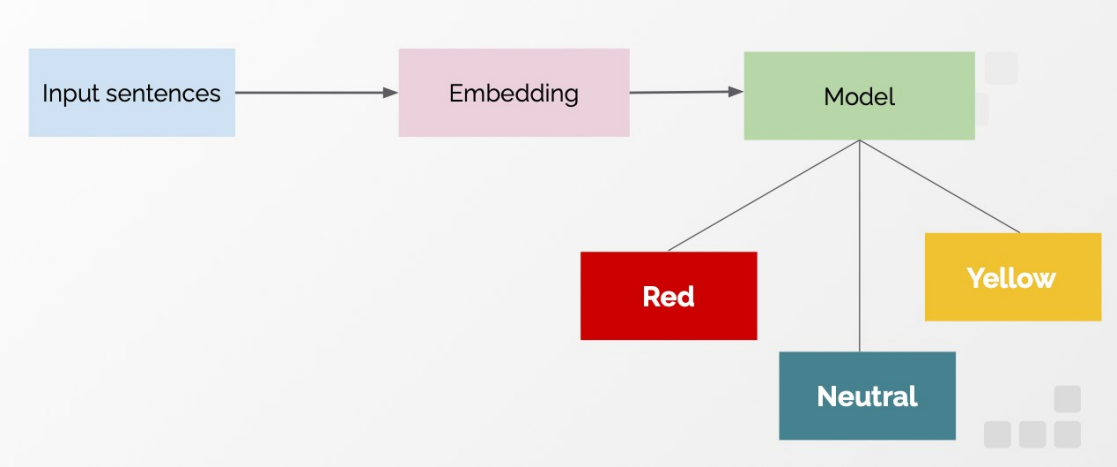

Figure 4.3: Overall Workflow of Text Classifications

4.3.1 TF-IDF

TF-IDF stands for Term Frequency and Inverse Frequency. It is the technique that quantifies words in the dataset in order to select the relevant words in the document by considering at Term frequency and Inverse Document frequency. Term Frequency is the frequency of a particular term in the dataset. In term of mathetical expression, tf is a count of a particular word in a document is divided by total number of words in that document as shown in figure 4.4. In the language, there are unmeaningful words; however, are abundant. To solve the issue, IDF palys a part. IDF basically defines whether the words are meaningful or not. In mathematical terms, it is a log of total

number documents divided by number of document containing the word as shown in figure 4.5.

$$TF(W,D) = \frac{\text{Count of W(word) on D (Document)}}{\text{Total number of words in D (Document)}}$$

Figure 4.4: Mathetical Expression of TF

$$IDF(W,D) = \log\left(\frac{\text{Total number of D (Documents)}}{\text{Number of D(Document) contain W (Word)}}\right)$$

Figure 4.5: Mathetical Expression of IDF

4.3.1.1 Model with TF-IDF

```
articles = []
for i,row in tqdm(df.iterrows()):
  new_token_sent = []
  article = row['article']
  split_chunk = article.split("<_>")
  for chunk in split_chunk:
      token = deepcut.tokenize(chunk)
      for word in token:
        new_token_sent.append(word)
  articles.append(" ".join(new_token_sent))
```

```
363/? [32:04<00:00, 3.93s/it]
```

```
df["tokenized"]

0      15 มีค 2565 นายนริศโรจน์ เฟื่องระบิล อดีต เอกอ...
1      การ สลาย การ ชุมนุม ของ กลุ่ม " แนวร่วมประชาธิ...
2      ข่าว การ เมือง sunday september 19 2010 1703 –...
3      " แรมโบ้ " สมเพช " ทักษิณ " เพื่อเจ้อ การ เมือง...
4      " แรมโบ้ " ชัด " โหนี่ " ก่อน กล่าวหา นายก ฯ ใ...
                             ...
468    ' ฝ่าย ค้าน ' พร้อม ยื่น ชัก ฟอก ทันที ก่อน ' ...
469    " เทพไท " หนุน แนว คิด " อานันท์ " รัฐประหาร ไ...
470    ' ยุทธพงศ์ ' แจง ปม เรือ ดำ น้ำ ไม่ มี เครื่อง...
471    ' เพื่อ ไทย ' จี้ หยุด ใช้ พรกฉุกเฉิน หลัง ต่า...
472    " อุบลศักดิ์ " แนะ " บิ๊ก ตู่ " ลา ออก ก่อน 22...
```

Figure 4.6: Process of selected 20 words with the highest TF-IDF score

The dataset was tokenized by the library called Deepcut. Next, TF-IDF score of each words was calculated, and only 20 words acquiring the highest TF-IDF score were selected to be trained next. In the training, word2vec was not implemented instead there is an addition layer called Embedding layer generates the matrix representation of each words [11]. As shown in figure 4.6.

4.3.2 Fasttext

Fasttext generates word embedding matrices. Fasttext is similar to Word2Vec, in fact, it's an extension of Word2Vec but Fasttext represents each word as a sum of n-gram of characters [4].The distance of the vector can be determined by the meaning of each words. Words with a similar meaning will be generate closely to each other while the vector of the distinct words will be generate further away [6]. For instance, in the figure 4.7, the word "sabai" and "saduak", they both mean convenient. The distance between these two vectors representations should relatively be closer than the vector representation of "toramarn" which means torturing.

Figure 4.7: Example of how the distance of each words determine the meaning.

4.3.2.1 Model with FastText

The same dataset and tokenizer were utilized to be trained. The model has the same structure as the previous model but there is an absence of Embedding layer since it was substituted with Fasttext.

4.3.3 BERT

BERT is an NLP framework meant to assist computers in understanding language by establishing context via the use of surrounding text [10]. BERT was trained using Wikipedia text and may be fine-tuned using question and answer data sets. Furthermore, Word embedding is calculated using attention techniques.

```
model.get_word_vector("สบาย")

array([-0.18490309,  0.01415917,  0.19842336,  0.19839667,  0.06000606,
        0.09737544,  0.0091152 , -0.10471837, -0.07241923,  0.09690195,
       -0.13328528,  0.12690865,  0.05586451,  0.04766925,  0.14185551,
       -0.13236272,  0.0201336 ,  0.05975136, -0.03869424,  0.04371473,
        0.07019685, -0.07494999, -0.00674674,  0.15173095, -0.04820947,
       -0.05320934, -0.00927462,  0.07044205,  0.09223807, -0.07702143,
        0.0110035 , -0.05349493,  0.1234849 , -0.07352224, -0.05518612,
        0.04050654,  0.11372646, -0.06044541,  0.05456267,  0.02107452,
        0.01718373,  0.01866977,  0.04123905, -0.04798517, -0.0052929 ,
        0.01274821, -0.1713537 ,  0.00566839, -0.05491246,  0.01721531,
       -0.00716366,  0.04047325, -0.00138203, -0.11270501, -0.11415495,
        0.04446452,  0.03905062, -0.06689805,  0.00119306,  0.07470857,
       -0.00916243, -0.09009416, -0.04916331,  0.08073813,  0.04559639,
        0.001015  , -0.00567308, -0.01451922, -0.06979126,  0.04496589,
       -0.0497834 , -0.17941013, -0.18689436, -0.05730148, -0.08283162,
       -0.14768784,  0.00119025, -0.05276813,  0.13011178, -0.0252868 ,
       -0.04874878, -0.00342584, -0.04593055,  0.01815317,  0.01940273,
       -0.01352687,  0.10130398, -0.07213007,  0.02877728,  0.05478133,
       -0.03210478, -0.08129535,  0.01807626, -0.05231233, -0.00717102,
        0.04961552,  0.0987471 ,  0.0340124 , -0.05660535, -0.08371805],
      dtype=float32)
```

Figure 4.8: Vector representation of "" from FastText

4.3.3.1 Problems of Previous Models

Before be looking deeper in BERT, let us first understand problems of previous models. Each word is analyzed independently, and input words are transmitted one after the other. One time step at a time, the word embedding is created (shown in figure 4.9). Furthermore, context information or the arrangement of words in phrases were not taken into account.

4.3.3.2 How Bert Works?

There is no idea of time step for the input in Transformer. At the same time, we pass in all of the sentence's words and identify the word embedding [14].

Components of a transformer, composted with an encoder and a decoder. The BERT transformer, on the other hand, just has an Encoder part (shown in figure 4.10).

