NUS
National University
of Singapore

# IRS-PM-ISY5001
# PROJECT REPORT

**SureBoT – An Intelligent End-To-End Fact Checking Telegram BOT**

**TEAM MEMBERS**

KUEK YONG JIE ADRIEL (A0229985H) - 1

LAVANYA M (A0229979B) - 2

CHUA HAO ZI (A0229960W)- 3

FRANCIS LOUIS (A0079985B)- 4

For the fulfilment of the Practice Module under the Intelligent Reasoning Systems (ISY5001) Graduate Certificate
as part of the requirements for the Master of Technology in Intelligent Systems

## EXECUTIVE SUMMARY & ABSTRACT

With the rapid advancement in technology and the push for digitalisation around the globe, news media outlets have gradually moved away from traditional print media to digital online news offerings. People can now get access to live news updates with just a simple click on their smart devices such as handphones or tablets. However, within the massive amount information on the Internet, not only are there genuine news, but there also exist a sizeable amount of fake news or misinformation shared for the purpose of deceiving and misguidance. When individuals cannot distinguish between genuine and fake news, there will be widespread sharing and spreading of false information due to its virality nature. This, unfortunately, can lead to major repercussions in a country's economic, political and social areas.

Currently, to verify news of doubtful content, one must follow a series of steps to gather information online with authentic sources such as Factcheck.org and gov.sg/Factually, or news media providers such as The Straits Times (ST) and Channel News Asia (CNA). During information gathering, one would need to read and understand the article-in-context. In addition, there may also be the need to cross-reference to multiple sources to reach a decision. Such a verification process will take considerable time and not many people may be apprised of such methodology, especially the elderly and those who are not technology-savvy.

Our team of four would like to resolve this global issue by creating an intelligent chatbot to provide an end-to-end automated fact-checking tool. We established the chatbot on a common and popular messaging platform like Telegram to increase the accessibility to the masses. The beauty of the chatbot lies in its simplicity, ease-of-use and 24/7 availability. One only needs to input a claim to fact-check and the chatbot takes care of everything else. This minimalistic approach ensures that even the elderly will be able to use it as well.

By adopting the techniques acquired during the course, we designed the chatbot based on a cognitive framework that takes in a user input, applies feature extraction and knowledge reasoning techniques to produce a classification output that decides whether the supporting evidence "SUPPORTS" or "REFUTES" the input claim.

We believe that this chatbot has great value, as the shift towards digital news consumption through online will only increase exponentially. The chances of a person intentionally or unintentionally proliferating misinformation will only become higher. Through this project, we aim to provide to the masses a tool to combat the rise of fake news and improve digital literacy. That, in turn, can further reduce or eliminate negative societal impacts such as radicalisation, religious or racial discord.

**CONTENTS**

# 1. INTRODUCTION

## 1.1. PROBLEM DESCRIPTION & BACKGROUND

The pandemic year 2020 has disrupted and changed many lives and industries across the world. In Singapore, the new "normal" has brought about rapid changes and shifts towards digitalisation and technology. While this has been the silver-lining in the crisis, it has further exacerbated problems that comes with the pervasive use of such technology.

One such problem that continues to plague us today is the rapid rise and proliferation of misinformation and fake news. As we continue to be highly connected on social media, and messaging applications, it is not likely that one may be immune to the exposure of misinformation. A survey study conducted by the National Centre of Infectious Diseases (NCID) during the COVID-19 outbreak publishes that at least 6 in 10 people in Singapore have received fake COVID-19 news (Channel News Asia, 21 May 2020).

While Singaporeans, many of whom pride themselves as technology-savvy, struggled to distinguish the validity of a piece of news article without additional information. An independent study conducted by the Institute of Policy Studies found that unfortunately, most Singaporeans were unable to detect false information. With 73.1% of seniors and low-income dwellers failing to discern the truth, followed by 77.7% of the middle-aged group who are well-read and knowledgeable, and even up to 46.9% of youths and young adults who are technology-savvy (Todayonline.com, 17 Dec 2020).
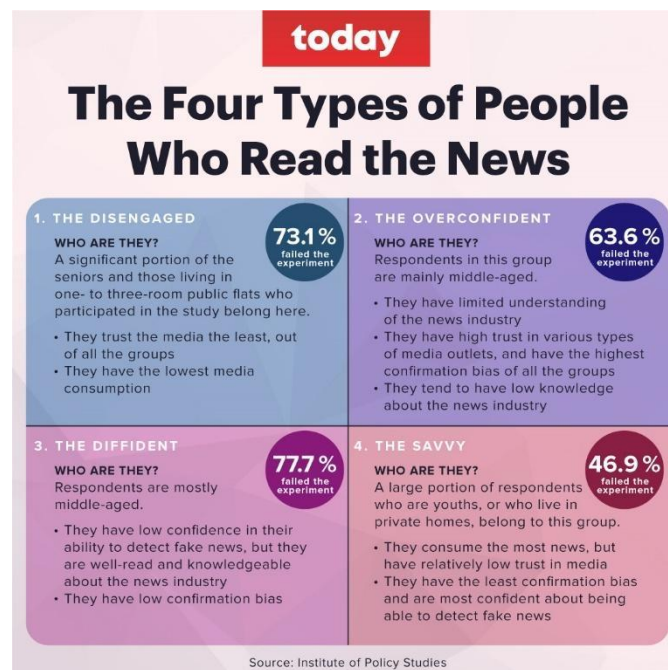


*Figure 1: Majority of Singaporeans failed to identify false information when put to the test (source: Institute of Policy Studies, extracted from Todayonline.com, 17 Dec 2020)*

The main reasons cited by the studies, showed over-confidence and diffidence leading to the failure in the detection of false information. The impact of misinformation and online falsehoods has a severe repercussion on the people's way of life and businesses. The government highlighted 6 crippling factors that could be brought about by misinformation. This include being used as a tool for foreign interference, undermining the confidence in public institutions, causing public health scares, provoking violence, destabilising of countries, and finally, creating discord and division within the society.

*Figure 2: Online falsehood as a global problem (source: Gov.sg proposed Select Committee on the problem of deliberate online falsehoods)*

In a particular recent example of one such misinformation propagated as a chain message through popular messaging application platform WhatsApp, inaccurately misled on of the dangers of the COVID-19 vaccination by causing death in older folks.
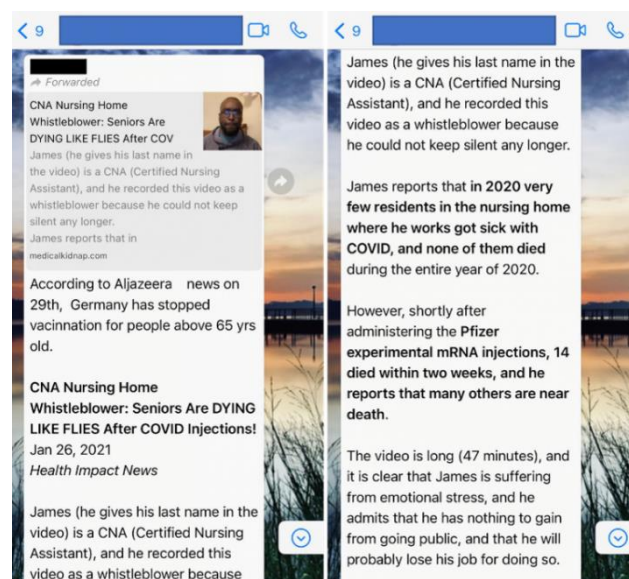


*Figure 3: CNA Nursing Home Whistleblower: Seniors are dying like flies after COVID vaccination (source: https://blackdotresearch.sg/covid19-vaccine-elderly-cna-nursing-home/)*

While the cost of forwarding the message to others is negligible, the downstream impact of the message generating fear, distrust and contention is severe and not to be taken lightly. This would impact the government's effort in the National Vaccination Programme to quickly build up a safe and secure population, in preparation for the re-opening of borders for economic recovery. This will especially affect the older generation population whom may be lesser informed and possessing lower digital literacy.

In the recent years, many companies have started to take a stand to combat the proliferation of false information

on the internet. From small independent fact-checking firms to large technology conglomerates such as Facebook and Google, there is a concerted effort to provide substantiated information validation. However, many of the validation processes are largely manual, involving intricate levels of cross-validation and examination performed by experts. There are still limitations on the number of articles that can be validated as compared to the overwhelming quantity of misinformation floating on the internet today. Moreover, due to data privacy restrictions and information policies, there would still be many more hurdles moving forward in pushing for information validity in countries.

## 1.2. PROJECT OBJECTIVE

The National Library Board (NLB) educates the public in digital literacy and proper fact-checking techniques using the S.U.R.E. steps highlighted as Source, Understand, Research and Evaluate. This includes a series of information retrieval, reading, understanding, cross-referencing and comparison, and finally, evaluation and decision-making. In particular, the framework encourages fact-checking across multiple articles and sources to ensure an unbiased viewpoint and perspective.



*Figure 4: Fact-check across multiple sources using NLB's S.U.R.E. steps (source https://sure.nlb.gov.sg/blog/seniors/sn0004)*

While the methodology is sound and rigorous, many may find the task of information validation menial and time-consuming. Therefore, making evaluations and decisions about a particular piece of news information purely based on one's personal confirmation bias. This, unfortunately as shown in earlier studies, has a low success rate in identifying false information.

The team's project objective is therefore, to mimic the basic cognitive framework in resource gathering, data cleaning, contextual understanding, and decision rules to automate an end-to-end fact-checking pipeline realised as an interactive chatbot that is based on a popular messaging platform that many people have accessibility to.
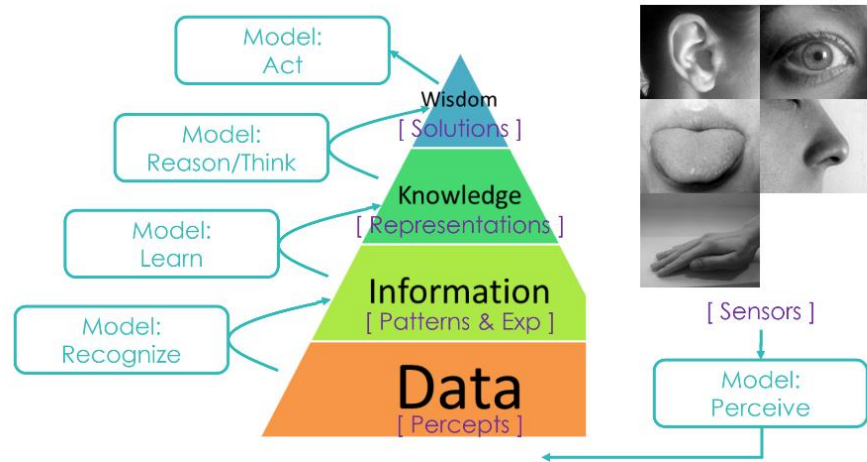
*Figure 5: Intelligent Systems Cognition Process (NUS-ISS Intelligent Reasoning System)*

In similar fashion, we based the pipeline on the cognition process in a model view of an intelligent system. Starting with the perception of the real-world input, and in our case, it refers to the input query claim that a user would like to fact check. This is followed by the data mining and information retrieval block, which forms the basis of which the system can draw insights from. We will then apply a series of textual pre-processing and feature extraction through contextual embeddings to exploit the latent information within the data. After which, we stack the knowledge representation block in sequence to enable the model to reason and arrive at a decision through a fully-connected graph neural network. Finally, we perform a decision with the use of a rule-based aggregator to provide the output for the user's query.

## 1.3.  PROPOSED MEASUREMENT METRICS

To evaluate on the overall performance of the intelligent chatbot built, we will assess the system's metrics based on the following 2 parameters:

- Time-taken for fact-checking and validation process.

- Relevancy accuracy score based on the number of articles found.

As highlighted in the previous section, the framework for fact-checking and cross-referencing across multiple articles is a rigorous and time-consuming task. We aim to benchmark the system performance with the human-level time taken to perform a fact-checking task.  The current performance baseline for human-level fact-checking task is summarised below. We assume that the test subject has sufficient proficiency in digital literacy and is overall technology-savvy.

| Task-level difficulty (Article Fact-Checking) | Total Time Taken (Seconds) |
|---|---|
| Easy | 510 |
| Difficult | 1500 |

(Note: Total time taken includes the reading of the entire article and cross-referencing with at least 2 or more relevant sources)

*Table 1 Human-level benchmark on Fact-Checking Task*

Due to the lack of a local domain news article evidence-claim dataset that is readily available.  We target to baseline the system's performance based on real-world queries from multiple users to obtain a relevancy accuracy score matrix. The target score output would be a number from 0 to 10, with 0 being the least relevant and 10 being the most relevant. The scores will be collected and aggregated across different participants with different backgrounds and age-groups, performing real-world queries on current affairs based on the task-level difficulty defined in Table 1 above.

## 2. SYSTEM SOLUTION

### 2.1. OVERVIEW

The team's approach is to exploit off-the-shelf pre-trained models in Abstractive Text Summarisation, Text Semantic Similarity comparison and Fact Verification as the fundamental building blocks for the end-to-end Fact-Checking pipeline. The aim is to mimic and automate the human cognitive process highlighted in the earlier section of data acquisition, extraction of key information, knowledge representation, and decision-making in Figure 5.

### 2.2. KNOWLEDGE MODELLING AND REPRESENTATION

In building the knowledge model for the purpose of fact validation, we leverage on the usage of the FEVER shared task (Thorne et al., 2018), which was hosted as a competition on Codalab (https://competitions.codalab.org/competitions/18814). The task challenges participants to develop automatic fact-checking systems to cross-check the veracity of human-generated claims with extracted evidence from Wikipedia. The dataset consists of 185,455 annotated claims with a set of 5,416,537 Wikipedia documents from the June 2017 Wikipedia dump. We adopted the statistics distribution of the shared task dataset as follows:

| Split | SUPPORTED | REFUTED | NOT ENOUGH INFORMATION (NEI) |
|---|---|---|---|
| Train | 80,035 | 29,775 | 35,639 |
| Development | 6,666 | 6,666 | 6,666 |
| Test | 6,666 | 6,666 | 6,666 |

*Table 2 FEVER Shared Task Dataset*

We then utilised a BERT-pair fine-tuning system which encodes each evidence-claim pair independently and then aggregates the result. In the training phase, the ground truth evidence for SUPPORTED and REFUTED claims, and retrieved claims for NEI claims are selected. In the test phase, label prediction is performed for all retrieved evidence-claim pairs. An aggregator is then used to obtain the final claim label. The BERT-pair model for sentence encoding is then trained for one epoch. From literature survey, it was found that the BERT-pair fine tuning on the FEVER shared dataset was critical in capturing semantic and logical relations between the evidence and claim pairs. Sentence embeddings extracted from generic encoder-decoder methods such as ELMo, CNN showed that results on the Development Test set was almost close to random guess.

We therefore concluded that the domain knowledge representation is crucial in ensuring model accuracy, especially when we adapt the knowledge information across to apply on our local news claim-evidence context for fact checking and verification.

### 2.3. SYSTEM ARCHITECTURE

### 2.3.1. TELEGRAM BOT SERVER PLATFORM

We built the system architecture based on the Telegram Bot server platform due to its open-sourced framework and ease-of-use application interface. Telegram not only combines the speed of WhatsApp messaging with Snapchat's ephemerality to allow information to expire over time, but it also provides end-to-end encrypted security for its users.

Our Bot server is a Flask Application (a Python web framework) that processes the GET and POST requests received through our webhook whenever user sends a message.

Flask comes with a simple Web server that will not be able to handle multiple requests at a time as it only has a single worker to accept one HTTP connection at a time. Hence, we use Gunicorn that allows us to create multiple workers and sockets that handle HTTP requests by calling the Flask App object. We also have a reverse proxy server, NGINX that sits in front of Gunicorn to handle load balancing and add a layer of security. Additionally, we also use Redis and Celery to manage our background processes. Redis is an in-memory data store that holds information and status of each task and Celery is a powerful task queue that handles background tasks and schedules that can run independently.

The Bot server is hosted in a GCE (Google Compute Engine) instance. GCE is an IaaS (Infrastructure as a Service) provided by Google and enables users to launch virtual machines. We have chosen an instance of N2 Machine Type and an Ubuntu 20.04 LTS Boot disk.

Whenever a user send message to our Bot, we get a request to our Flask App and after processing it we send a message back to the user by calling one of the Telegram Bot API's. Each user is identified by their unique chat ID.

Our fact-checking pipeline is a computationally intensive process and will take some time to provide an inference to the user query. Hence, we have added Celery to handle the pipeline execution as a background task, so that it does not block the Flask App and the user receives an acknowledgement almost immediately, informing the user that their query is being processed. After the Flask App extracts the user query, it adds it to the Redis database with the user's chat ID. The Celery worker takes tasks from the Redis queue and executes it.

Because of the constraints of our VM Instance, we have only enabled Celery to execute one task at a time. When there are multiple queries, Flask will process the request, pass the job to the Redis queue, and send a message to the user with their queue number. Once pipeline execution is completed for a query, we respond back to the user with the results of the fact-checking.
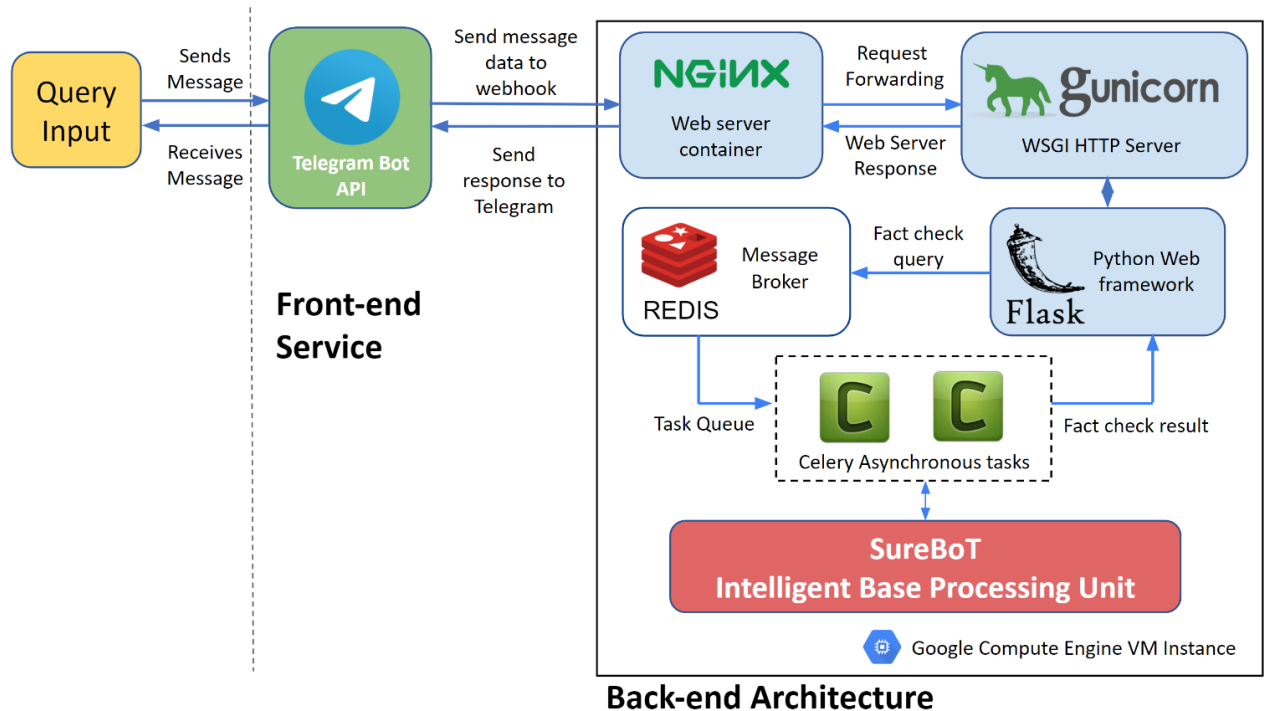


*Figure 6: SureBoT Telegram Bot Server Platform*

## 2.3.2. INTELLIGENT BASE PROCESSING UNIT

We utilize a 5-stage sequential pipeline beginning with the target input query. The 5 stages are highlighted as follows:

- Article Retrieval

- Text Summarisation

- Evidence Selection

- Claim Verification

- Score Aggregation

The SureBoT base architecture is built upon a series of Natural Language Processing (NLP) techniques and models for the purpose of knowledge extraction, representation, and decision-making. The pipeline, as show below, forms the intelligent Base Processing Unit (iBPU) that is part of the larger Query-Answer SureBoT system.
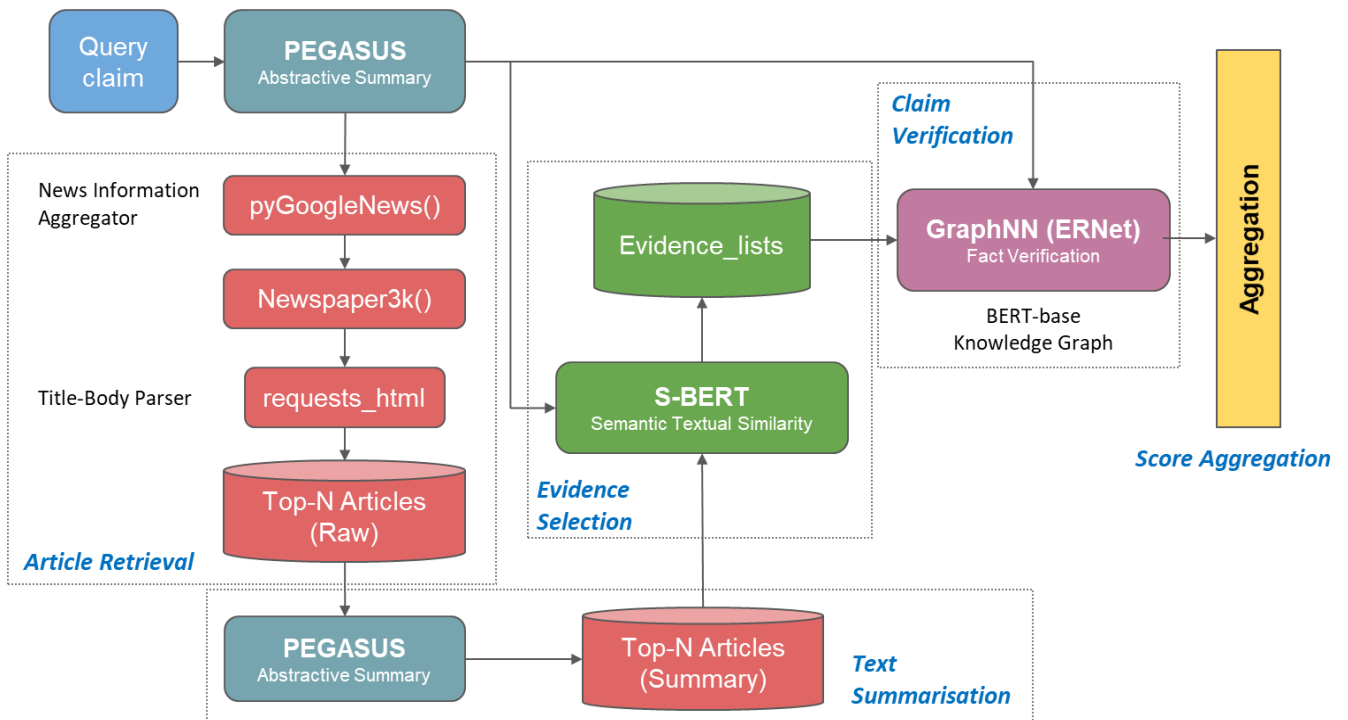


*Figure 7: SureBoT Fact Checking Pipeline Architecture*

## 2.3.2.1.  Article Retrieval

The 1st stage of the processing pipeline involves a series of NLP textual pre-processing tasks such as tokenisation, and data-cleaning such as symbol and artefacts removal. Depending on the token length of the input query claim, we will perform abstractive summarisation for condensation when the input text size is too large. As the results of the Article Retrieval process impinges greatly on the quality of the input query claim, this pre-processing stage is required to ensure high quality downstream results.

Due to the rapid and dynamic rate at which information gets updated on the internet, for an accurate and fair assessment of a piece of information, the first stage Article Retrieval process utilizes a real-time news article web-scrapper based on the pyGoogleNews and newspaper3k APIs. Google News allows for most relevant search results based on an input query to sorted. We then proceed to retrieve Top-N articles where N is set to 5 as default. Selecting a larger N allows for a wider scope search at the expanse of a longer verification processing time.

## 2.3.2.2.  Text Summarisation

The second stage Text Summarisation utilizes an abstractive downstream summarization model based on a pretraining large Transformer-based encoder-decoder model framework named PEGASUS (Zhang Jingqing et al. 2020) that is trained on the CNN/DailyMail (Hermann et al. 2015) dataset containing 93k articles from CNN and 220k articles from the Daily Mail newspapers. The framework utilises a Gap Sentence Generation (GSG)

pretraining together with the downstream task specific dataset to provide robust and linguistically fluent abstractive summaries for news articles.

The main objective for the second stage is to summarise the salient points within the selected Top-N news articles accurately from the first stage. This would later then serve as the input baseline for Claim Verification Stage subsequently. We set hyperparameter beam-alpha as 1.5 to generate enough summarised sentences for the n-sentence input parameter into the Graph-based fact verification model. This is important to minimise inconsistency between the input of the fact verification model that was used during training with the FEVER dataset and inference.

### 2.3.2.3.  Evidence Selection

The third stage Evidence Selection provides a filtering decision stage to only select the most relevant articles that corresponds to the input query claim. In this stage, we utilise the popular state-of-the-art Sentence-BERT Siamese network (Reimers and Gurevych, 2019) to derive semantically rich sentence embeddings. Sentence-BERT or SBERT is computationally efficient and derives a fixed sized sentence embedding utilising a Siamese network for weights updating. The network structure uses a Regression Objective Function to compute the cosine-similarity between the 2 sentence embeddings with a Mean-Square-Error (MSE) loss as the objective function.
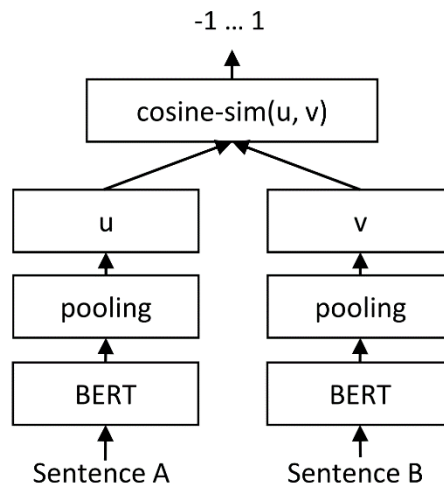


*Figure 8: SBERT Inference Architecture with regression objective function*

To filter noisy data input, and provide relevant and accurate topics, we employ a global threshold filter score of 0.4 to only select articles with high semantic similarity scores with the input query claim. As the output from the Article Retrieval stage could yield a large number of topics upon scrapping, the filtering stage is critical to maintain and select articles are highly relevant for cross-examination and fact-checking.

### 2.3.2.4.  Claim Verification

The fourth stage utilises a novel fact verification model based on a fine-tuned BERT-pair encoder trained on the FEVER shared task dataset and a graph neural network model (Zhou jie et. al, 2019) to verify all input evidence against the target input query claim.

The main idea for the graph neural network architecture is that mainly claims require to simultaneously integrate and reason over several pieces of evidence for verification. Building a fully-connected evidence graph together with an aggregator allows for information to be flowed and transferred throughout the network, thereby allowing sufficient relational and logical information to propagate through. Consider the following trivial example in which contextual information could be pieced together by flowing information through the sentence chain.

---

**Input_claim** = {'*Tan Ah Seng* is a **Singaporean**.'}

**Input_Evidences** = {'*Mount Elizabeth Hospital* is a general hospital located in **Novena**.',

'*People who are born in Singapore or citizens of Singapore are called **Singaporeans**.*',

'*Novena* is considered as an upscale district in *Singapore*.',

'*Tan Ah Seng* was born in 1985 at the *Mount Elizabeth Hospital*.'}

---

The input claim requires the connection of the subject "Tan Ah Seng" to be made to the description of "Singaporean". Based on the input evidence sentences which form the basis as the comparator, we can observe that information from multiple sentences is required to form the linkage {Tan Ah Seng → Mount Elizabeth Hospital → Novena → Singapore → Singaporean}. Therefore, methods dealing with just claim-evidence pairs, or simply evidence concatenation used in Natural Language Inference (NLI) would not be sufficient to extract salient information linkages which may be critical in the claim verification.

### 2.3.2.4.1. Evidence Reasoning Net (ERNet)

At the heart of the Claim Verification Stage is the Evidence-Reasoning Network (ERNet), which is a fully-connected attention network graph with input evidence sentences and claim as input nodes. Before the input claim and evidence are fed into the graph network, the knowledge representation model BERT-pair sentence encoder is first utilised to obtain sentence embeddings.
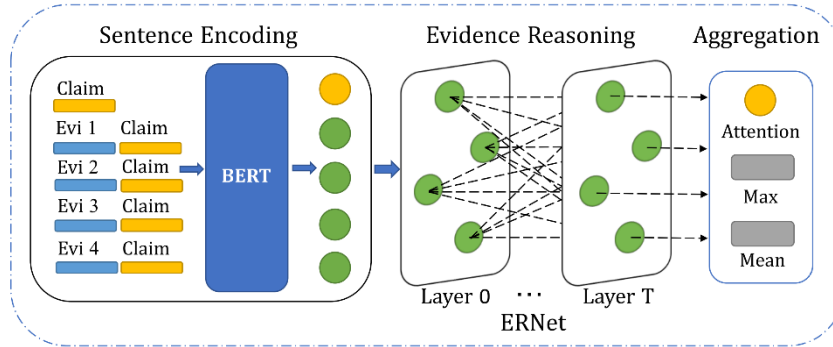


*Figure 9: Claim Verification Stage*

To encourage information propagation across the network, a self-loop within every node is created to facilitate the transfer of information. The hidden states of nodes at layer *t*, can be represented as:

$$h^t = \{h_1^t, h_2^t, \dots, h_N^t\}, \qquad \textbf{(1)}$$

*where* $h_i^t \in R^{F \times 1}$ and $F$ is the number of features in each node

Next a Multi-Layer Perceptron (MLP) is used to compute the attention coefficients between a node $i$ and its neighbour $j (j \in N_i)$:

$$p_{ij} = W_1^{t-1} \left( ReLU \left( W_0^{t-1} \left( h_i^{t-1} \parallel h_j^{t-1} \right) \right) \right), \qquad \textbf{(2)}$$

*where* $N_i$ denotes the set of neighbours of node $i$, $W_0^{t-1} \in R^{H \times 2F}$ and $W_1^{t-1} \in R^{1 \times H}$ are weight matrices. In our case, the representation of input claim embeddings is defined as $h_i^{t-1}$ while the evidence hidden states are represented

by $h_j^{t-1}$.