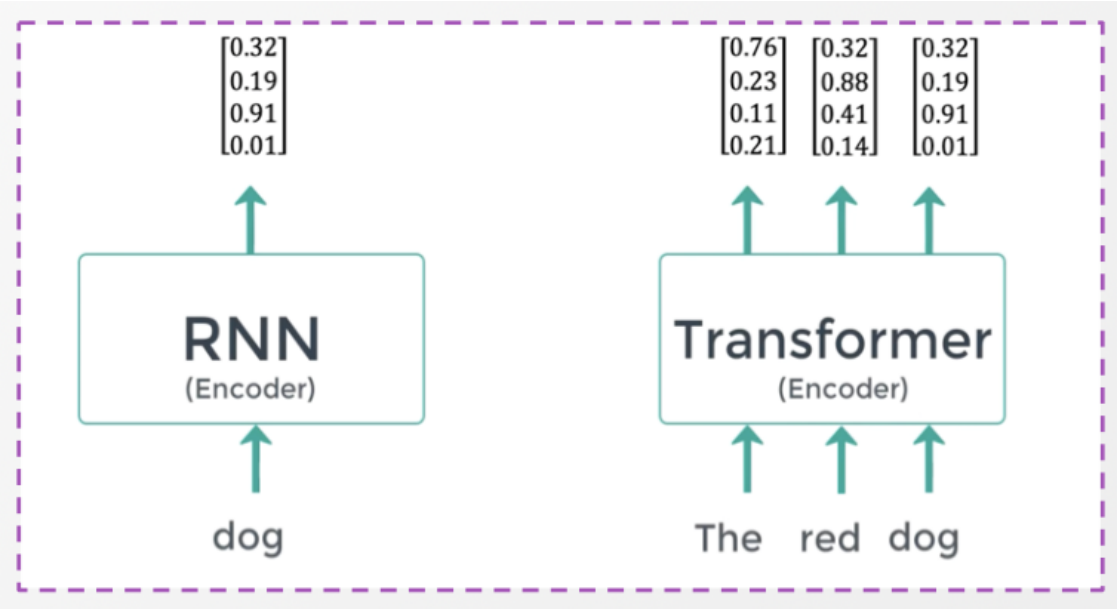

Figure 4.9: Comparison between RNN and Transformer

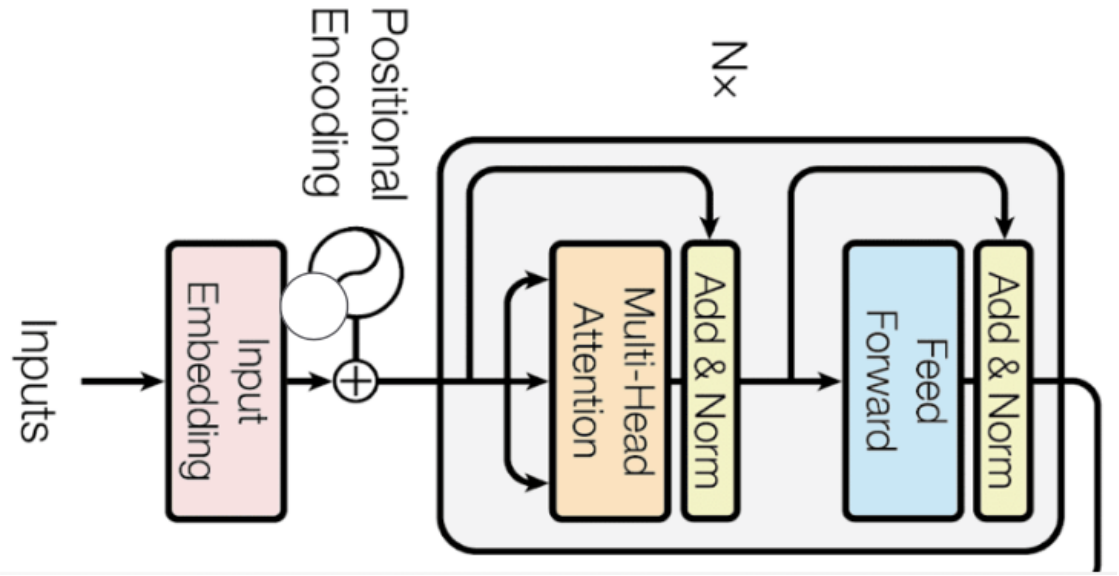

Figure 4.10: Bert Encoding

According to what we know, earlier models did not take into consideration context information, but BERT did. We understand that embedding space transfers a word to a vector, but the same word may have multiple meanings in different phrases (shown in figure 4.11). For example, AJ's dog is adorable, and AJ resembles a dog. This is when

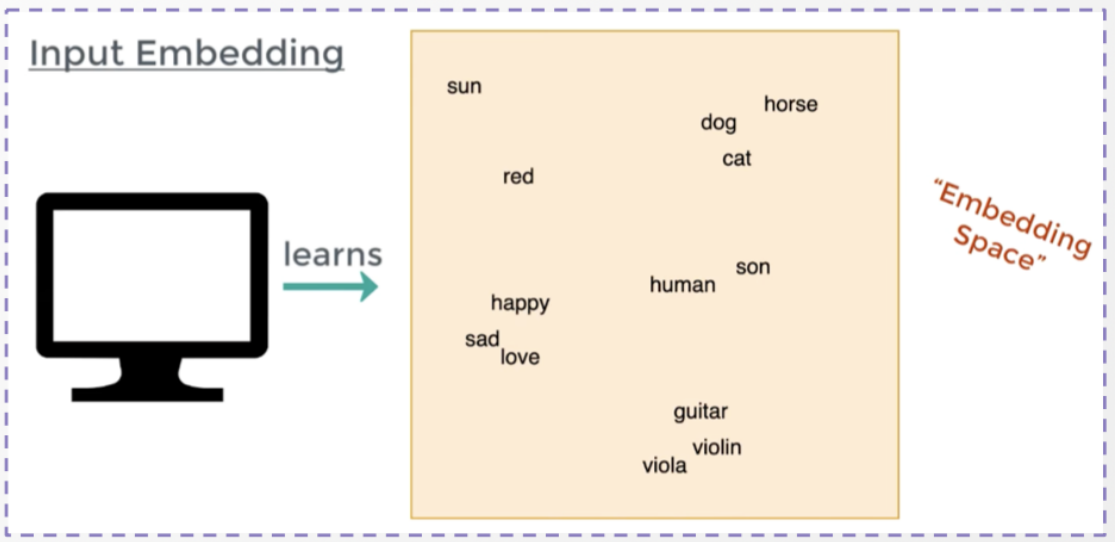

Figure 4.11: Bert Input Embedding

the Positional Encoder comes into play.

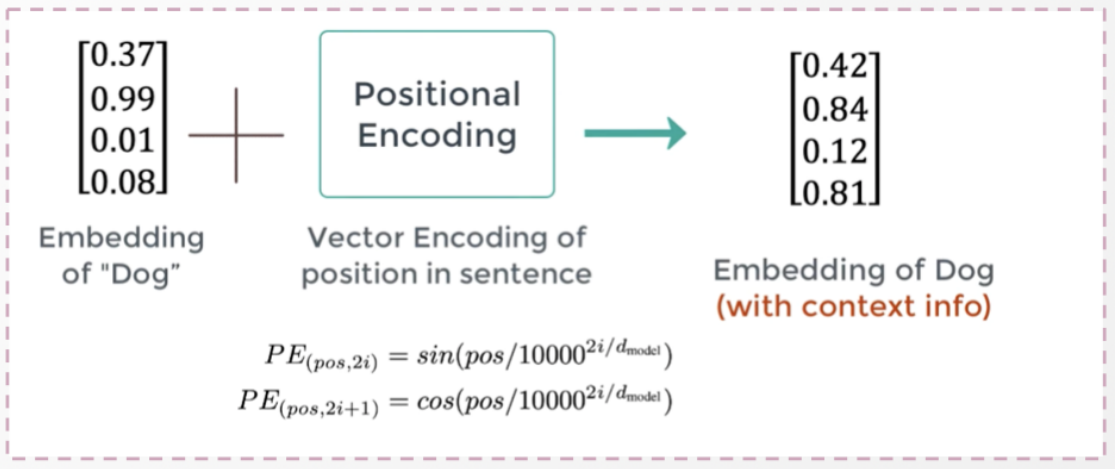

Figure 4.12: Positional Encoder

The Positional Encoder is a vector that provides context based on the location of a word in a sentence, as well as information on the distances between words and the phrase. To produce the vector, the original study employs a sine and cosine function. After running the English phrase through the input embedding and applying positional encoding, we acquire word vectors with context-specific positional information (shown

Figure 4.13: Example after Apply Positional Encoder

in figure 4.12 and figure 4.13) [5].