The coefficients are then normalised by a softmax function:

$$\alpha_{ij} = softmax_j(p_{ij}) = \frac{\exp(p_{ij})}{\sum_{k \in \mathcal{N}_i} \exp(p_{ik})}, \quad \textbf{(3)}$$

The features for node $i$ at layer $t$ are computed using a linear combination of neighbour features from the normalised attention coefficients:

$$\boldsymbol{h}_i^t = \sum_{j \in \mathcal{N}^i} \alpha_{ij} \boldsymbol{h}_i^{t-1}, \quad \textbf{(4)}$$

In addition, by stacking graph layers $T$ of ERNET, allows for information to be repetitively propagated over the same sentences to retrieve additional salient information. The final portion of the reasoning framework involves an attention-based aggregator in which information is gathered from the different nodes to obtain the final hidden state $o \in R^{F \times 1}$ .

### 2.3.2.4.2.  Hyperparameter Settings

For hyperparameter settings, we set batch size to 256, the number of features $F$ to 768 and the dimension of weight matrices $H$ to 64. The model was trained to minimise negative log-likelihood loss on the predicted label utilising Adam Optimiser (Kingma and Ba, 2015) with an initial learning rate of 5e-3 and L2 weight decay of 5e-4. An early stopping strategy with patience of 20 epochs was also put in place to prevent model overfitting. Finally, we found that a graph network with layers $T = 2$ provides the most optimal results, with the stacking of more layers $T = 3, 4, ...,$ not yielding any improvements.

The output of the ERNet produces a 3-way classification score which includes "SUPPORTED", "REFUTED" and "NOT ENOUGH INFORMATION" (NEI). For the purpose of generating a meaningful verification output for SureBoT, we drop the NEI class and force the output to a binary classifier.

### 2.3.2.4.3.  Network Attention – Explainable Predictions

One of the critical components that ERNet utilises is the attention mechanism. The attention coefficients determine the salient components within the sentence nodes that form information linkages. In our model study and analysis, we were able to extract the attention map from the graph layers and display as a heatmap to provide a visual overview as to why a certain classification prediction was made. To showcase such an analysis, we consider the following input_claim and input_evidences on a sample real-life case of misinformation that was propagated on popular messaging platform WhatsApp in 2020:

---

*Input_claim* = {'A nurse in the states has just had the vaccine and she died 8 hours later. Politicians in the West including Pfizer CEO have NOT Taken the vaccine.'}

*Input_Evidences* = {'No health care workers died after Alabama began administering COVID-19 vaccines on Tuesday',

'Some online posts falsely claimed that a nurse had died after receiving the vaccine',

'Alabama Department of Public Health officials checked with hospitals that administered the vaccine to confirm that the information was false',

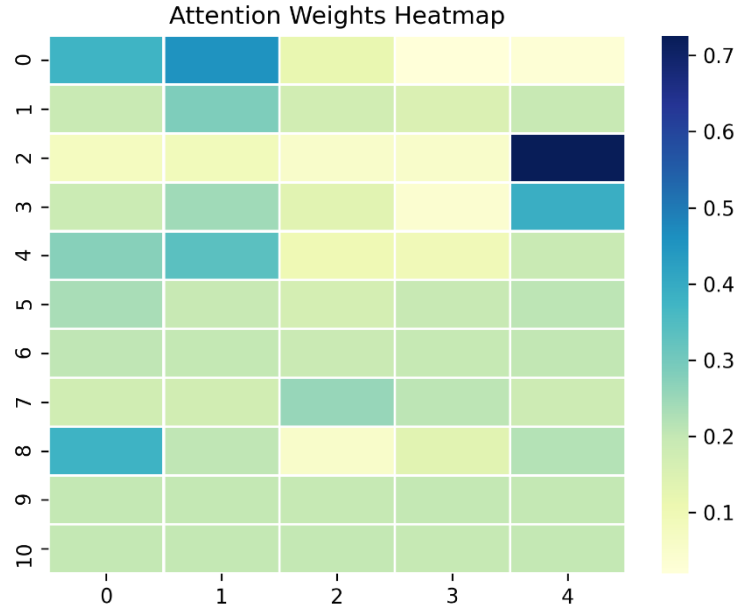'The department released a statement on social media to combat the misinformation'}

---

*Figure 10: Attention Weights Heatmap – Alabama Nurse Vaccination Death Claim*

The first 5 rows indicate the attention weights from nodes 1 to 5 in the 1st-layer of the ERNet and the second 5 rows are from the 2nd-layer of ERNet. The last row shows the weights from the attention aggregator. The attention heatmap shows a darker shade with a higher attention coefficient score.

From the heatmap, we can observe that the graph network evidence nodes attend specifically to the 1st and 2nd sentences {*'No health care workers died after Alabama began administering COVID-19 vaccines on Tuesday',* *'Some online posts falsely claimed that a nurse had died after receiving the vaccine'*}. This indicates that most of the higher-value information were retrieved from these 2 sentences while the other evidence provided a much lesser weightage. An interesting point to note was that additional attention could be extracted from the 2nd layer graph network as shown in the darker blue shading regions from nodes 7 and 8. From this observation, we can also understand that ERNet is able to extract useful information from multiple sentences repeatedly to ascertain and provide an eventual score classification.

## 2.3.2.5. Score Aggregation

In the final stage of the Fact Verification pipeline, we apply a majority voting aggregator based on the classification score outputs from all the different news article evidence and the input claim. The following pseudo-code logic flow outlines the voting aggregation approach:

*FOR LOOP i over n_articles*

      *IF article_i == "SUPPORTS"*

           *Majority_vote_count += 1*

*IF Majority_vote_count/Total_Num_Articles > 0.5*

      *THEN Final_Verdict = "SUPPORTS"*

*ELSE IF Majority_vote_count/Total_Num_Articles == 0.5*

      *THEN Final_Verdict = "NOT ENOUGH EVIDENCE"*

*ELSE Final_Verdict = "REFUTES"*

SureBoT then proceeds to return the final verdict scores, together with the summaries of the respective articles and their corresponding URLs that was obtained in stage 1 for the users' assessment on the input claim.

## 2.4. SYSTEM IMPLEMENTATION

The iBPU is fully developed on Python (3.7), utilising various Transformers' models from huggingface.co, PyGoogleNews and newspaper3k news extraction API(s) for web article scrapping. In addition, we utilise PyTorch (1.8.0) library for model training and fine-tuning. The iBPU is then implemented together with FLASK API to form the core of the backend processing framework for the intelligent chatbot.

The following table denotes the software libraries and models utilised:

| S/N | Software Lib/Models | Usage | Version/url |
|-----|--------------------|-------|-------------|
| 1. | Pegasus-cnn_dailymail | Abstractive Summarisation | https://huggingface.co/google/pegasus-cnn_dailymail |
| 2. | Msmarco-distilroberta-base-v2 | Sentence Semantic Similarity | https://huggingface.co/sentence-transformers/msmarco-distilbert-base-v2 |
| 3. | BERT-pair | Feature Embedding Extraction | Trained model |
| 4. | 2layerbest.pth.tar | GraphNet model | Trained model |
| 5. | Spacy | Tokeniser | 3.0.5 |
| 6. | Pygooglenews | Google News API | 0.1.2 |
| 7. | Newspaper3k | Article Extraction | 0.2.8 |
| 8. | Transformers | Build transformer models | 4.4.2 |
| 9. | Torch | Machine Learning Libraries | 1.8.0 |
| 10. | Pytorch_pretrained_bert | Pretrained models | 0.5.1 |
| 11. | Sentence_transformers | Feature extraction | 1.0.2 |
| 12. | Scipy | Misc Computation and Plotting | 1.6.1 |
| 13. | Numpy | Misc Computation and Plotting | 1.20.1 |
| 14. | Matplotlib | Misc Computation and Plotting | 3.3.4 |
| 15. | Seaborn | Misc Computation and Plotting | 0.11.1 |
| 16. | Pandas | Misc Computation and Plotting | 1.2.3 |
| 17. | Scikit-learn | Misc Computation and Plotting | 0.24.1 |
| 18. | requests | Misc Computation and Plotting | 2.25.1 |
| 19. | Validators | Misc Computation and Plotting | 0.18.2 |

*Table 3 Software libraries and model lists*
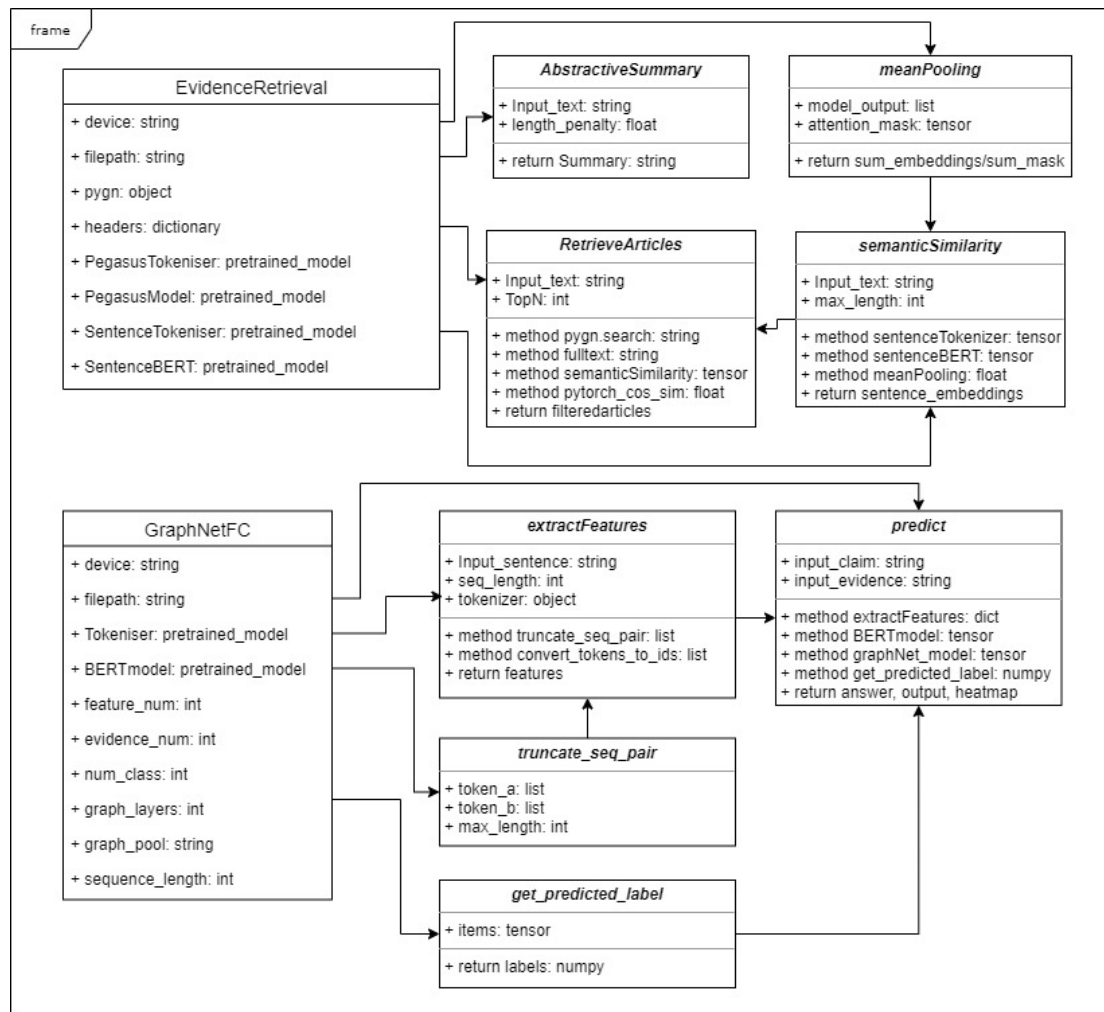
## 2.4.1. CLASS DIAGRAM

*Figure 11: SureBoT Class Diagram*

## 2.4.2. SYSTEM USE-CASE SEQUENCE

The pipeline workflow follows a sequential input stream from user input to bot response. We built in a timeout window of 360 seconds to prevent unintended pipeline freeze due to failures in extraction or processing. The software sequence diagram is defined as follow:
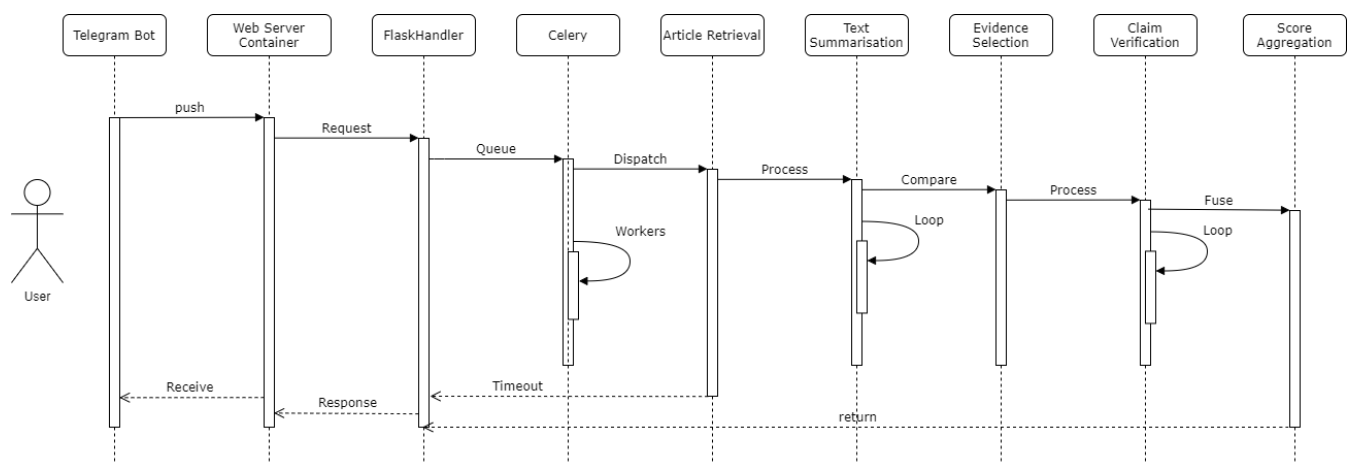


*Figure 12: Pipeline Use-Case Sequence Diagram*

## 2.5.   ASSUMPTIONS

In the process of designing and building the system architecture, the following assumptions were made by the project team:

- All returned articles from the Google News site are from validated and legitimate sources. This is important as the requirement for the veracity of evidence collection must be upkept and adhered to.

- Bot server platform has access to internet for data mining to build knowledge base. Offline processing is not supported unless query claim results have been cached locally. Otherwise, the system relies on real-time updated news sources to maintain information validity.

## 2.6.   SYSTEM PERFORMANCE

To evaluate the overall performance of the system and enhance the accuracy and efficiency of the model, we will assess the system's metrics based on the following 2 parameters:

a)   Time savings

b)   Relevancy Accuracy Score

For Time savings wise, we compare the time taken to do the fact-checking with SureBoT as compared to the manual fact checking, i.e., manually searching online for relevant articles / news that support or refute the query. The comparison is further broken into 2 sub-tasks: "easy" and "difficult" queries. For "easy" queries, this would refer to query where there are significant number of articles online that are able to provide sufficient evidence to support or refute the claim. On the other hand, "difficult" query will refer to query that have little or close to no article online to substantiate the claim.