4.3.3.3 Model with BERT

We utilize Bert's pre-trained model, hence we also use bert's pre-train tokenizer named Monsoon-nlp/bert-base-thai. Following that, we divided the train test as normal (70 percent). Then we used the tokenizer that we had previously loaded to tokenize our train and test data. In this situation, we loaded the pretrain bert and assigned the number of labels, which is three. We input the pre-train bert our tokenized train dataset.

4.3.4 BERT with Siamese Model

Since Bert is the pre-trained model opens for many techniques to fine tune. One of the most famous techniques of fine tuning and being implemented in this case is called Freeze the entire architecture. This particular technique is when all of the BERT layers are frozen, then the retrain process occurs by attaching one or more network layers after the pre-trained BERT. This causes the weight of the model to only be modified in the attached layers. In this case, this particular technique are used. However, only one layer is used to be an attached layer.

4.3.4.1 Attached Layer

The structure of the attached layer adopts a new neural network architecture called Siamese Networks. Siamese network originally is solve the problem like facial recognition or signature verification. However, problem arises when numbers of data is not enough

to be trained. Even the simplest solution is to finding more data. However, it is not sustainable and efficient to depend on number of data. This is where Siamese model comes to place.

4.3.4.2 What is Siamese Network?

Siamese Network is a neural network that take a pair of Input and inside the network it consists of one or more identical networks. Identical in this case implies that each model are required to have the same parameters as well as weight. In each network, they are also responsible for computing feature of their input. Subsequently, Euclidean distance of two features is calculated to identify the similarity of both features. In other words, if the two features are close in terms of distance, then they are likely to be in the same thing, otherwise it is different (shown in figure 4.14).

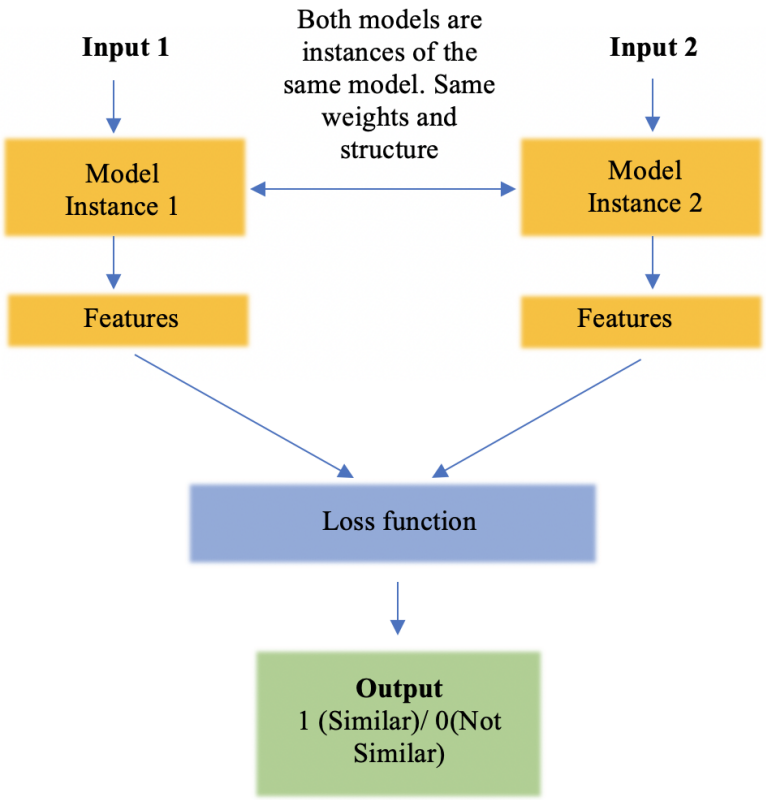

Figure 4.14: Siamese's Structure

4.3.4.3 Structure of Attached Layer

Although the last layer of the model follow the same structure as the Siamese, the purpose of the Siamese model and our attached layer are different. Our purpose of is not to find the similarity or difference between two inputs like Siamese model. Moreover, the two input are not the same kind. However, the fundamental reason why the attached layer are two-headed alike pattern and receive title as another parameter is because when the tokenizer Bert that being used, airesearch/wangchanberta-base-att-spm-uncased demand max-length parameter[8] . This parameter limits the length of each sequence up to only 416 tokens. The problem is 416 tokens is not sufficient to represent all the content in the article, which clearly can be tokenized more than 416 tokens. By limiting the max length only 416 tokens, some significant part of the news may be left out. The title can fill in the missing part. This is because title of the news usually sum up the main idea of the article as well as the title tends to biased if the article is biased. Inputting the title might lead to more accurate result.

4.3.5 BERT with Softmax activation function

In this subsection, we will discuss about BERT with Softmax activation function.

4.3.5.1 What is Softmax activation function?

Softmax Function or known as SoftArgMax Function is a function that accepts the input value as a vector of k real values and converts it into a vector of k real values that add up to 1. Those output real number vector values can be elucidated into probabilities since Softmax Function alters it into values between 0 and 1. Softmax will convert the value into small probabilities, if it happens to accept the negative or small input value. Likewise, large probability if the input value is large. However, it still always falls between 0 and 1. Softmax formula is $\frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$ [12] .

4.3.5.2 Our model with Softmax?

Softmax is one of the activation functions that we decided to experiment and explore, since it is one of the most common activation functions. We decided to apply softmax function as the last layer of activation function and let it split out the probabilities of each possible class which are red, yellow, and neutral.

4.3.6 BERT with Tanh activation function

In this subsection, we will discuss about BERT with Tanh activation function.

4.3.6.1 What is Tanh activation function?

Another method that we have experimented on is to apply Tanh hidden layer activation function or as well referred as hyperbolic tangent activation function with our BERT classifier. Tanh activation function is considered quite close to the sigmoid activation function as it has a similar S-shape, this activation accepts any real value as input and returns values ranging from -1 to 1. As the input value is larger it is likely that the output or the returning value will approach 1. Likewise, if the input value is smaller it is likely that the output or the returning value will approach -1. $\frac{(e^x - e^{-x})}{(e^x + e^{-x})}$ formula is used to calculate the activation function.[12]

4.3.6.2 Why does the Tanh activation function not work?

In our trial and run, we decided to alter our last layer's activation function from softmax to tanh. Since we want to make sure that we have experimented and explored on all of the possible options and as well get to see the varies out come, we have come to a conclusion that Tanh hidden layer activation function is fail to serve our propose. As mentioned above, Tanh activation is actually split out the returns values ranging from -1 to 1 and as we know that our prediction can only be three classes which are red, yellow, and neutral. Thus, we have to observe the overall returning data points (which is the number output, but as we plot it for clearer vision we called it as data point) from tanh and try to find the possible clear cut for each cluster (our classes) that was form by each

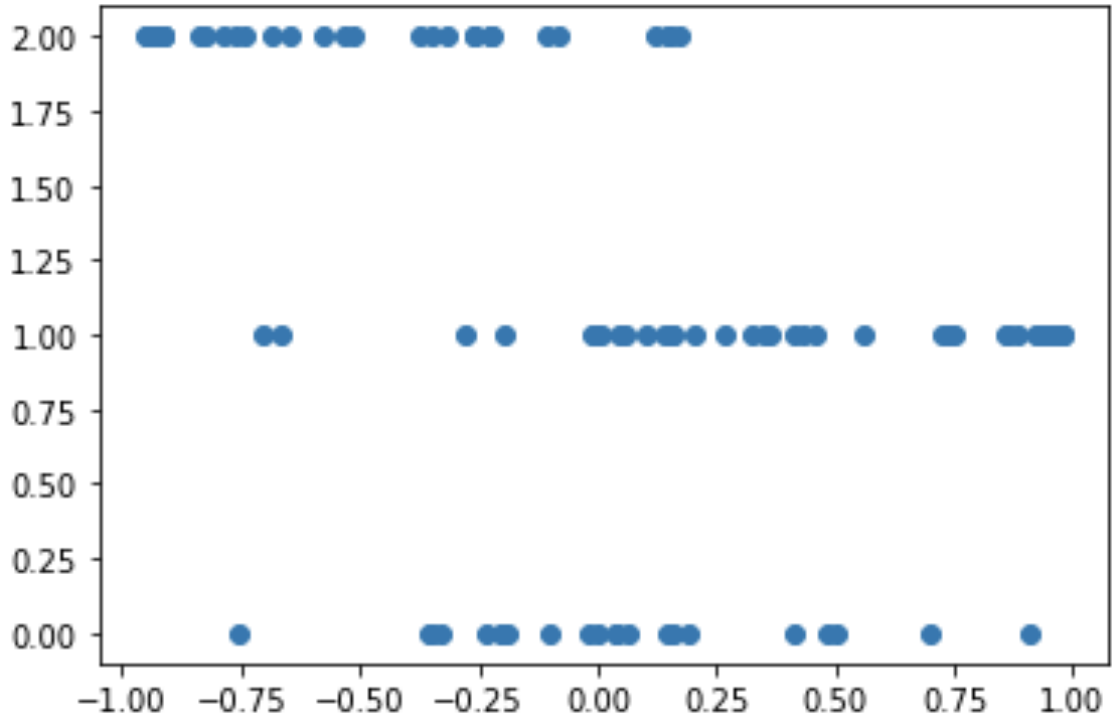

Figure 4.15: Plotted Result of BERT with Tanh activation function

data point value returning from tanh ranging from -1 to 1, so we know exactly until what range we have to classify our news as class red, yellow, or neutral .

From the result shown in figure 4.15, the data points turn out to be overlapping each other as we plot it out. Hence, it is impossible for us to find a clear range of numbers as a cut or the clear cut for each clutter as an accurate classifying class for our news.

4.3.7 BERT with Triplet Network

We have experimented with the structure of the attached layer by following one of the most popular architectures which is Siamese Network. Nowadays, there are newer methods called the Triplet network proposed by the Department of Electrical Engineering Technician Israel Institute of Technology.Triplet network is an improvement of the Siamese Network.

What is Triplet Network?

Triplet Network aims to learn useful representations by distance comparisons to use in image classification tasks. The structure of the Triplet Network consists of 3 parameters input feeding to the network.The three input will denoted as x(reference), x^+(positive), x^-(negative). The reference is one in interest. The positive is the one that belongs to the same class as the reference. The negative is the one belonging to a different class as the reference. We feed these three inputs into the process called NET. This NET process will transform these inputs into some vector representation. How does the NET determine what is the best output vector representation? There will be a cost function to determine if the vector from NET accurately represents the inputs or not. The lost function is depicted as follows. [3]

$$Loss(d_+, d_-) = \|(d_+, d_- - 1)\|_2^2 = const \cdot d_+^2$$

where

$$d_+ = \frac{e^{\|Net(x) - Net(x^+)\|_2}}{e^{\|Net(x) - Net(x^+)\|_2} + e^{\|Net(x) - Net(x^-)\|_2}}$$

and

$$d_- = \frac{e^{\|Net(x) - Net(x^-)\|_2}}{e^{\|Net(x) - Net(x^+)\|_2} + e^{\|Net(x) - Net(x^-)\|_2}} .$$

The idea of loss function aims to maximize the difference of the distance between the trained embedding vector of of the reference x and the positive x^+ and the trained embedding vector of of the reference x and the negative x^- in order to retain the distance between the reference and positive lesser than the distance between the reference and negative.

Structure of Attached layer

The input parameter required six inputs. The first three input parameters are the same as what the triplet network requires, which are x(reference), x^+(positive), x^-(negative). For example, the reference belongs to red a shirt political group. Hence, the positive must be the different article, but belongs to the same political group as the reference,

which is red shirt. For the negative, it must be the different article does not belong to red shirt. The remaining parameters are the title of reference, positive and negative article. The intention behind the inputting title is already mentioned in BERT with the Siamese model section. Afterwards, those six inputs are fed to tokenizer, which is airesearch/wangchanberta-base-att-spm-uncased. Then, Both title tokens and article tokens are processed by the pre-train BERT. The vector representation from pre-train BERT will be feed in the attached layer where the layer return the vector representation of reference, positive and negative. Ultimately, those vector will be input to calculate triplet loss, and repeats.

4.4 Result

The following are the outcome of each models from three approaches.

4.4.0.1 Result of Model with TF-IDF

The accuracy, recall and precision are estimately 0.577. When it predicts yellow, it scores only 6 c orrect and predicts 23 wrong. For Red, the model correct prediction is as much as the wrong prediction. All in all, the performance is not promising.

	Neutral	Yellow	Red
Neutral	61	5	19
Yellow	15	6	8
Red	9	4	15

Figure 4.16: Result of Model with TF-IDF

4.4.0.2 Result of Model with Fasttext

The performance of the two is also undesirable. The model prediction lean toward neutral more than red or yellow. The model predicts neutral for 28 out of 30 and 0 yellow where it is meant to predict yellow. This also applies to the red data as well.

	Neutral	Yellow	Red
Neutral	83	0	2
Yellow	15	14	0
Red	17	0	11

Figure 4.17: Result of Model with Fasttext

4.4.0.3 Result of Model with BERT

BERT model performs the best out of all. It performs great on predicting neutral class; however, perform poorly on the rest, especially predicting red.

	Neutral	Yellow	Red
Neutral	83	0	2
Yellow	28	0	1
Red	24	0	4

Figure 4.18: Result of Model with BERT

4.4.0.4 Result of BERT with Siamese Model

The performance from BERT with the Siamese Model is highly acceptable for us. Since the model predicted correctly and less bias toward one side. As you can see from the image below: The model predicts neutral 41 correctly out of 54, predicts yellow 52 correctly out of 65, and a very good performance on predicting red which is 54 out of 59.

4.4.0.5 Result of BERT with Tanh

As mentioned in the section of BERT with Tanh, it is impossible to find the exact threshold to determine what value belongs to each of the classes. Hence, the result interpretation is executable.

	Neutral	Yellow	Red
Neutral	41	6	7
Yellow	5	52	8
Red	1	4	54

Figure 4.19: Result of Model BERT with Siamese Model

4.4.0.6 Result of BERT with Softmax

The performance of BERT with Softmax is moderately good. It performs best on speculating neutral class where it predicts 10 wrong in the total of 53. However, the model accomplishes the same outcome when predicting yellow and red classes, where the number of incorrect guesses are 15 on both classes.

	Neutral	Yellow	Red
Neutral	43	3	7
Yellow	5	47	10
Red	5	10	50

Figure 4.20: Result of Model BERT with Softmax

4.4.0.7 Result of BERT with Triplet Network

In this subsection, we will discuss about the Result of BERT with Triplet Network with and without normalization.

4.4.0.8 Normalize

The result of BERT model with Triplet Network and Normalization yields an appealing outcome. It is exceptionally predicted well on the neutral class where it predicts 48 correct out of 54. Nevertheless, when it comes to predicting yellow class, it does not

perform as well as predicting neutral class. It speculates 43 correct and 22 wrongs. For predicting a red class, the model's execution is preferable, where there are a total of 6 incorrect speculations out of 59.

	Neutral	Yellow	Red
Neutral	48	2	4
Yellow	12	43	10
Red	3	3	53

Figure 4.21: Result of Model BERT with Normalize Triplet Network

4.4.0.9 Non-Normalization

The result of BERT model with Triplet Network and Non-Normalization is acceptable. The number of wrong executions are the same for all classes, which is 13. It predicts a total of 41 correct out of 54 for neutral class. On the other hand, the model guesses the right answer for 52 out of 65 for yellow class, and it predicts 47 correctly in the total of 59 when it comes to red class.

	Neutral	Yellow	Red
Neutral	41	11	2
Yellow	3	52	10
Red	1	11	47

Figure 4.22: Result of Model with BERT Non-Normalization Triplet Network

4.5 Result Comparison

In this section, we will compare results from different methods

4.5.1 Performance of TF-IDF Model

The model predicting red and yellow correct is as much as when the model predicts incorrect.

Figure 4.23: Performance of TF-IDF Model

4.5.2 Performance of FastText Model

The model prediction lead toward neutral more than red or yellow. Not as accurate.

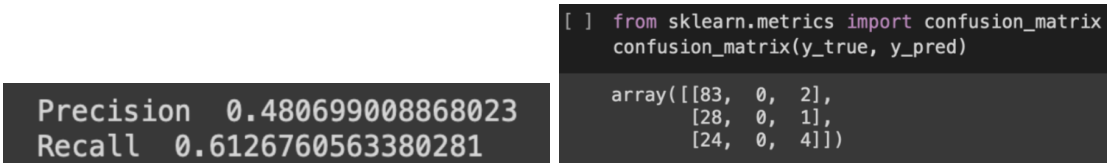

Figure 4.24: Performance of FastText Model

4.5.3 Performance of BERT Model

The model performed great on predicting neutral class. However, performed poorly on the rest especially predicting red.