Testing for these queries is conducted by our team of 4 and the aggregated scores are consolidated in the table below.

| | "Easy" Query | "Difficult" Query |
|---|---|---|
| Manual Fact-Check | 8 min | 25 min |
| SureBoT | 2 min 30 s | 5 min |
| Time saved (%) | 68.75 % | 80.00 % |

*Table 4: Time Savings Performance Table*

Based on the results gathered, SureBoT has indeed significantly saved a lot of time in the fact-checking process. Moreover, the system processes the fact-checking task in the background, freeing up precious time and resources for the user to attend to other things while the system is processing. User will be notified of the results when the system finished processing.

In addition, we have also created a short survey and invited 20 attendees to provide their inputs on SureBoT. The test involves them testing the SureBoT with 3 queries each, and input a relevancy accuracy score based on the results produced by the SureBoT. The survey link and results are appended in Appendix E.

The relevancy accuracy score ranges from 1 (least relevant) to 10 (most relevant). The score is based on the accuracy of the results returned, and how relevant the articles in the results are to the query. From the survey, the results returned an approximate score of **6.17 out of 10**. This was partially due to users inputting queries that revolves around general knowledge, e.g., is Portsmouth in Singapore? Such queries do not return any articles from Google News, and thus is unable to fact-check the claim. Also, this was due to some limitations of the system which is further elaborated in Section 2.7.

## 2.7. LIMITATIONS AND IMPROVEMENTS

While the system has in place certain constraints and checks to enable robust and reliable performance, we note that these limitations may present as opportunities to develop the system further and improve the overall usability.

| S/N | LIMITATIONS | IMPROVEMENTS |
|---|---|---|
| 1. | No multi-lingual support.<br><br>The current pipeline only processes text information based on the English language. This is due to the lack of readily available multi-lingual fact-checking data sources. | To source or curate custom data set for multi-lingual language support such as Chinese, Tamil and Bahasa Melayu. |
| 2. | Limited contextual understanding for colloquial Singlish:<br><br>As certain text article claims may come in the form of Singlish the affects the model's ability in forming reasoning due to the lack of domain understanding for colloquial phrasing | To source or curate custom data set for colloquial language support and text understanding for commonly used slangs in Singlish. |
| 3. | The current fact-checking framework purely based on Natural Language Processing and is unable to handle multimodal data sources. While we understand that predominantly, news articles are presented in textual forms. There is increasingly more occurrence of false information propagated through images such as Memes, and video clips. | Strengthen the pipeline with the use of multimodal and video content analysis. This could be realised with the use of a combination of Computer Vision techniques and Visual-Linguistic (VL) processing. |
| 4. | The current bot platform is hosted on Google Compute Engine VM running on a pure CPU-based computational environment. The system is created under the free account credits package, which provides limited selection of computation instances for quick prototyping.<br><br>The pipeline, however, is structured and programmed to exploit GPU processing using CUDA libraries to speed up model loading and inference. Unfortunately, this speed up is not currently being realised.<br><br>The current framework also has a limit capacity on scalability due to the number threads allowable and instances that could be spun up. | To source for funding through value-investors to support platform deployment and development. Google Compute Engine currently offers GPU-enabled processing with V100s and automatic instance load-balancing and scalability. |
| 5. | The web scrapping functions are limited to search results within Google News platform. While the platform provides multiple news sources from various media outlets globally, it does include verified fact-checking media sources such as fact-check.org or gove.sg/factually.<br><br>An alternative method would be to utilise a HTML parsing framework via Google search results. However, there exists multiple engineering challenges due to restrictions set in place to prevent automated web crawling and bot activities on Google. | More development work would need to be done on the information retrieval block to ensure a robustness in article selection and evidence gathering. |
| 6. | Deployment on messaging platforms is currently limited to Telegram due to open-source libraries support. While it is possible to create bots in WhatsApp, there are still strict requirements to adhere to and the API framework is not openly available. Therefore, the target user reach may be limited due to lack of cross-platform support. | To develop identical platform architecture that would be migratable and implementable in other popular messaging platforms such as WhatsApp, WeChat and Signal. |
| 7. | BERT-pair sentence encoder and GraphNet is trained on FEVER shared task dataset based on Wikipedia. While the knowledge representation do share some | We require to build a target domain evidence-claim dataset based on Singapore's news article offerings. |

| | | |
|---|---|---|
| | commonalities, in order to further improve accuracy, it would be ideal to fine-tune and train on the target domain dataset. | |
| 8. | The current summary abstraction model can only take in input size of up to 1024 token lengths. This is due to the pretrained model structure and limitations. Unfortunately for long articles, we would require applying a truncation operation to ensure correct input to the model. | The summariser model can be custom trained to obtain a longer article structure, however, the overall accuracy may decrease due to increasing context complexity.<br><br>Another alternative method would be to spilt long articles into 1024 chunks for summarisation before downstream fusion to ensure all information is captured. However this would increase complexity of the architecture due to the fusion mechanism. |

*Table 5: Table of limitations & improvements*

## 3. CONCLUSION

We have developed an automated end-to-end fact checking chatbot deployed on the Telegram messaging platform. The interactive bot is capable of extracting relevant news articles from the internet, performing contextual reasoning and finally, providing an inference output with the article summaries to aid in the decision-making process. The system is easy to use, readily available, and provides the ability to scale up to support deployment in production.

The project team has demonstrated the knowledge understanding in designing, building, and deploying an Intelligent Reasoning System. We identified a pertinent social problem regarding the flagrant spread of misinformation and conducted extensive studies and market survey to understand the underlying reasons. Finally, we proposed a systematic framework based on a cognitive model framework to tackle the issue through the use of technology.

Through the course of research and development, the team surmounted many technical challenges to prototype a baseline system with a reasonable performance. The experience gained, knowledge uncovered, and friendships forged were the most valuable take-aways from the entire journey. Overall, the team developed confidence and a deeper understanding in building and deploying practical Artificial Intelligence systems.

### 3.1. RECOMMENDATIONS AND FUTURE WORK

From Table 5, we outline the current limitations and possible improvements that can be further developed in the project. As of priority, the current immediate recommendation for development would be to enable production scaling upwards of the platform to handle higher bot request loads. Also, with sufficient financial funding, we would aspire to utilise GPU computation to speed up the pipeline query-results turn-around time. As with all software usability guideline, a faster and more efficient turn-around time will always benefit user experience and encourage higher traffic volume. With the increased awareness and usage of the chatbot, there can be further improvements through constructive feedback and suggestions.

We envisage the iBPU developmental framework to be open source to encourage collaboration and facilitate knowledge sharing. It is through this spirit of candour that we believe would rapidly push the technology advancement and inspire like-minded teams or individuals to come onboard to build and develop software systems for the benefit of society. Combating the spread of misinformation requires the responsibility of every citizen in the society, and therefore, it is imperative that firms and individuals with the authority and ability to make a difference to take a stand and lead the future.

## 3.2. REFERENCES

- "6 in 10 people in Singapore have received fake COVID-19 news, likely on Social Media: Survey." https://www.channelnewsasia.com/news/singapore/fake-covid-19-news-study-ncid-messaging-platforms-whatsapp-12756084, 21 May 2020, Channel News Asia

- "4 in 5 Singaporeans confident in spotting fake news but 90 per cent wrong when put to the test: Survey." https://www.straitstimes.com/singapore/4-in-5-singaporeans-confident-in-spotting-fake-news-but-90-per-cent-wrong-when-put-to-the, 27 Sep 2018, The Straits Times

- "Online falsehoods are a global problem. They create discord and division, provoke violence, and undermine confidence in public institutions." https://www.facebook.com/gov.sg/posts/online-falsehoods-are-a-global-problem-they-create-discord-and-division-provoke-/10156003127008686/, 5 Jan 2018, gov.sg on Facebook

- James Thorne, Andreas Vlachos, Oana Cocarascu, Christos Christodoulopoulos, and Arpit Mittal. 2018b. The Fact Extraction and VERification (FEVER) shared task. In Proceedings of the First Workshop on FEVER, pages 1–9.

- James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018a. Fever: a large-scale dataset for fact extraction and verification. In Proceedings of NAACL-HLT, pages 809–819.

- Zhang, J., Zhao, Y., Saleh, M., and Liu, P. J. 2019b. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. arXiv preprint arXiv:1912.08777.

- Hermann, K. M., Kocisky, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., and Blunsom, P. 2015. Teaching machines to read and comprehend. In Advances in neural information processing systems, pp. 1693–1701.

- Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. arXiv preprint arXiv:1908.10084.

- Jie Zhou, Xu Han, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2019a. Gear: Graph-based evidence aggregating and reasoning for fact verification. arXiv preprint arXiv:1908.01843.

- Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018. Graph attention networks. In Proceedings of ICLR.

- Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In Proceedings of ICLR.

- Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. arXiv preprint arXiv:1910.01108.

- Andreas Hanselowski, Hao Zhang, Zile Li, Daniil Sorokin, Benjamin Schiller, Claudia Schulz, and Iryna Gurevych. 2018. Ukp-athene: Multi-sentence textual entailment for claim verification. arXiv preprint arXiv:1809.01479.

- Yixin Nie, Haonan Chen, and Mohit Bansal. 2019. Combining fact extraction and verification with neural semantic matching networks. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 33, pages 6859– 6866.

- Ray Oshikawa, Jing Qian, and William Yang Wang. 2018. A survey on natural language processing for fake news detection. arXiv preprint arXiv:1811.00770

- Qifei Li, Wangchunshu Zhou. 2020. Connecting the dots between Fact Verification and Fake News Detection. arXiv:2010.05202.

- Amir Soleimani, Christof Monz, Marcel Worring. 2019. BERT for Evidence Retrieval and Claim

Verification. arXiv:1910.02655

- Jackson Luken, Nanjiang Jiang, and Marie-Catherine de Marneffe. 2018. Qed: A fact verification system for the fever shared task. In Proceedings of the First Workshop on FEVER, pages 156–160.

- Andreas Hanselowski, Hao Zhang, Zile Li, Daniil Sorokin, Benjamin Schiller, Claudia Schulz, and Iryna Gurevych. 2018. Ukp-athene: Multi-sentence textual entailment for claim verification. Proceedings of EMNLP, pages 103–108.

# 4. APPENDIX

## APPENDIX A: Project Proposal

**GRADUATE CERTIFICATE: Intelligent Reasoning Systems (IRS)**

**PRACTICE MODULE: Project Proposal**

| |
|---|
| **Date of proposal:**<br><br>20 February 2021 |
| **Project Title:**<br><br>ISS Project – SureBoT (An intelligent End-To-End Fact checking Telegram BOT) |
| **Sponsor/Client:** *(Name, Address, Telephone No. and Contact Name)*<br><br>Institute of Systems Science (ISS) at 25 Heng Mui Keng Terrace, Singapore<br>NATIONAL UNIVERSITY OF SINGAPORE (NUS)<br>Contact: Mr. GU ZHAN / Lecturer & Consultant<br>Telephone No.: 65-6516 8021<br>Email: zhan.gu@nus.edu.sg |
| **Background/Aims/Objectives:**<br><br>A large part of messages we receive today on social media like WhatsApp, Telegram, Facebook and twitter are partially true or misinformation.<br>Misinformation is defined as false or inaccurate information. Most of the Singaporeans are exposed to misinformation.<br><br>The task of determining the validity or authenticity of information received in social media are menial and time consuming. Wide spread circulation of Misinformation and fake news in social media has economic, social and political consequences. Misinformation causes unintentional fear and anxiety. It also undermines our trust in the public institutions, causing public health scares, provoking violence and causing discord and divisions withing the society.<br><br>It is our social responsibility to stop the spread of fake news and misinformation.<br>To combat the evils of misinformation we are proposing SureBot; an automated fact checking telegram chatbot. It connects people to trusted sources of information.<br><br>The proposed intelligent chatbot to provide and end-to-end automated fact-checking tool. The chatbot to be based on a cognitive framework that accepts user input, applies feature extraction and knowledge reasoning techniques to produce a classification output that decides whether the supporting evidence "SUPPORTS" or "REFUTES" the input claim. |
| **Requirements Overview:**<br><br>• SureBot makes use of an intelligent cognitive framework to help understand and validate textual information.<br>• Perform automated web queries for evidence selection.<br>• Make use of contextual cross referencing to provide unbiased classification algorithm based on user input query. |

- SureBot needs to be fast, efficient and simple to use and it should work 24/7
- A minimalist approach to input a claim to be fact checked and let the chatbot to take care of everything else to ensure that this tool can be used by elderly and less tech savvy.
- The proposed performance measurement metrics to evaluate the overall performance of the SureBot are Time-taken for fact-checking and validation process and Relevancy accuracy score based on the number of articles found. We aim to achieve significant improvements in these metrics compared to human level fact checking.
- Baseline the system's performance based on real-world queries from multiple users to obtain a relevancy accuracy score matrix. The target score output would be a number from 0 to 10, with 0 being the least relevant and 10 being the most relevant.
- Scores to be collected and aggregated across different participants with different backgrounds and age-groups, performing real-world queries on current affairs.
- Use off-the-shelf pre-trained models in Abstractive Text Summarization, Text Semantic Similarity comparison and Fact Verification as the fundamental building blocks for the end-to-end Fact-Checking pipeline.
- News article retrieval based on real-time news article web scraper based on pyGoogleNews and newspaper3K API.
- Text Summarization utilizes large Transformer-based encoder-decoder framework PEGASUS.
- Evidence selection utilizes the state-of-the-art Sentence-BERT Siamese network.
- Claim verification utilizes a novel fact verification model based on a fine-tuned BERT-pair encoder trained on the FEVER shared task dataset and a graph neural network model.
- The heart of the Claim verification stage is the Evidence-Reasoning Network (ERNet) which is a fully connected attention network graph with input evidence sentences and claim as input nodes.

**Resource Requirements (please list Hardware, Software and any other resources)**

Hardware proposed for consideration:
- To setup the backend on Google compute engine VM which provides cloud-based storage for the processing models required and the computation resource to perform fact-checking interface.

Software proposed for consideration:
- Open-source web server container NGINX
- GUNICORN Web Server Gateway Interface (WSGI)
- REDIS message broker.
- Flask a python web framework
- Celery for message synchronization between the messages and processes
- Telegram Bot API as Front-end service.

**Number of Learner Interns required: (Please specify their tasks if possible)**

a team of four to six project members (or individual work upon lecturer approval)

**Methods and Standards:**

| Procedures | Objective | Key Activities |
|---|---|---|
| Requirement | The team should meet with ISS to | 1.    Gather & Analyze Requirements |

| Gathering and Analysis | scope the details of project and ensure the achievement of business objectives. | 2. Define internal and External Design<br><br>3. Prioritize & Consolidate Requirements<br><br>4. Establish Functional Baseline |
|---|---|---|
| Technical Construction | · To develop the source code in accordance to the design.<br><br>· To perform unit testing to ensure the quality before the components are integrated as a whole project | 1. Setup Development Environment<br><br>2. Understand the System Context, Design<br>3. Perform Coding<br>4. Conduct Unit Testing |
| Integration Testing and acceptance testing | To ensure interface compatibility and confirm that the integrated system hardware and system software meets requirements and is ready for acceptance testing. | 1. Prepare System Test Specifications<br><br>2. Prepare for Test Execution<br><br>3. Conduct System Integration Testing<br><br>4. Evaluate Testing<br>5. Establish Product Baseline |
| Acceptance Testing | To obtain ISS user acceptance that the system meets the requirements. | 1. Plan for Acceptance Testing<br>2. Conduct Training for Acceptance Testing<br><br>3. Prepare for Acceptance Test Execution<br><br>4. ISS Evaluate Testing<br><br>5. Obtain Customer Acceptance Sign-off |
| Delivery | To deploy the system into production (ISS standalone server) environment. | 1. Software must be packed by following ISS's standard<br><br>2. Deployment guideline must be provided in ISS production (ISS standalone server) format<br><br>3. Production (ISS standalone server) support and troubleshooting process must be defined. |

**Team Formation & Registration**

| |
|---|
| Team Name: IRS-PM-16-01-2021-ISY5001PT-SureBoT |
| Project Title (repeated):  SureBoT – An Intelligent End-To-End Fact Checking Telegram BOT |
| System Name (if decided): SureBoT |
| |
| Team Member 1 Name:    KUEK YONG JIE ADRIEL |
| Team Member 1 Matriculation Number:  A0229985H |
| Team Member 1 Contact (Mobile/Email): 91888750 / e0687393@u.nus.edu.sg |
| |
| Team Member 2 Name: M. LAVANYA |
| Team Member 2 Matriculation Number: A0229979B |
| Team Member 2 Contact (Mobile/Email): 91705264 / e0687387@u.nus.edu |
| |
| Team Member 3 Name:  CHUA HAO ZI |
| Team Member 3 Matriculation Number: A0229960W |
| Team Member 3 Contact (Mobile/Email): 83829287/e0687368@u.nus.edu |
| |
| Team Member 4 Name: Chammanikodathu Louis Francis |
| Team Member 4 Matriculation Number: A0079985B |
| Team Member 4 Contact (Mobile/Email): 90613560 / e0689729@u.nus.edu |
| |

| For ISS Use Only | | |
|---|---|---|
| **Programme Name:** | **Project No:** | **Learner Batch:** |
| **Accepted/Rejected/KIV:** | | |
| **Learners Assigned:** | | |
| **Advisor Assigned:**<br><br>Contact: Mr. GU ZHAN / Lecturer & Consultant<br>Telephone No.: 65-6516 8021<br>Email: zhan.gu@nus.edu.sg | | |

## APPENDIX B: Installation & User Guide

## 4.1.  Current SureBoT Baseline Deployment on Telegram

The bot server is temporarily hosted on a GCE Instance and can be tested with the Telegram Messenger. Type SureBoT in the search tab and you should see a Bot with the username '@fact_check_bot'. Choose the Bot and you should be seeing a message as shown below. Click start and start fact checking!
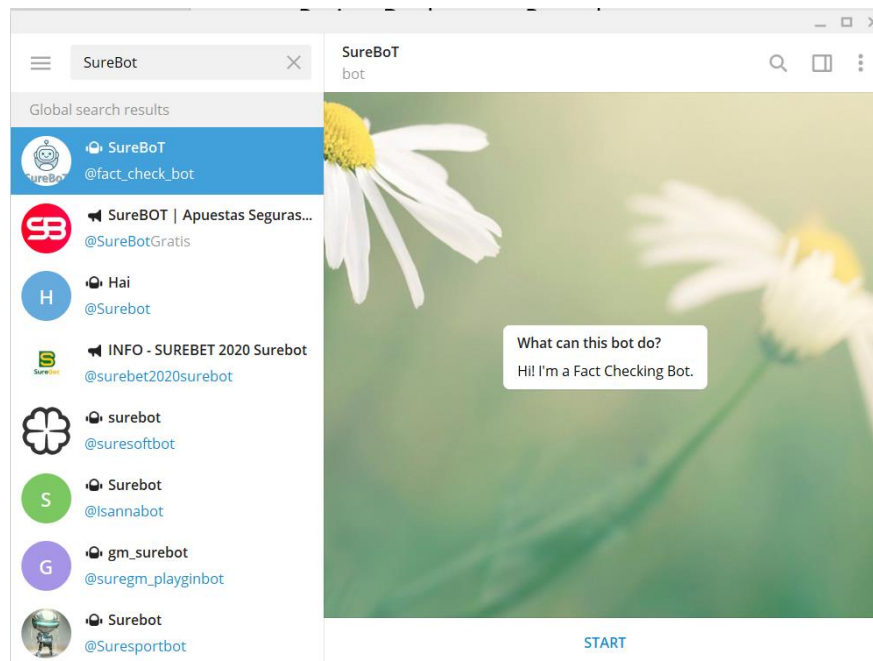


## 4.2.  Test Fact Checking Pipelining

If you want to locally test the fact-checking pipeline, without using the Bot please follow the steps indicated below:

1.  Clone the Github repo and install the requirements by running below command in terminal.
    ```
    pip install -r requirements.txt
    ```

2.  Open DownloadModels.py and replace "<MODEL-DOWNLOAD-URL>" with the link mentioned in the README.md file. Run DownloadModels.py. This file will download and unzip the folder containing the pretrained models in the current directory. If the download is unsuccessful, manually download the zip file from below link and unzip it in the directory of the repo:

3.  Link: https://drive.google.com/file/d/1jxUd27-K51AkRX20yEHtban8Uvab7lkX/view?usp=sharing

4.  Run SureBoT_main.py and you will see a command line interactive mode in which you can input your queries.

## 4.3.  Test Bot Server locally

If you want to create your own telegram bot and run the bot server locally, please follow the below steps:

### Create a Telegram Chat Bot:

1. Download the desktop version of Telegram from below URL and login to telegram.

(https://desktop.telegram.org/)

2. Enter bot father in the search tab and choose the Bot

3. Click 'Start' or 'Restart' if you have already communicated with BotFather before.
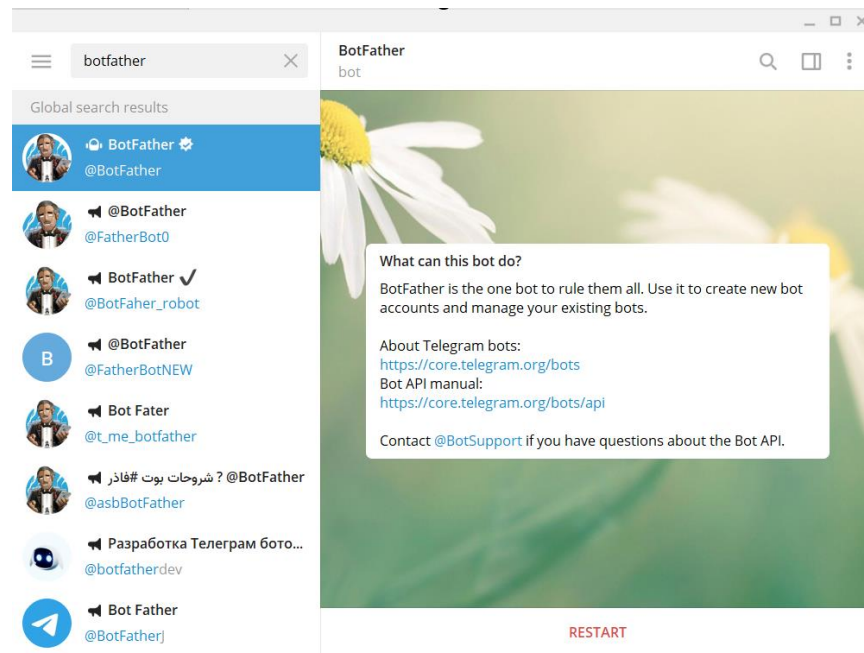


4. Choose or type the **/newbot** command and send it.

5. Choose a name for your bot. The username must be unique and end with the word "bot."

6. After you choose a suitable name for your bot — the bot is created. You will receive a message with a link to your **bot t.me/<bot_username>**

7. Copy the token from the above message. You will be using it later in the configuration file.
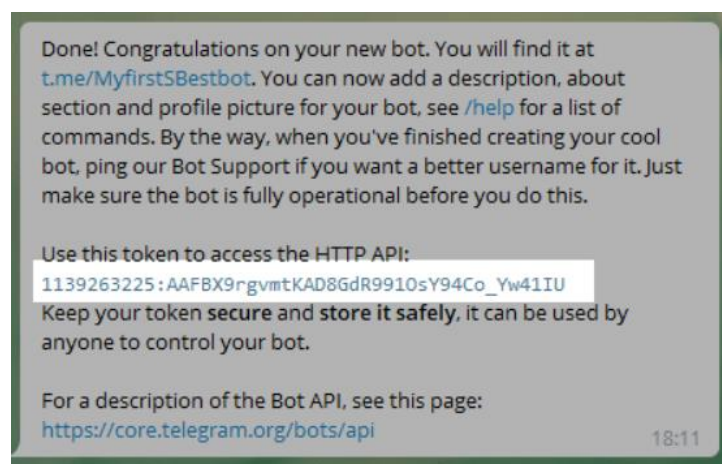


## Running the bot server locally in Ubuntu:

1. Setup your environment by entering below commands in terminal.

```
# update system packages and install the required packages
sudo apt-get update
sudo apt-get install bzip2 libxml2-dev libsm6 libxrender1 libfontconfig1

# clone the project repo
git clone <Enter the GITHUB Repo URL>
```

2. Install conda if you don't have it already to create a virtual environment.

```
# download and install miniconda
wget https://repo.anaconda.com/miniconda/Miniconda3-4.7.10-Linux-x86_64.sh

bash Miniconda3-4.7.10-Linux-x86_64.sh
```

Follow the instructions and agree to the terms to install Miniconda

```
export PATH=/home/<your name here>/miniconda3/bin:$PATH
rm Miniconda3-4.7.10-Linux-x86_64.sh

# confirm installation
which conda

# create and activate a new environment
conda create -n telegram-bot python=3.8
conda activate telegram-bot
```

3. Install all the requirements as shown below:

```
# go to project root and install the requirements
cd SureBoT/Code
pip install -r requirements.txt
```

4. Setting up NGINX

   a) Install and start NGINX using below commands

```
cd
sudo apt-get install nginx-full
sudo /etc/init.d/nginx start
```

   b) Remove default configuration file and create a new site configuration file for our application.

```
# remove default configuration file
sudo rm /etc/nginx/sites-enabled/default

# create a new site configuration file
sudo touch /etc/nginx/sites-available/telegram-bot

sudo ln -s /etc/nginx/sites-available/telegram-bot
/etc/nginx/sites-enabled/telegram-bot
```

   c) Edit configuration file for our app by opening file with an editor

```
sudo nano /etc/nginx/sites-enabled/flask_project
```

   d) Copy and paste below code and save configuration file

```
server {
    listen 80;
    server_name localhost;
    location / {
        proxy_pass http://0.0.0.0:5000;
    }
```

```
}
```

e) Restart NGINX Server

```
sudo /etc/init.d/nginx restart
```

**Note**: If you see an error on restart, please try to change the port number in step (d) and retry. Also take note of the port number as you would need to use the same when starting ngrok locally.

5. Install Redis server

```
sudo apt update
sudo apt install redis-server
sudo systemctl status redis-server
```

```
(telegram-bot) irsprj_factcheck@fact-check-bot-instance:~/SureBo_T$ sudo systemctl status redis-server
● redis-server.service - Advanced key-value store
     Loaded: loaded (/lib/systemd/system/redis-server.service; enabled; vendor preset: enabled)
     Active: active (running) since Mon 2021-04-12 13:12:43 UTC; 5min ago
       Docs: http://redis.io/documentation,
             man:redis-server(1)
    Process: 1231 ExecStart=/usr/bin/redis-server /etc/redis/redis.conf (code=exited, status=0/SUCCESS)
   Main PID: 1241 (redis-server)
      Tasks: 4 (limit: 19206)
     Memory: 2.8M
     CGroup: /system.slice/redis-server.service
             └─1241 /usr/bin/redis-server 127.0.0.1:6379

Apr 12 13:12:43 fact-check-bot-instance systemd[1]: Starting Advanced key-value store...
Apr 12 13:12:43 fact-check-bot-instance systemd[1]: redis-server.service: Can't open PID file /run/redis/redis-ser
Apr 12 13:12:43 fact-check-bot-instance systemd[1]: Started Advanced key-value store.
```

6. Replacing your bot token and the download URL for the Pretrained models used in the projects

a) Change directory to project root and open the config file

```
cd SureBoT/Code
nano bot_config.py
```

b) Replace "<YOUR-BOT-TOKEN>" with the token generated when creating the chatbot

c) Replace the "<MODEL-DOWNLOAD-URL>" with the URL provided in the ReadMe file

7. Installing NGROK

```
sudo snap install ngrok
```

8. Running the bot server → Run the shell script – **./StartSureBot.sh** in terminal and it will do the following:

a) Run Gunicorn and Celery daemon processes using supervisord

b) Start ngrok at the specified port number

c) Replace the tunnel url generated by ngrok as the Telegram bot's webhook

**Note**: If you had changed the port number in step 4 (d)/(e) please replace the port number in the file start_backend.py

You will see below message when you enter the tunnel URL in browser.
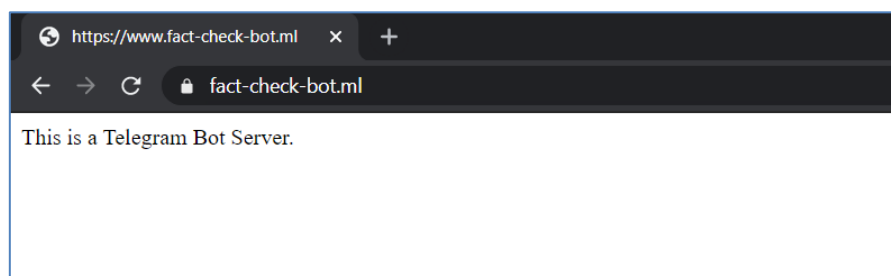
9. Stopping the bot server → Run the shell script – **./KillSureBot.sh** in terminal.

10. The logs generated by the Bot can be checked by inspecting the files below:

   a) gunicorn_log.log

   b) celery_log.log

```
cd SureBoT/Code
tail -f celery_log.log
tail -f gunicorn_log.log
```

## 4.4. Hosting the Bot server in GCE (Google Compute Engine) VM Instance
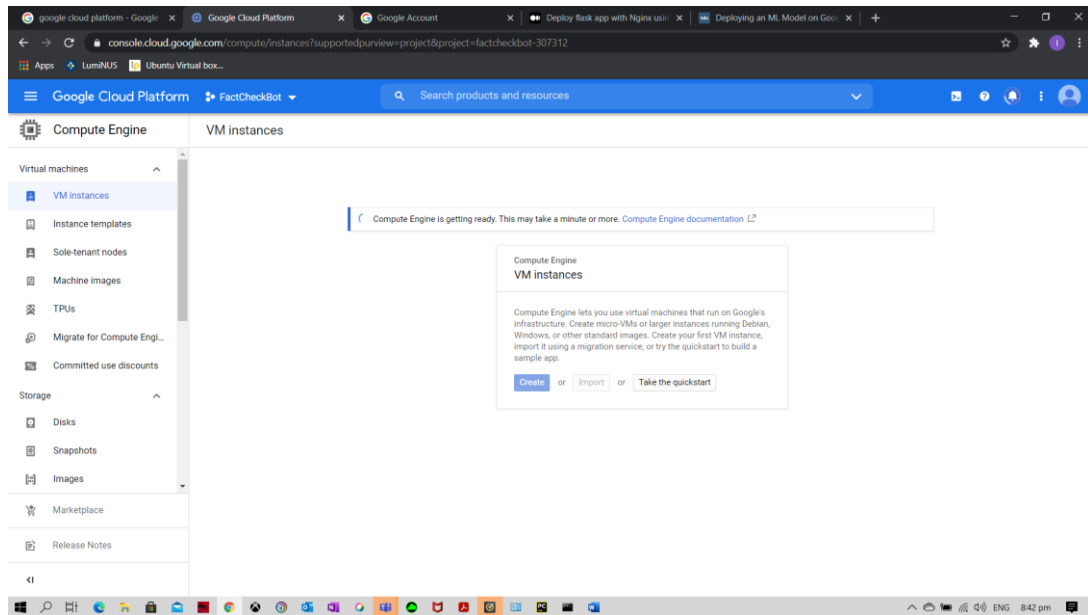
**Creating a Telegram Chat Bot:**

Please refer to the section on 'Create a Telegram Chat Bot' in the guide to 'Test Bot Server locally'.