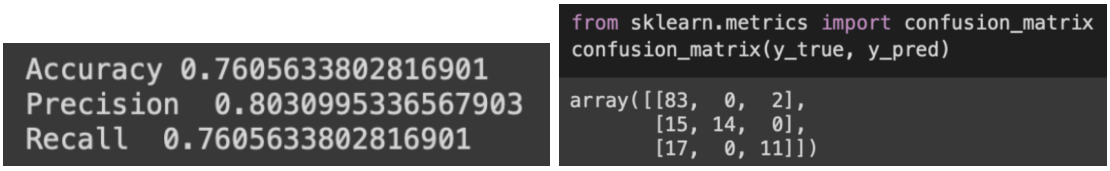

Figure 4.25: Performance of BERT Model

4.5.4 Performance of BERT with Siamese Model

The Model results out the desirable outcome where precision is 0.83 or 83 percent. It is considered the highest precision percentage out of all models that we have performed.

whereas the recall and accuracy are 0.82 or 82 percent. BERT with Siamese Model performed much better and resulted in greater accuracy than the model that implements only BERT, TF-IDF, fasttex, softmax and as well as triplet network.

```
[31] print_test_scores(answer, x_class)

     Precision  0.8303169946879762
     Recall  0.8258426966292135
     Accuracy 0.8258426966292135
     Confusion Matrix [[41  6  7]
      [ 5 52  8]
      [ 1  4 54]]
```

Figure 4.26: Performance of BERT with Siamese Model

4.5.5 Performance of BERT with Softmax

The accuracy of the model prediction is approximately 77.81, whereas the Precision and recall are equal with the value of 77.77. The model performance is better than Model with only BERT, TF-IDF and fasttex. Nonetheless, the model outcome is beaten by Siamese Model as well as Triplet Network. Since there are evidently better performed models, thus, this model is not the final model.

```
Precision  0.7781896075179658
Recall  0.7777777777777778
Accuracy 0.7777777777777778
Confusion Matrix [[43  3  7]
 [ 5 47 10]
 [ 5 10 50]]
```

Figure 4.27: Performance of BERT with Softmax

4.5.6 Performance of BERT with Triplet Network

In this subsection, we will discuss about the performance of BERT with Triplet Network with and without normalization.

4.5.6.1 Normalization

The outcome of Normalized BERT with Triplet Network is considered desirable since it achieves 80.08 accuracy and recall. For precision, it receives 82.04. Although the overall

```
Precision  0.8204700314638684
Recall  0.8089887640449438
Accuracy 0.8089887640449438
Confusion Matrix [[48  2  4]
 [12 43 10]
 [ 3  3 53]]
```

Figure 4.28: Performance of BERT with Normalization Triplet Network

metrics score of this classifier does not yield the best result, all in all, it is still preferable. Additionally, this classifier is not selected as the final model since the prediction result is calculated by looking at the difference of distances, it is impractical to return the percentage of what is the likely chance this news article belongs in each class. Therefore, it is not suitable to be the final model.

4.5.6.2 Non-Normalization

```
Precision  0.7970543577285l5
Recall  0.7865168539325843
Accuracy 0.7865168539325843
Confusion Matrix [[41 11  2]
 [ 3 52 10]
 [ 1 11 47]]
```

Figure 4.29: Performance of BERT with Non-Normalization Triplet Network

The outcome of the Non-Normalized BERT model was considered acceptable as it resulted in 79 percent of precision and 78 percent of accuracy and recall. The overall metrics score is acceptable but doesn't provide the best result and as well get beaten by Normalized BERT and BERT with Siamese Model.

4.5.7 Potential issues

There is not enough datasets. Deep learning model heavily rely on the numbers of dataset in order to achieve favorable outcome model. The data labeled as Red and yellow is dominated by neutral ones resulting the model predicts many as neutral or becomes over-fitted. The model structure such as change the learning rate, number of epochs, or different dimensions in each layer, or add numbers or layers has not been

altered yet. Adjusting of model structure is possibly make the difference in terms of result.

4.6 Tools in Building a Model

- PyTorch is the library that is implemented to build three different models. It is the free open framework to build a machine leaning model, developed by the Facebook's AI Research lab.
- Colab Notebook is a platform that being used in the project. It is a cloud based Jupyter notebooks allows to corporation and integrated with Google drive.
- Pandas is a library used for Python programming. It provides operations and certain data structures in order to facilitate the manipulation of data table.

4.7 Mobile Application

To implement the cross-platform mobile application module, we utilize an Flutter and Dart, where in the beginning we have explored Ionic and Vue-JS as our initial choice. However, we have faced some difficulty with Ionic, which later we decided to change into Flutter and Dart. However, we have experimented both prototypes by using Ionic as well as Flutter. Nevertheless, our focuses and finished product was all based on Flutter and Dart [16].