**Creating a GCE Instance & domain name for Telegram Webhook**

1. Login to Google Cloud Platform

2. Create a new Project. You can create a new one here.
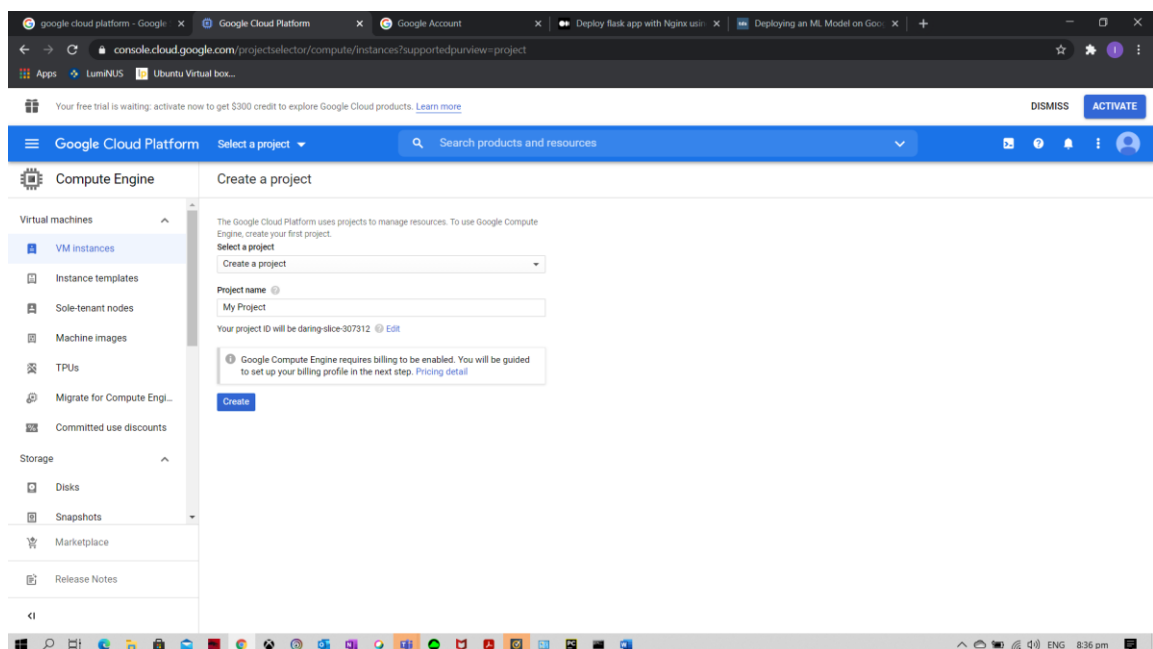


3. Create a new GCE Instance by clicking on Compute Engine > VM instance in the left panel

4.  Configure the VM Instance

    a)  Provide a relevant name for your instance.

    b)  Select Region. [asia-southeast1(Singapore) for Singapore]

    c)  Under Machine Configuration, select Series as "N2" and Machine Type as "n2-standard-4 (4 vCPU, 16 GB Memory)"

    d)  Change boot disk to Ubuntu 20.04 LTS

    e)  Set Firewall settings to allow HTTP and HTTPS traffic

    f)  Click Create.
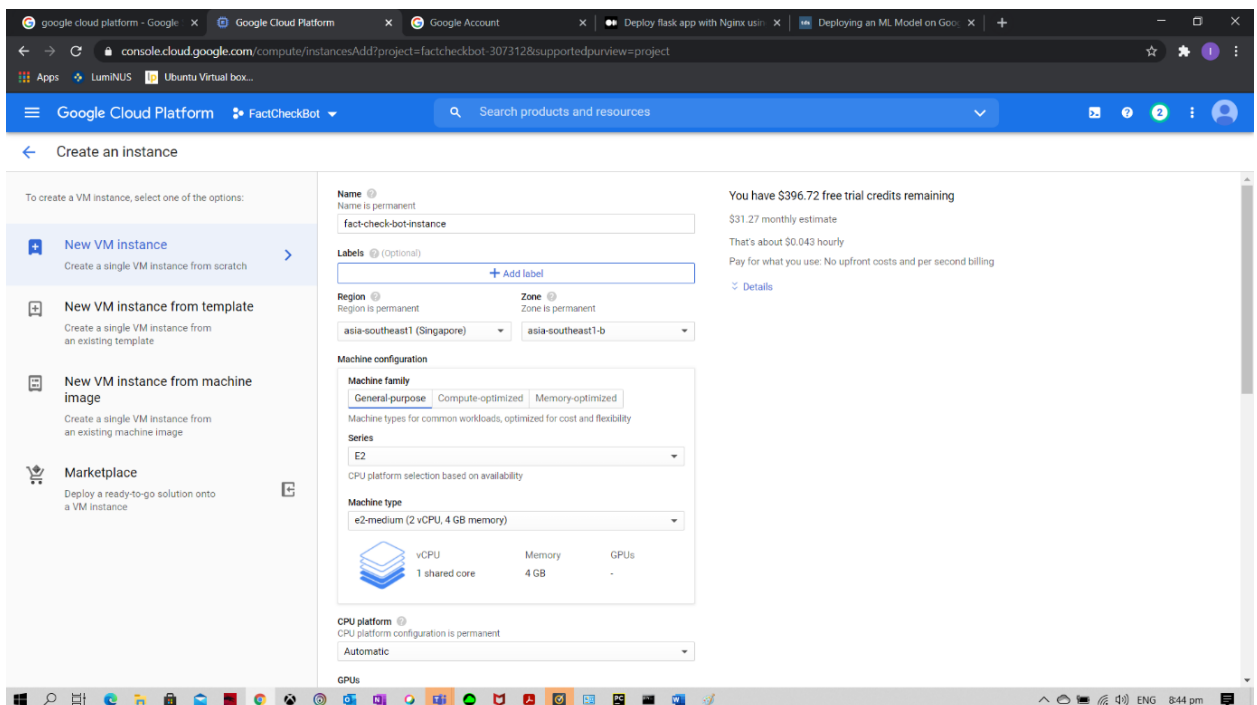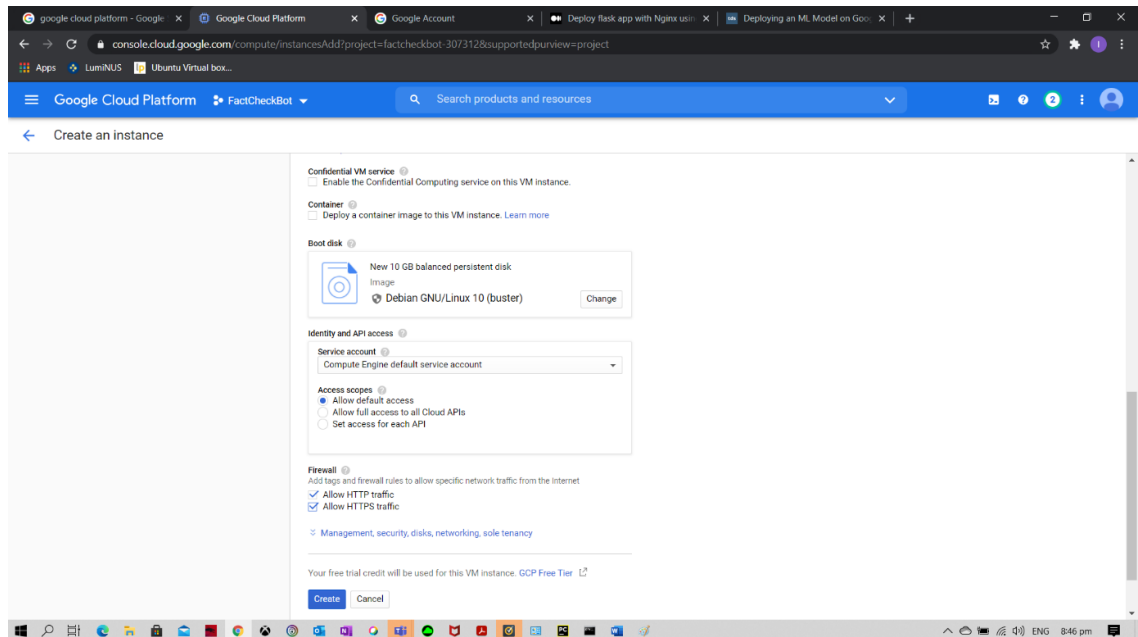
5. Firewall Settings

    a) Navigate to VPC network > Firewall rules from left panel
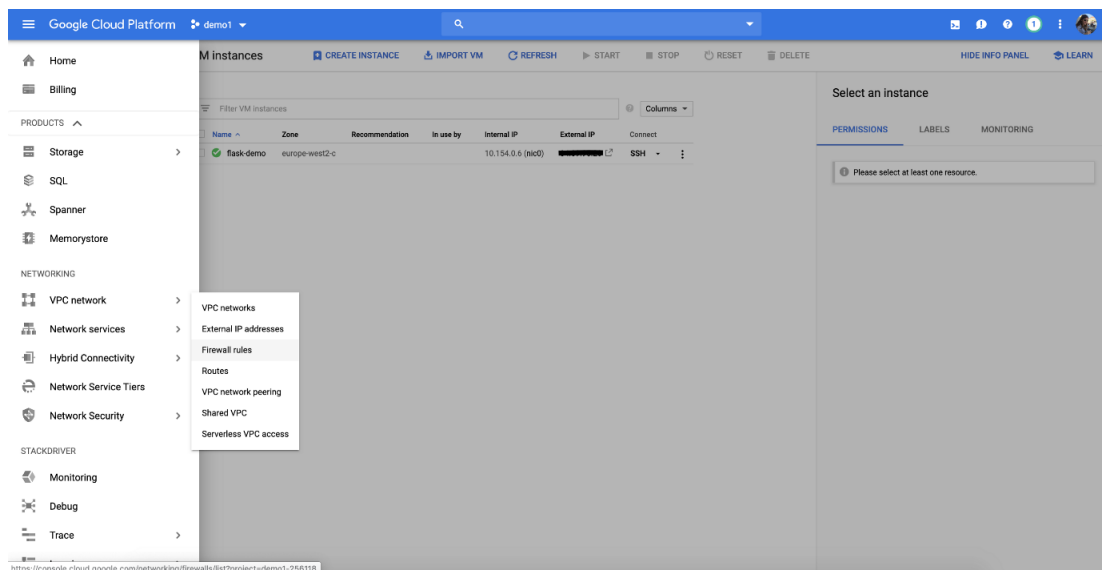


    b) Create firewall rule and configure settings

    Note: Make sure to add http-server and https-server to target tags, *0.0.0.0/0* to source IP ranges and set *5000* as the specified port.

6. Reserve Static IP Address for your VM Instance here

   a) Enter a Name and Description for the new address.

   b) Choose Version as IPV4 version

   c) Choose Type as Regional

   d) Select the Region as "asia-southeast1 (Singapore)"

   e) Attach this IP Address to your VM Instance. It can be selected from the drop down.

   f) Click Reserve to reserve the IP.
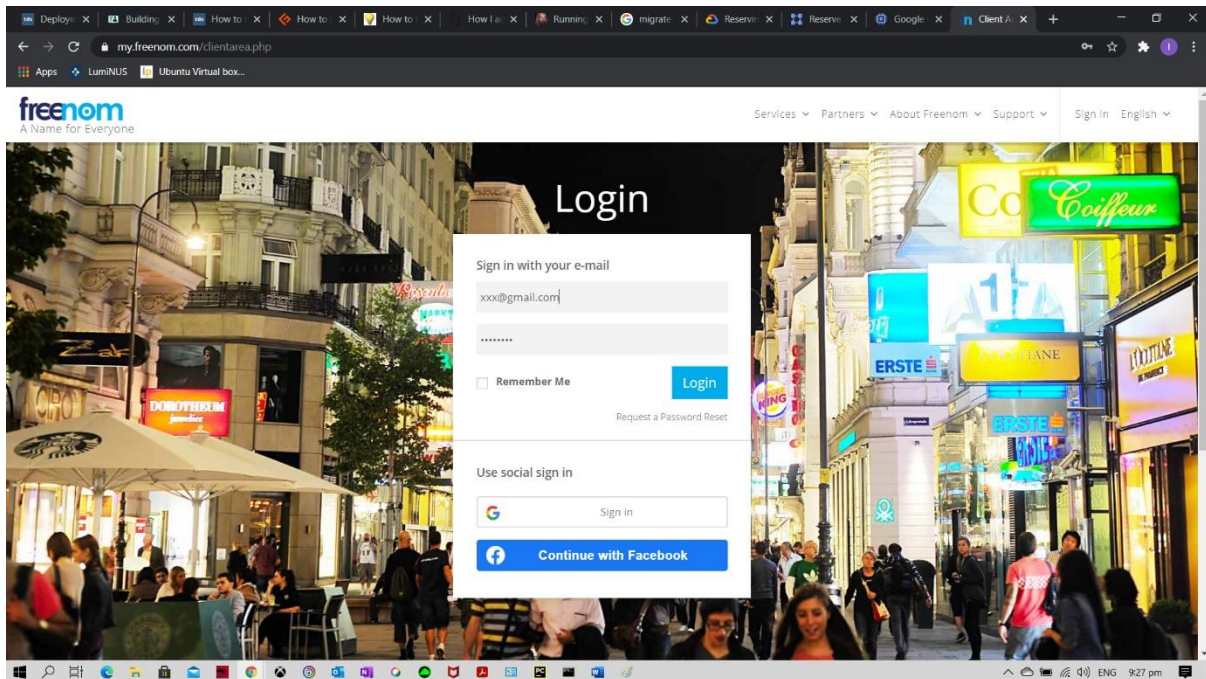
   g) Your static IP will be reflected in your VM

7. Create a domain name using the Static IP Address.

   a) There are many websites that provide free domain name.