4.7.1 Application Draft using Ionic and Vue-JS

Login Page and Register Page

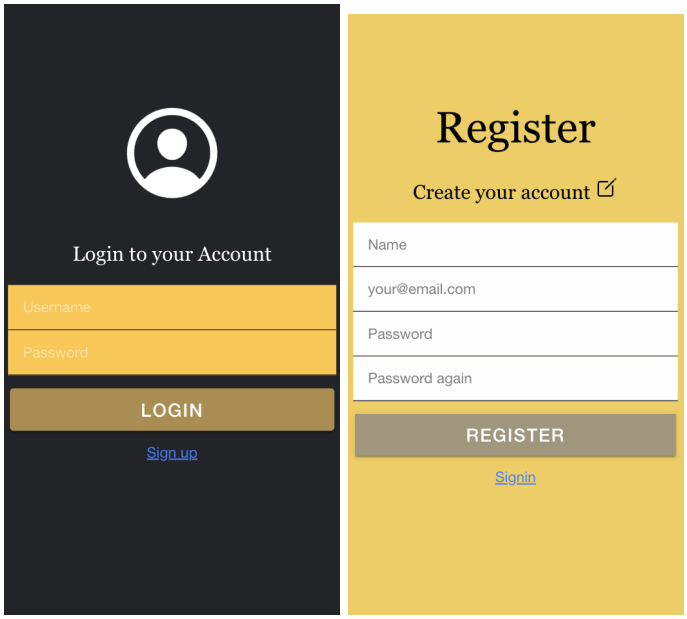

Figure 4.30: Login Page and Register Page

Introduction Page and Instruction Page

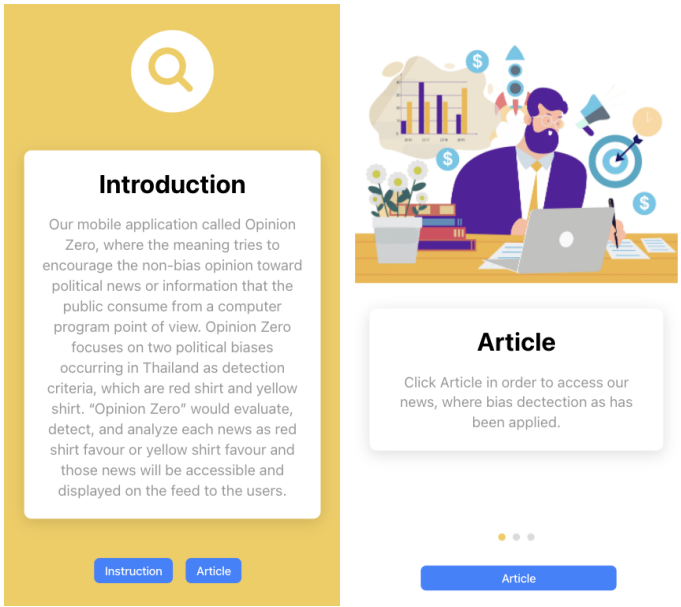

Figure 4.31: Introduction Page

Article Page and Profile Page

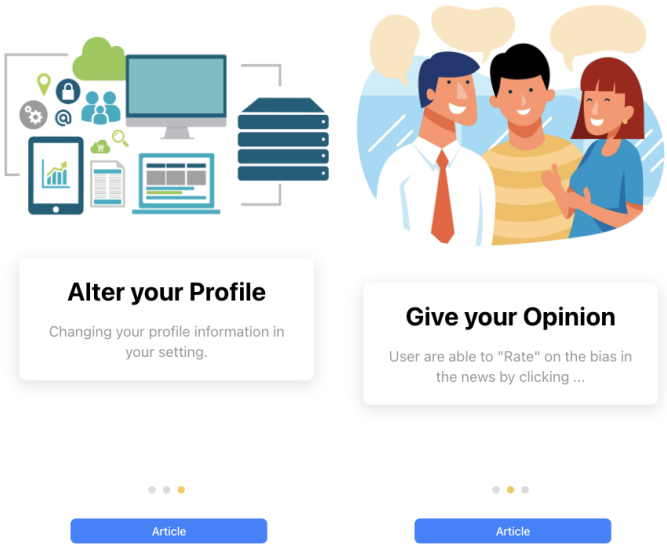

Figure 4.32: Instruction Page

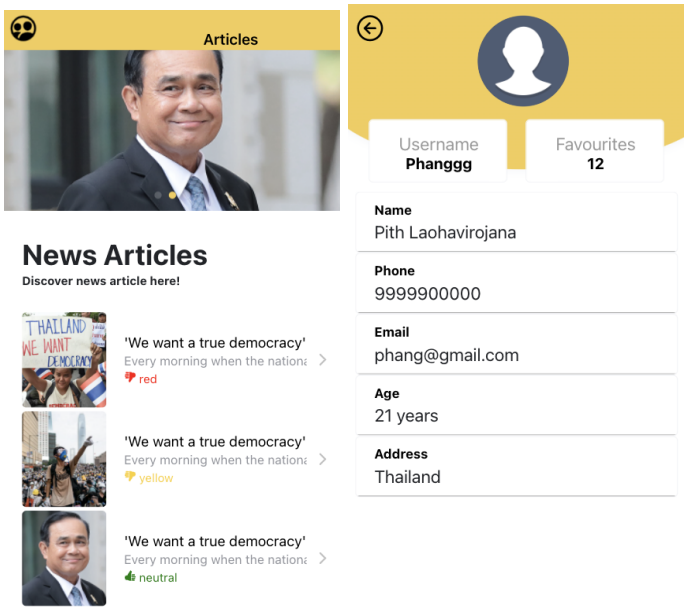

Figure 4.33: Article Page and Profile Page

4.8 Finished Prototype

In the finished prototype, flutter and dart are the primary framework utilised in front end development. For the backend implementation, the database is managed by Mongodb, which is non-relational document database where it adopts similar pattern like JSON.

In terms of the logic behind the mobile application, Python is being used.

4.8.1 Frontend Implementation

In this section, we will discuss about frontend Implementation.

Splash Screen, Login Page and Register Page

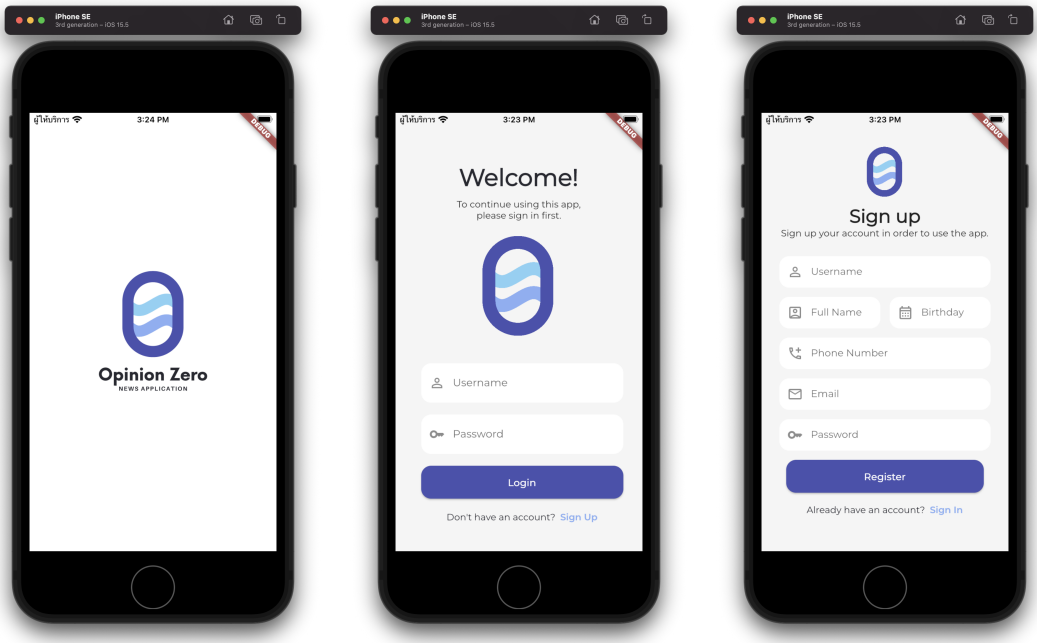

Figure 4.34: Splash Screen, Login Page and Register Page

Once the user open the application, the user will be introduced to the splash screen. Afterwards, the user will redirect to the login page where it asked to put username and password. If the user has not created an account, the user can click sign up and it will link to the registration page in order to register an account. Only authenticated user is allowed to view the news articles.

Home Page

Home Page composed with two sections: Breaking News and list of categories

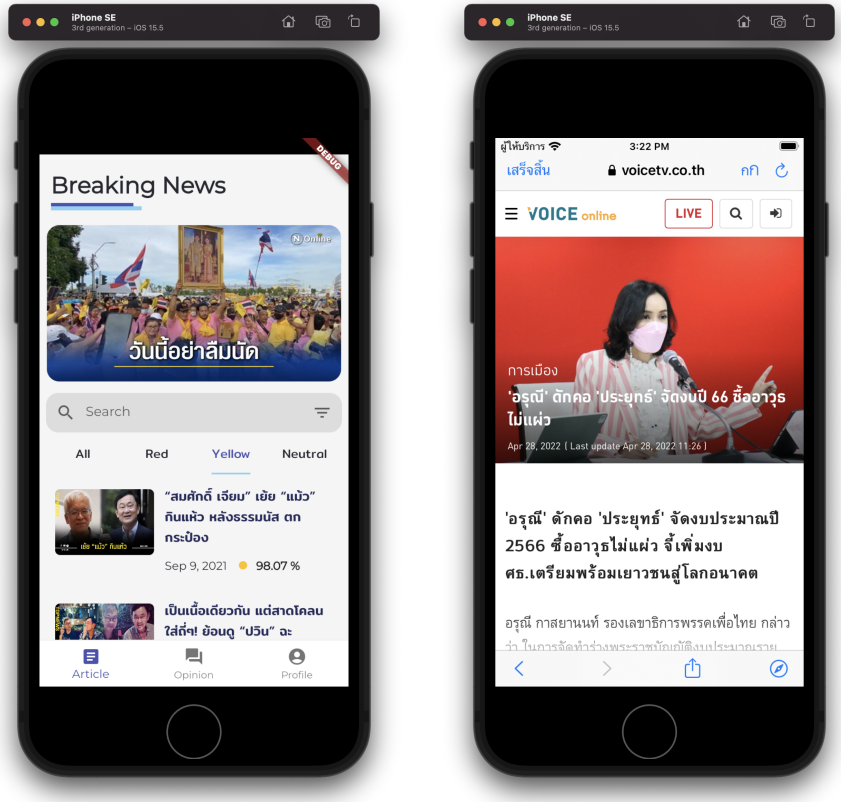

Figure 4.35: Home Page

of news. Where Breaking New is the above section using page news to contain breaking news images and information. As the user click into the image it will link into the that particular news article. Second section contain Tab-Bar where each category of news are store separately, which are all, red, yellow, and neutral. As the user clicked into that particular category, it will display accordingly. Nonetheless, each row of the news contain the title, percent likely to be red, yellow, and neutral, and the date of the news. The user can access the news article by clicking the news which will link to its article. Another feature in the home page the user can use the search bar to search for the desire news that the user wanted to find.

Add Opinion Page

In Add Opinion page, we provided news articles that have not been voted yet. Where the user can engaging in rating or voting the news according to their opinion or judgment in order for us to utilize those results and reinforce it into our model training.

The purpose of this page is to display the list of unvoted news, as the user click into that particular news they will be bring them to another screen page. This screen page contains new's title, description, date, read button (can link the user to news article) and as well as vote button. As the vote button get clicked, pop-up dialog appear and the user can decide what political bias category it stand.

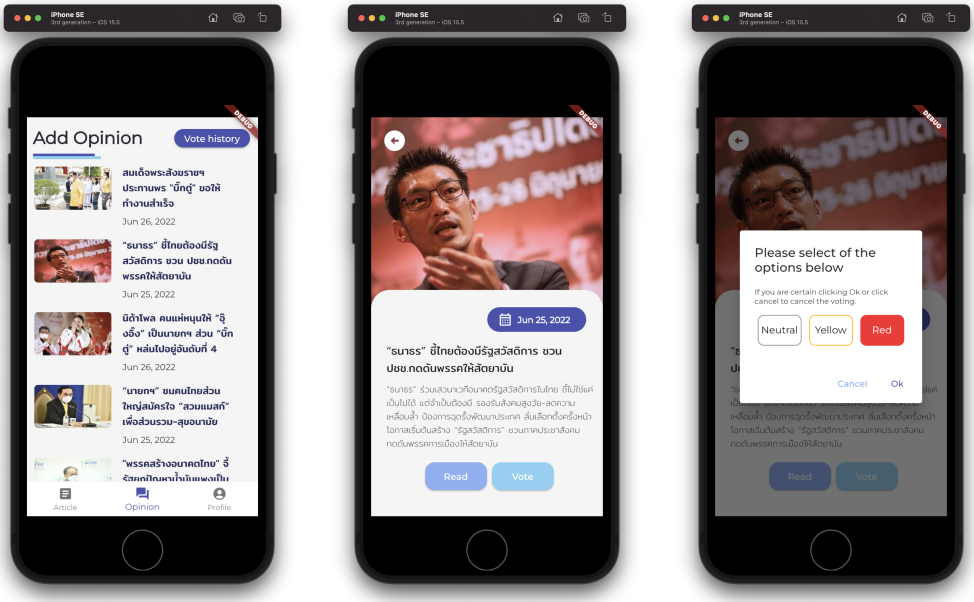

Figure 4.36: Add Opinion Page

Voted History Page

As the user are done with their voting, the voted articles will be store into voted History page. Where it will display the list of the voted news along with its title, date and the vote category that you approached. News article can be read again by clicking that particular row of that news.

Profile Page

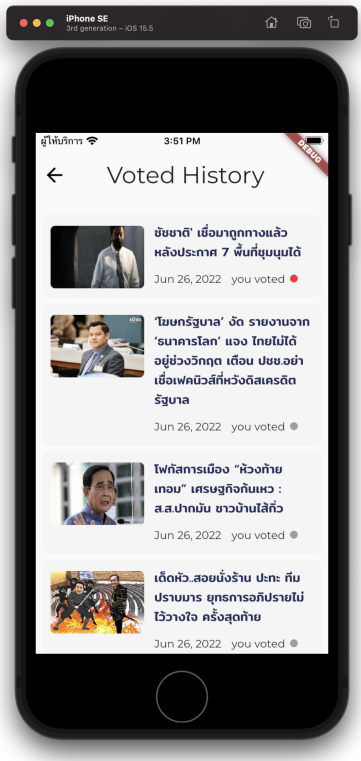

Figure 4.37: Voted History Page

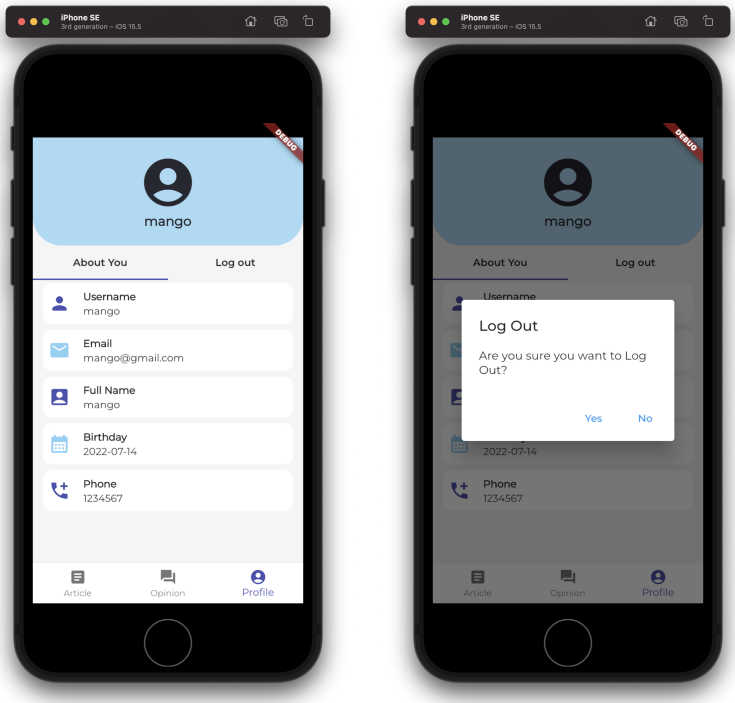

Figure 4.38: Profile Page

Profile page contain the information of the login user such as their email, username, birthday, full name and phone number. Moreover, login button are located in this page as well.

4.8.2 Backend Implementation

In this section, we will discuss about backend Implementation.

4.8.2.1 Database: MongoDb

MongoDb is the document-oriented database where it is different from traditional SQL row-table style. Instead of storing data as table and row, it stores data in key value pair where its key is uniquely identified and the value can be any data types. [1]

4.8.2.2 Database: ER Diagram

The Diagram in figure 4.39 is referred to as ER-Diagram or Entity Relationship Diagram. ER-Diagram shows the relationship of entity sets that are contained in a database. Our database consists of four entities: test_user, test_vote, test_news, and test_opinions. Attributes that are contained in test_user are id, email, username, password, full name, birthday, and phone. Id, user_id, new_id, and xclass are attributes of the test_vote entity. id, title, article, link, xclass, predicted, img, date, and percent are test_news's attributes. test_opinions contain id, title, article, description, link, img, and date as an attribute. Where cardinality between test_user and test_vote is (optional) one to (optional) many, means one user can have multiple votes. However, the relationship between test_vote and test_user is (optional) many to (optional) one means each particular vote belongs to only one particular user him/herself. Test_vote and test_opinions maintain one to many cardinality since one category of vote can be used by many news articles, for example, multiple news can vote political bias toward red. Vise versa, many to one is a cardinality between test_opinions and Test_vote, where that multiple particular article can only be voted as either red, yellow or neutral.

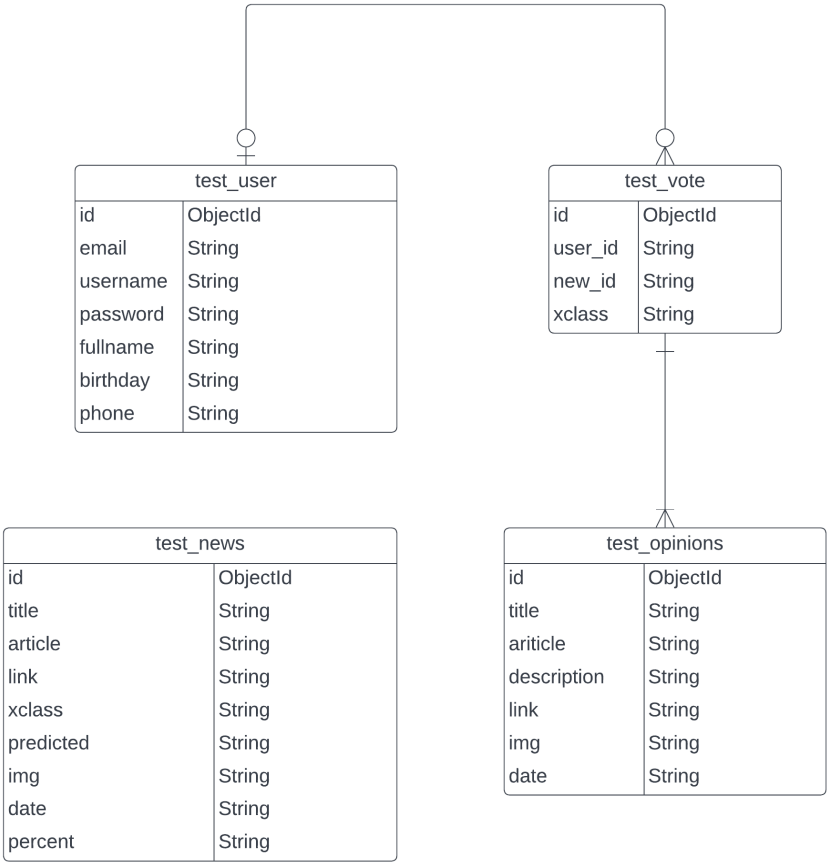

Figure 4.39: diagram Page

4.8.2.3 Logic: Python

We retrieve data from MongoDb by using api. To build efficient api, we also installed FastApi to help us create api. FastApi is a robust and modern web framework for creating APIS in Python. It offers an easy accessible api from numbers of users as well as it is faster to run. For our web server, we installed Uvicorn which is the web sever specifically for Python. Another benefit of fast api is that it automatically generates api document. The picture below (figure 4.40) is our api document