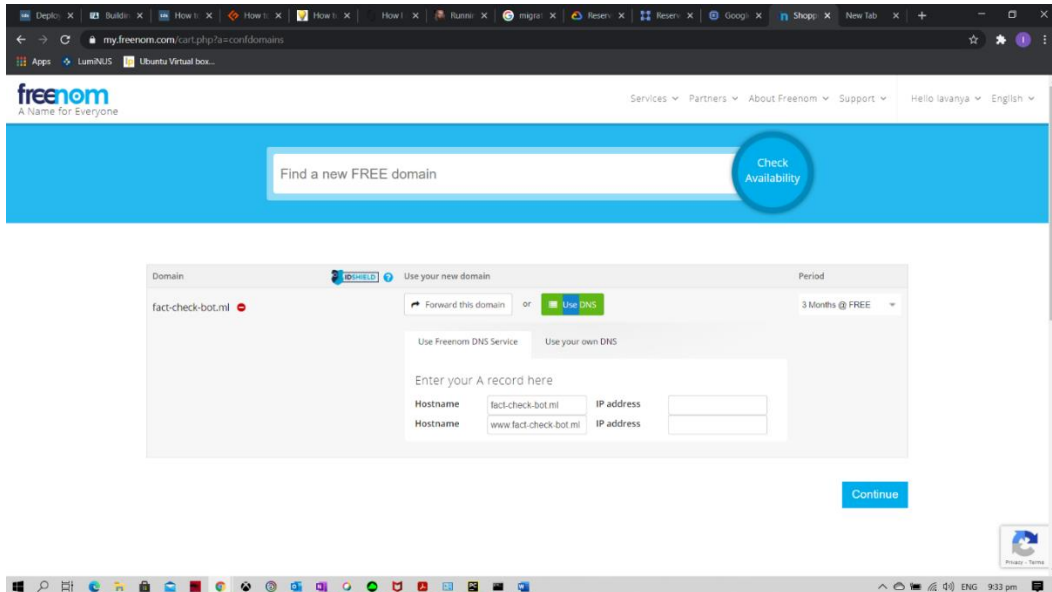      One of them is → https://www.freenom.com/en/freeandpaiddomains.html
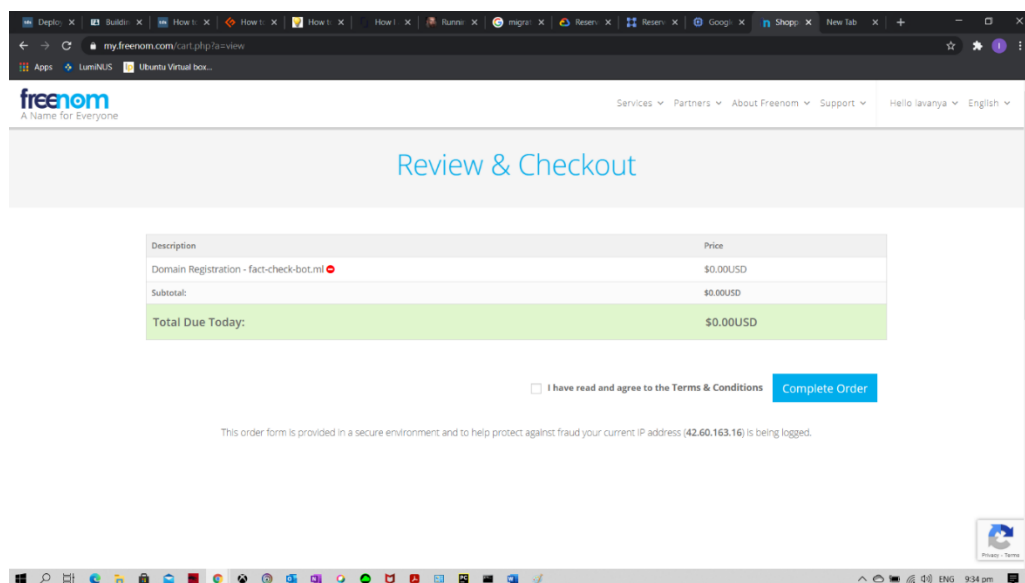
   b) Create an account and login to the above site.

c) Click on **Services** -> Register a New Domain

d) Enter a domain name and check availability.
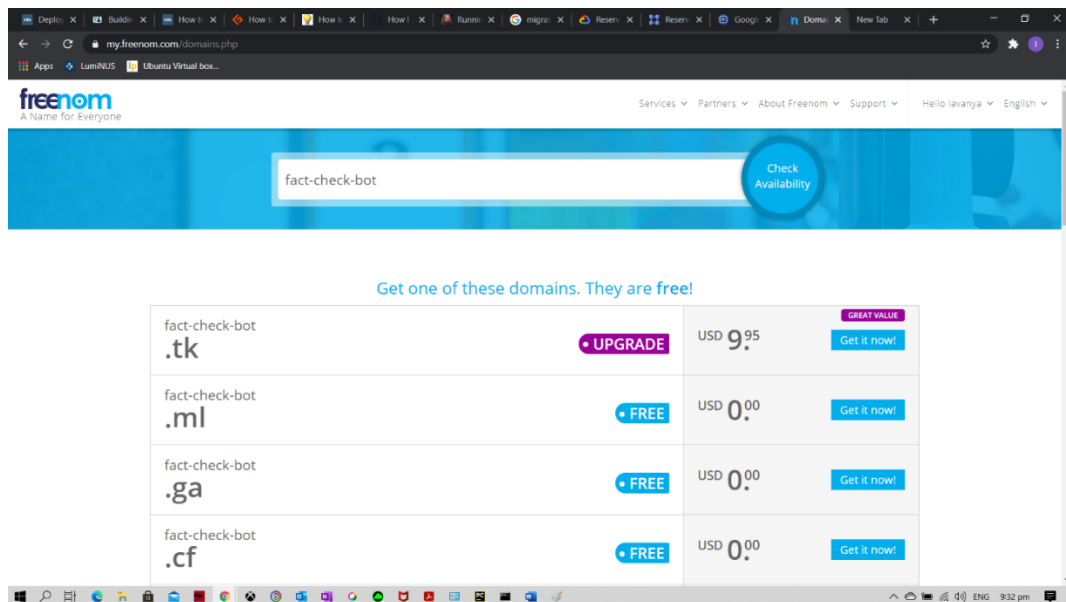


e) Click on **Get it now**

f) Enter the **static IP Address** in the 2 IP Address fields shown below and click on **Continue**

g) Check **TnC** and click on **Complete Order**



## Configuring the Compute Engine Instance:

1. Configuring the Compute Engine Instance:

   a) Navigate to list of Compute Engine Instances at Compute Engine > VM instances

   b) Click SSH Button to login to instance from your browser



2. Setup your environment. Enter below commands in the SSH Terminal.

```
# update system packages and install the required packages
sudo apt-get update
sudo apt-get install bzip2 libxml2-dev libsm6 libxrender1 libfontconfig1

# clone the project repo
git clone <Enter the GITHUB Repo URL>

# download and install miniconda
wget https://repo.anaconda.com/miniconda/Miniconda3-4.7.10-Linux-x86_64.sh
bash Miniconda3-4.7.10-Linux-x86_64.sh
```

Follow the instructions and agree to the terms to install Miniconda

```
export PATH=/home/<your name here>/miniconda3/bin:$PATH

rm Miniconda3-4.7.10-Linux-x86_64.sh


# confirm installation
which conda

# create and activate a new environment
conda create -n telegram-bot python=3.8
conda activate telegram-bot
```

Close and reopen SSH Terminal for the changes to take effect.

Install requirements:
```
# go to project root and install the requirements
cd SureBoT/Code
pip install -r requirements.txt
```

3.  Setting up Server with Gunicorn and NGINX

a)  Install and start NGINX using the commands below

```
cd
sudo apt-get install nginx-full
sudo /etc/init.d/nginx start
```

b)  Remove default configuration file and create a new site configuration file for our application

```
# remove default configuration file
sudo rm /etc/nginx/sites-enabled/default

# create a new site configuration file
sudo touch /etc/nginx/sites-available/telegram-bot
sudo ln -s /etc/nginx/sites-available/telegram-bot /etc/nginx/sites-enabled/telegram-bot
```

c)  Edit configuration file for our app by opening file with an editor

```
sudo nano /etc/nginx/sites-enabled/flask_project
```

d)  Copy and paste below code and save configuration file

```
server {
    proxy_read_timeout 400;
    proxy_connect_timeout 400;
    proxy_send_timeout 400;
    listen 80;
    server_name www.fact-check-bot.ml;
    location / {
        proxy_pass http://0.0.0.0:5000;
```

```
    }
}
```

e)  Restart NGINX Server

```
sudo /etc/init.d/nginx restart
```

4.  Implement Let's Encrypt TLS certificate in NGINX.

a)  Execute below to install certbot plugin

```
sudo apt-get install software-properties-common
sudo apt-add-repository -r ppa:certbot/certbot
sudo apt-get update
sudo apt-get install python3-certbot-nginx
```

b)  Execute below command to modify necessary file to configure certificate

```
sudo certbot --nginx
```

You will see something like below for configuring the certificate. Follow the steps accordingly.

```
root@instance-1:/etc/nginx/sites-available# certbot --nginx
Saving debug log to /var/log/letsencrypt/letsencrypt.log
Plugins selected: Authenticator nginx, Installer nginx
Starting new HTTPS connection (1): acme-v01.api.letsencrypt.org
No names were found in your configuration files. Please enter in your domain
name(s) (comma and/or space separated)  (Enter 'c' to cancel): bloggerflare.com
Obtaining a new certificate
Performing the following challenges:
http-01 challenge for bloggerflare.com
Waiting for verification...
Cleaning up challenges
Deployed Certificate to VirtualHost /etc/nginx/sites-enabled/default for
bloggerflare.com
Please choose whether or not to redirect HTTP traffic to HTTPS, removing HTTP access.
-------------------------------------------------------------------------------
1: No redirect - Make no further changes to the webserver configuration.
2: Redirect - Make all requests redirect to secure HTTPS access. Choose this for
new sites, or if you're confident your site works on HTTPS. You can undo this
change by editing your web server's configuration.
-------------------------------------------------------------------------------
Select the appropriate number [1-2] then [enter] (press 'c' to cancel): 2
Redirecting all traffic on port 80 to ssl in /etc/nginx/sites-enabled/default
-------------------------------------------------------------------------------
Congratulations! You have successfully enabled https://bloggerflare.com
You should test your configuration at:
https://www.ssllabs.com/ssltest/analyze.html?d=bloggerflare.com
-------------------------------------------------------------------------------
IMPORTANT NOTES:
 - Congratulations! Your certificate and chain have been saved at:
   /etc/letsencrypt/live/bloggerflare.com/fullchain.pem
   Your key file has been saved at:
   /etc/letsencrypt/live/bloggerflare.com/privkey.pem
   Your cert will expire on 2018-05-27. To obtain a new or tweaked
   version of this certificate in the future, simply run certbot again
   with the "certonly" option. To non-interactively renew *all* of
   your certificates, run "certbot renew"
 - If you like Certbot, please consider supporting our work by:
   Donating to ISRG / Let's Encrypt:   https://letsencrypt.org/donate
   Donating to EFF:                    https://eff.org/donate-le
root@instance-1:/etc/nginx/sites-available
```

c) After doing above steps, open the site configuration file to view changes done by certbot. Your configuration file should now look something like below.



5. Install Redis-server:

```
sudo apt update
sudo apt install redis-server
sudo systemctl status redis-server
```



6. Replacing your bot token and the download URL for the Pretrained models used in the projects.

a) Change directory to the Github repo

```
cd SureBoT/Code
```

b) Replace **"<YOUR-BOT-TOKEN>"** with the token generated when creating the chatbot

c) Replace the **"<MODEL-DOWNLOAD-URL>"** with the URL provided in the ReadMe file

7. Run Supervisord config file to start and test your domain name in browser.

a) Start Gunicorn & Celery daemon processes using below command.

```
supervisord -c supervisord.conf
```

b) Enter your domain name in a browser and test the output
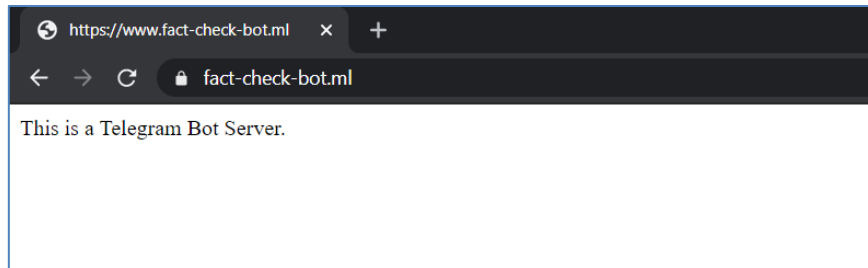


8. Set Webhook for Telegram bot

Call the below API using chrome browser.

[https://api.telegram.org/bot<BOT-TOKEN> /setWebhook?url=https://<YOUR DOMAIN NAME>](https://api.telegram.org/bot<BOT-TOKEN>)

9. The logs generated by the Bot can be checked by inspecting below files

a) gunicorn_log.log

b) celery_log.log

```
cd SureBoT/Code
tail -f celery_log.log
tail -f gunicorn_log.log
```

## APPENDIX C: Individual Project Report

| Your Name: | Kuek Yong Jie Adriel |
|---|---|
| Certificate: | Graduate Certificate in Intelligent Reasoning Systems |

### 1. Personal Contribution

In this project, my contributions can be broadly categorised as follows:

- Business value proposition and Project Management
- Core system architecture design and implementation
- Analysis and fine-tuning
- Report documentation and video-content creation

The team had a common interest to explore a project using Natural Language Processing (NLP) to tackle a potential problem use-case. Through the progress of self-research, I found that the issue of tackling misinformation and fake news remains a pertinent problem today. With the proposal to the team, we then embarked on a series of research and market-survey to understand the problem statement and identify possible areas where Artificial Intelligence could be applied as a solution. When the team had decided to embark on building a fact-checking tool, I proceeded to design and propose an NLP component framework. The architecture framework was based-off inspiration that I have gathered from a couple of state-of-the-art (SOTA) paper implementations on the task of fact verification and information extraction. Together with some brainstorming, I helped guide the team to shape the end-to-end framework and custom it to our local Singapore context to develop the Intelligent Base Processing Unit (iBPU).

During the implementation of the iBPU, there were many challenges posed that hindered the technical realisation of the system. As identified, most of the existing pretrained models do not readily adapt well to what we had in mind for the system due to an incorrect knowledge model application. As such, there was a need to perform a series of ablation studies at the component-level to understand in detail how each block positively or negatively affects the overall performance. One of the key examples was the decision on the hyperparameter beam-alpha value for the output of the abstraction summary. To arrive at the most optimal solution, I tested over a series of pretrained models on varying datasets and tuning values to closely match the training parameters that were used to develop the model for the Graph Attention Network.

Consecutively, I had to perform a series of knowledge modelling to ensure that the relevant information would be captured in the final model that best reflects the actual testing environment. In addition, during fine-tuning, I utilised various techniques gained over the course of the programme such as Early-Stopping on training to prevent overfitting, data pre-processing and cleaning to ensure an accurate input to help boost the overall performance of the system.

Finally, I helped to provide project management guidance using Gantt-Scheduling to keep track of the team's progress throughout the project timeline. On top of this, I had the opportunity to explore various presentation toolkits such as using an AI-enabled voice-over narration for video content generation, and team software co-development and co-creation utilising git.

### 2. Learning Journey

Through the project journey from conceptualisation, implementation to knowledge documentation, I have been exposed to various fields of knowledge. These were gathered through the process of

self-discovery and knowledge-sharing with my project mates. Broadly-speaking, I had a better appreciation on understanding the balance between creating a marketable product for profit and one that addresses a critical issue that helps enrich the lives of people.

In addition, I also had a deeper understanding of applying cognitive-modelling and Natural Language Understanding to solve problems and create valuable tools. Overall, this project has increased my confidence in building intelligent systems that are based on well-grounded knowledge and simple components to realise a much larger and more complex system that can tackle the difficult and real-world problems we see today.

## 3. Knowledge Application

The knowledge acquired through the course and the project will be highly relevant to my work in defence science research and development. It has broadened my perception towards transiting systems from merely a paper study to something that is tangible and useful. In research and development, it is not merely just fulfilling an idea or a concept, but to produce and transit workable systems to provide solutions that creates value and are game-changing. I am confident that given a problem task, I would be able to design and select the best tools to produce a decent outcome, followed by iterative fine-tune and improvements to create a software solution that best addresses it.

| Your Name: | CHUA HAO ZI |
|---|---|
| Certificate: | Graduate Certificate in Intelligent Reasoning Systems |

## 1. Your personal contribution to the project

In this project, I am responsible to build the model pipeline that encompasses "Article Retrieval", "Text Summarization" and "Evidence Selection" phases, as well as creation of the Marketing Video for SureBoT. This model pipeline mainly consists of using Python and is built upon a series of Natural Language Processing (NLP) techniques and models.

In the "Article Retrieval" stage, it starts off when user provide an input claim to the SureBoT. This claim can be either a sentence, paragraph or a URL link to an article website. For URLs, the pipeline will extract the raw full text of the article using requests API and Newspaper3k library. If the input texts are too long, the pipeline will extract the salient points out using abstractive summarization. The model used is a Transformer-based encoder-decoder model framework named PEGASUS. Based on the summarized inputs, the pipeline will run a real-time news article web scrapping process from Google News using pyGoogleNews and Newspaper3k APIs. Top-N articles are then retrieved, where N is set to 5 as default.

In the "Text Summarization" stage, the raw full texts of the top N searched articles extracted in the previous stage will then be summarized using the same PEGASUS model.

Lastly, in the "Evidence Selection" stage, the pipeline will execute the filtering decision phase, where it will return articles that are most relevant to the input query. The model used is the state-of-the-art Sentence-BERT Siamese network, where it will create sentence embeddings for the articles and query, followed by computing cosine-similarity between them. Threshold filter score of 0.4 is used to remove noisy data. The list of filtered articles are then processed in the later stages of "Claim Verification" and "Evidence Reasoning Net" to produce classification outputs of "Support Claim" or "Refute Claim", else "Not Enough Evidence".

In addition, as part of the project requirements, I also created the marketing video to demonstrate the business case of our SureBoT. A short walkthrough of the chatbot is also included. The video demonstrated the necessity and the impactful change that SureBoT can bring about to the society, and the convenience, accessibility and ease of use that SureBoT brings to people.

## 2. What you have learnt from the project

Through this project, I have learnt a lot. As the main script is in Python language, this project has improved greatly my programming ability, which will be useful in the workplace or future works. Also, I learnt to explore, utilize and compare various open-source pre-trained models in various websites like Github or Huggingface. This is particularly essential where in order to get the right results, one has to evaluate what can be the right model to be used. And this experience acquired in this project will serve greatly in future projects in terms of model-building or model-implementation.

In addition, I am able to utilize the concepts learnt in the course into this project. This includes NLP-related tasks like text pre-processing and cosine-similarity reasoning, retrieval-based approach where we implement sentence embeddings and using encoder-decoder model framework, as well as the flow of how to create and host a functional chatbot on social media like Telegram. This project allows me to better understand how to implement reasoning in intelligent system and build the end-to-end model pipeline to retrieve the outputs that we need.

This project also allows me to gain deeper insights to how the backend architecture of the chatbot

is created. The establishment of the architecture that extends from setting up API to telegram for chatbot creation to setting of cloud architecture on Google Compute Engine to host the chatbot provides greater exposure and understanding of how the backend architecture works, which will be useful for future work projects.

From this project, teamwork is very important. This project involves building up the backend architecture to host the chatbot on telegram and building up the model pipeline which can accurately determine if a certain claim / query is true or false or undetermined due to insufficient information. This requires the unique expertise of each team mate to work on different sections of the work, before combining them together to form an end-to-end hybrid system. Scheduling of project tasks are also in place to ensure everyone is able to achieve their project deliverables on time.

## 3. How can you apply this in future work-related projects

Python is a highly common and popular programming language that is widely adopted by many organizations. Strengthening my understanding and programming ability of Python will enable me to produce high-quality scripts that are of production-level, which is crucial in all organizations. This particular pipeline where it involves web scraping and doing similarity reasoning can be applied in other automation projects in the workplace that requires interaction with websites / systems and extraction of relevant information.

Project management knowledge or tools (like Gantt Chart scheduler) implemented in this project are precious techniques and methodologies that can be utilized for future projects in the workplace. More importantly, the ability to build and implement an end-to-end system architecture for the intelligent systems is useful and crucial for subsequent development of intelligent systems in the work environment.

| Name: | M LAVANYA |
|---|---|
| Certificate: | Graduate Certificate in Intelligent Reasoning Systems |

### 1. Personal Contribution to group project.

In this project, my task was to design and build the Back-end architecture for the Telegram chat bot which serves as a container for the Fact-checking pipeline.

The Bot server is a Flask Application that directly calls the Telegram Bot API's without using any wrapper libraries (e.g., python-telegram-bot). This was done so that it will be easier to customize the Bot to our requirements. Initially, the Bot was made to poll Telegram Server periodically to get the list of messages sent by users and I hosted it on Heroku. But I realised that Heroku could not be used for hosting the server as we could not even install pyTorch due to the limit of 500mb memory. So, we checked for alternatives and I decided to host the Bot on Google Cloud Engine.

Also periodically polling the Telegram server for messages did not suit our use case so I switched to using Webhooks that allows our server to receive a request whenever user sends a message. To support this, we had to create a domain and SSL certificate because Telegram server only allows us to set a Webhook if the URL is secure. And I had to integrate Gunicorn HTTP Server and Nginx framework with Flask to support reverse proxy and load balancing. Once the request reaches the bot server, I extract the query and chat Id from the payload and double confirm the user's intention to Fact-check by using Telegram's Inline Keyboard. After user confirmation, I helped in implementing text pre-processing and extracting text from an Article when user sends a message with a URL before processing the query.

When integrating the Fact-checking pipeline with the Bot server, I faced some issues as the pipeline process is memory and computation intensive and by the time the execution ends the request from the Telegram Server times out. Hence, I had to find a way to handle the pipeline execution as a background task and inform the user that their request is being processed. For this, I did some research and decided to use Celery and Redis to handle the background tasks. Redis is an in-memory data structure that stores the jobs with the information of the query and user's chat id and passes it to the Celery worker. Once the celery worker completes the pipeline execution, I helped in formatting the changes into a user readable format and send it back to the user using the Flask Application's context.

After the integration and successful hosting of the server, our team started testing the Bot and I helped with debugging and fixing the issues that we encountered to improve the Bot's performance.

### 2. What learnt is most useful for you?

I joined this course with an interest in the field of AI but had no real knowledge or work experience. Working on this project has been an eye-opening experience.

With the help of my teammates, I had the opportunity to understand and experience first-hand on how an NLP system can be implemented. When debugging and fixing some of the issues that we encountered while testing the Bot, I was able to understand how the NLP pipeline works and see how the different processes and concepts like text pre-processing, web scraping, using pre-trained models, etc are crucial for an NLP system. This has really helped me in further understanding and appreciating the concepts that were taught in the Intelligent Reasoning Systems certificate.

Being a Frontend Developer, I had no prior knowledge of backend and had to learn about the backend tools from scratch and implement them based on the needs of the project. And with every issue that I encountered in the backend implementation I learned new concepts and even had to change the backend system design a few times.

Lastly, working with my team has been a pleasant experience and has shown me how important teamwork is for a project, especially when we all are working on different aspects of the project.

**3. How can you apply the knowledge and skills in other situations or your workplaces**

I was initially worried on how my current work experience as an Android Developer would help me since AI is a completely different field. But now I am reassured that for any project, to package it as a complete standalone system you would probably require both frontend and backend systems.

Also, one of the main reasons I took up this course was in the hope that I would be able to design and implement Intelligent Mobile Applications or any system that can improve user experience by utilising AI principles. Thanks to my team, by working on a project that heavily utilizes Natural Language Processing (NLP) concepts my knowledge base has widened and has given me an idea of how an end-to-end NLP system needs to be designed.

| Your Name: | Chammanikodathu Louis Francis |
|---|---|
| Certificate: | Graduate Certificate in Intelligent Reasoning Systems |

## 1. Your personal contribution to the project

Explored and learned about the implications of fake news and misinformation. Learned about the economic, social and political consequences of the misinformation and exposure of the Singaporeans to the misinformation. Contributed to the project initiation and discussions.

Researched about the proliferation of misinformation in social media and particularly in US and Singapore. Documented the project proposal.

Researched about the various web scraping libraries in python and done the coding using python and newpaper3k for the news article retrieval, summarization. Done the implementation of the top n articles.

## 2. What you have learnt from the project

As my past experience was mainly in conventional software development this project was a new experience for me in terms of learning and applying NLP and machine learning as a part of this project. The application of frameworks like Sentence-BERT Siamese network gave me a very good foundation of how to use a pre-trained model for developing a model pipeline.

I also had a chance to enhance my team work, collaboration and project management skills working together as a part of a dedicated and talented team.

The python skills I had managed to pick up as a part of this project helped me to reinforce the NLP concepts taught as a part of the curriculum. The backend architecture of the application gave me a good understanding of how to deploy applications on Google Cloud Platform making use of Nginx and Gunicorn HTTP server. Python Flask framework picked up as part of this project gave another perspective about web development using Python.

## 3. How can you apply this in future work-related projects

Python skills like Flask and application of NLP and Bot frameworks will help me in applying the same concepts and ideas for Bot and NLP related projects for work. At work for the real estate domain, we have a need to analyze customer queries which come through the chatbot and to recommend products. The NLP concepts and the application of frameworks like Sentence-BERT Siamese network will help me in applying these skills in the real estate chatbot project.

Smart Buildings is another project where I will have a chance to apply the NLP and machine learning models learned as a part of this project.

## APPENDIX D: Mapped System Functionalities against knowledge, techniques and skills of modular courses

| Modular Courses | System Functionalities / Technique Applied |
|---|---|
| Machine Reasoning (MR) | • **Knowledge Elicitation and extraction**<br>Web crawling from websites to form Knowledge Base (KB) |
| Reasoning System (RS) | • **Knowledge Representation**<br>Graph Neural Network (GraphNN) Fact Verification (BERT-base Knowledge Graph) |
| Cognitive System (CGS) | • **NLP Tasks**<br>  o Text Pre-processing, and text summarization due to long length of article<br>  o Cosine-Similarity Reasoning for Evidence Selection<br>• **Retrieval Based Approach**<br>  o BERT encoder-decoder model framework<br>  o Sentence Embeddings<br>• **Cognitive Systems**<br>  o Chatbot Development |

# APPENDIX E: Performance Survey

Survey Link:  https://bit.ly/3veVD7h

## Survey results:

| Aggregate Score | 6.17 | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | |
| Timestamp | Query 1 | [Query 1] Relevancy Accuracy Score | Query 2 | [Query 2] Relevancy Accuracy Score | Query 3 | [Query 3] Relevancy Accuracy Score | Feedback (if any) |
| 4/17/2021 0:21:55 | covid 19 vaccine triggers stroke in doctor after jab | 5 | Moderna vaccine proves better efficacy compared to Pfizer vaccine in covid 19 prevention. | 6 | Chan Chun Sing becomes the next PM of Singapore | 9 | |
| 4/17/2021 9:13:09 | Singapore will remove all hawker centres | 6 | Heng Swee Keat is the PM for Singapore | 2 | Pfizer vaccination is not effective | 9 | |
| 4/17/2021 21:27:31 | Covid can be prevented by wearing mask | 8 | Covid cannot be prevented by wearing mask | 7 | Wearing mask effective in minimizing covid | 4 | |
| 4/18/2021 17:26:57 | Lee Hsien Loong is Singapore PM | 4 | Covid-19 spread by air | 8 | https://www.channelnewsasia.com/news/singapore/malaysia-singapore-compensation-high-speed-rail-14517142 | 4 | |
| 4/18/2021 18:01:02 | Covid-19 vaccination got side effects | 9 | https://www.usatoday.com/in-depth/money/2020/05/12/coronavirushow-u-s-printing-dollars-save-economy-during-crisis-fed/3038117001/ | 6 | Travel bubble between Singapore and Hong Kong | 5 | |
| 4/18/2021 21:55:56 | https://blackdotresearch.sg/brain-haemorrhage-covid-19-vaccine/ | 5 | https://blackdotresearch.sg/vaccination-centre-moderna-pfizer-singapore/ | 4 | https://www.theonlinecitizen.com/2021/04/17/covid-death-toll-passes-three-million-as-india-cases-surge/ | 9 | |
| 4/18/2021 22:14:35 | https://www.theonlinecitizen.com/2021/03/23/maruah-singapore-calls-on-individuals-to-submit-three-finger-salute-photos-in-solidarity-with-the-people-of-myanmar/ | 5 | https://www.theonlinecitizen.com/2021/03/08/australia-suspends-defence-cooperation-with-myanmar-redirects-aid-to-ngos-amid-escalating-violence/ | 7 | Prince Philip is dead | 10 | |
| 4/18/2021 23:14:38 | Is Pluto a planet? | 3 | Is Shanghai in China? | 9 | Singapore is part of Malaysia? | 3 | |
| 4/19/2021 9:48:55 | Tanjong pagar bmw accident was caused by dangerous driving. | 9 | Massive floods seen all around Singapore after huge thunderstorm. | 10 | Singapore weather to dip to a low of 22 degrees in the coming weeks. | 10 | |
| 4/19/2021 20:37:14 | Is fact-checking profitable? | 3 | Singaporeans can travel to Malaysia from May 2021 | 4 | Employees are not allowed to work from home from April 2021 | 8 | |
| 4/19/2021 20:47:07 | Actor Vivek passed away due to Covid 19 | 5 | Singapore is a tropical country | 2 | IT professionals earn higher than those in other professions | 2 | |
| 4/19/2021 20:58:14 | Australian wildfires 2020 was caused by humans | 7 | Prince Harry did not attend Prince Philip's funeral | 8 | Megan Markle attended Prince Philip's funeral | 10 | |
| 4/19/2021 21:07:25 | First Covid-19 case originated in China | 9 | Corona virus is helping to reduce $CO_2$ emissions | 2 | Apple is holding an event on April 20, 2021 | 10 | |
| 4/19/2021 22:13:01 | Lee-Se-dol defeat AlphaGo? | 9 | Is it difficult to detect deepfake videos? | 8 | Is there advertisements on Youtube? | 8 | |
| 4/20/2021 23:04:47 | Did London Bridge collapsed? | 7 | Do cats have nine lives? | 1 | Is mRNA vaccines safe? | 9 | |
| 4/21/2021 0:03:00 | Is PAP the leading political party in Singapore? | 5 | Is walking more beneficial than jogging? | 7 | What is the nearest MRT station to Bukit Timah Hill? | 4 | |
| 4/21/2021 0:40:11 | Is Singapore part of china? | 6 | Who is the prime minister of Singapore | 8 | Is chr kyaw teow a Singaporean food? | 8 | |
| 4/21/2021 20:08:48 | Is portsmouth in singapore? | 1 | What is 1 inch in centimeter? | 2 | What is the capital of Switzerland? | 1 | |
| 4/22/2021 23:10:01 | Is Donald trump president? | 3 | Lewis Hamilton won 200 Grand Prix | 2 | Ted Mosby is a real person | 8 | |
| 4/24/2021 15:24:10 | Singapore cabinet reshuffle sees major changes to half of the ministries | 10 | Singapore cabinet reshuffle sees major changes to all ministries | 10 | US President Joe Biden receives lower popularity vote since election | 7 | |