The first api is called users. This api is primary for creating a new user. Front end called this api when in the registration page. If the username has already been created, the back end will send error message to front end and does not create user. However, if the user is able to be created, every user's information will be inserted in the

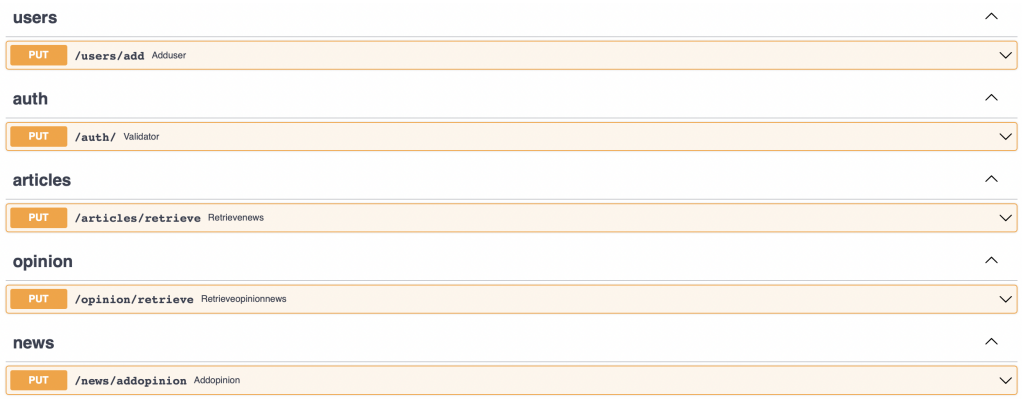

Figure 4.40: diagram Page

test_user collection. The second api is called auth. This api is for authenticate the user who wish to login. If there is no username in test_user collection, or the password does not match, the back end will return the success status as fail. The third api is articles. This api will return the list of all articles that has already been labeled by our model in test_news collection, the list of articles that the label is red, the list of articles that the label is yellow, and the list of articles that the label is neutral. The fourth api is called opinion. This api will return all news that has not been labeled by the model, but waiting for the user to help label as well as filtered out the news that user has already voted. For the news api, this api will receive the voted the result of by the current user from the front end as well as the new_id which will identify what news the current user has already voted.

4.8.2.4 User's feedback on Mobile Application

After we have completed our mobile application, we have gathered the user's opinion and feedback about the UI and features of our application. From what we perceive users are satisfied with UI as it is easy to understand and easy to use. It retains and

represents typical News applications. However, some users wish to see more up-to-date news articles as well as some videos relevant to the news article. Moreover, some users also demand the filtration system where the application should filter the latest news to the oldest and vice versa. Some groups of users are also concerned about the security of personal information that was given when they registered. Nonetheless, most users are thrilled about the idea of bias news detection application since it is quite unique and reminds them of how media is easily exposed toward bias and inaccurate ideas. Also, by categorizing each news based on different political biases, it raises user's awareness about what to look for when consuming any kind of media.

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1 Future work and improvement on Model

As mentioned, the major difficulty is on data collection. This is because it is time consuming to first collect data from many different news sources. Secondly, some news website prevents scraping data when there are too many requests; hence, it results limiting numbers of news that could be collected at once. On top of that, the structure of HTML tag such as the name of the class or the id of the tag in each news station are completely different. Therefore, it is inevitable to write web crawler for each new sources separately. Moreover, some websites such as voiceTV and dailyNews are not implemented pagination. To load more news, they implements what called lazy load. When the lazy load performs, the website URL is changes every time making crawling even more challenging. Hence, our advisor suggested that instead of training only news data, we can train on normal Thai text for the model to learn what kind of text is attacking and what kind of text is praising. Afterwards, the model can learn from the news data set to classify even further whether is this praise for neutral, red or yellow shirt or attack on red or yellow shirt. By doing this, the model can tell more than just a political class its belong, but also the tone of the text.

5.2 Future work and improvement on Mobile Application

Evidently, our mobile application is just a prototype mobile application where all of our main features are completed and workable. However, there's still a small details that we can further implement for example in term of cryptography such as hash and salt. Where hash turn your password in to a string of text that always have the same length, it work only way (can't decrepit back into original data) and it is an algorithms that are

optimised for speed. Where adding salt can ensure that hash is always unique even the user share same password. Another details that we can add on is user's favorite, as the user are done reading the article and might want to mark it as favorite, there will be an option for the user to select that article as a favorite. In addition, we would like to improve on voting method, where instead of the user reading our provided article and vote, the user can actually input an outside article and we will return the bias prediction. To add on the voted feature on the mobile application, the verdict vote result of all the users will come from the majority vote of overall users. Then, the news will be labeled and used to train the model in the future.

5.3 Conclusion

In conclusion, the primary solution to tackle Media bias is building a news mobile application as well as creating a bias detection model analyzing each news, follow three criterion which are neutral, yellow shirt and red shirt. Several method are used in order to train and teach the model to be able to detect those bias criterion such as TF-IDF, FastText, BERT, BERT with Siamese, BERT with Triplet loss, BERT with Tanh, and BERT with Softmax. The final model approach is BERT with Siamese since it yields the best performance estimated 82 percent as well as return a percentage of how likely each news belong to each criterion. For building mobile application, Opinion Zero, the fundamental tools have been implemented in front end are Flutter and Dart. For back end was implemented using MongoDB and Python. As far as we perceived, our model prediction percentage are highly acceptable thus it can mitigate the issue as well as raise awareness the media is not entirely objective although the performance of the model is not hundred percent effective.

REFERENCES

[1] Kyle Banker, Douglas Garrett, Peter Bakkum, and Shaun Verch. *MongoDB in action: covers MongoDB version 3.0*. Simon and Schuster, 2016.

[2] Martha R Bellows. *Exploration of classifying sentence bias in news articles with machine learning models*. University of Rhode Island, 2018.

[3] Elad Hoffer and Nir Ailon. Deep metric learning using triplet network. In *International workshop on similarity-based pattern recognition*, pages 84–92. Springer, 2015.

[4] Steeve Huang. Word2vec and fasttext word embedding with gensim, 2018.

[5] Taku Kudo and John Richardson. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*, 2018.

[6] Minghui Liao, Baoguang Shi, Xiang Bai, Xinggang Wang, and Wenyu Liu. Textboxes: A fast text detector with a single deep neural network. In *Thirty-first AAAI conference on artificial intelligence*, 2017.

[7] Duncan McCargo. New media, new partisanship: Divided virtual politics in and beyond thailand. *International Journal of Communication*, 11:4138–4157, 2017.

[8] Iaroslav Melekhov, Juho Kannala, and Esa Rahtu. Siamese network features for image matching. In *2016 23rd international conference on pattern recognition (ICPR)*, pages 378–383. IEEE, 2016.

[9] Sendhil Mullainathan and Andrei Shleifer. Media bias, 2002.

[10] Wannaphong Phatthiyaphaibun, Korakot Chaovavanich, Charin Polpanumas, Arthit Suriyawongkul, Lalita Lowphansirikul, and Pattarawat Chormai. Pythainlp: Thai natural language processing in python. *URL: http://doi.org/10.5281/zenodo*, 3519354, 2016.

[11] Juan Ramos et al. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, volume 242, pages 29–48. Citeseer, 2003.

[12] Sagar Sharma, Simone Sharma, and Anidhya Athaiya. Activation functions in neural networks. *towards data science*, 6(12):310–316, 2017.

[13] Yanchuan Sim, Brice DL Acree, Justin H Gross, and Noah A Smith. Measuring ideological proportions in political speeches. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 91–101, 2013.

[14] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[15] Minh Vu. Political news bias detection using machine learning. *URL: https://pdfs. semanticscholar. org/8445/2eb068bdfe7d5809734a5da8f5c7d10bebfa. pdf*, 2017.

[16] Eric Windmill. *Flutter in action*. Simon and Schuster, 2020.