Industry Oriented Mini Project Report

On

QUERY BOT

Submitted in partial fulfillment of the requirements for the award of Degree of

Bachelor of Technology

In

Computer Science and Engineering

By

AHMED ABDULLAH (17241A0563)

Under the Guidance of

Mrs. B. PADMA VIJETHA DEV,

Assistant Professor

Department of Computer Science and Engineering

GOKARAJU RANGARAJU
INSTITUTE OF ENGINEERING AND TECHNOLOGY
(Autonomous)

Bachupally, Kukatpally, Hyderabad-500090.

2019-2020

GOKARAJU RANGARAJU INSTITUE OF ENGINEERING AND TECHNOLOGY

(Autonomous)

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the mini project dissertation entitled **"QUERY BOT"** that is

being submitted by

AHMED ABDULLAH	**(17241A0563)**
MASANIPALLY SATHWIK	**(17241A0592)**
VISHESH AGARWAL	**(17241A05C0)**

in partial fulfillment of the requirement for the award of the degree in **Bachelor of Technology** in Computer Science and Engineering during the academic year 2019-2020.

Mrs. B. PADMA VIJETHA DEV
Guide

Dr. K. Madhavi
Head of Department

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

There are many people who helped us directly and indirectly to complete our project successfully. We would like to take this opportunity to thank one and all.

First of all, we would like to express our deep gratitude towards our internal guide**, Mrs. B. Padma Vijetha Dev, Assistant Professor,** for her help and constructive criticism in the completion of our dissertation.

We thank our mini Project coordinator **Dr. G. R. Sakthidharan** for his extensive Support. We would like to thank our faculty Coordinators **Ms. G. Padmaja** and **R. N. Ashlin Deepa** for giving consistent suggestions and inputs during the project period.

We bestow our sincere thanks to our HOD, **Dr. K. Madhavi, Professor,** for providing ample environment and exceptional encouragement to complete this project. We thank our Principal **Dr. J. Praveen** for providing the internal resources and facilities to complete the dissertation.

Finally, we are very much indebted to our parents for their moral support and encouragement to achieve our goals.

AHMED ABDULLAH

(17241A0563)

MASANIPALLY SATHWIK

(17241A0592)

VISHESH AGARWAL

(17241A05C0)

DECLARATION

We hereby declare that the project work titled **"QUERY BOT"** is the work done during the period from **09-12-2019 to 04-04-2020** and is submitted in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering** from **Gokaraju Rangaraju Institute of Engineering and Technology**. The result embodied in this project have not been submitted to any other university or Institute for the award any degree or diploma.

AHMED ABDULLAH
(17241A0563)

MASANIPALLY SATHWIK
(17241A0592)

VISHESH AGARWAL
(17241A05C0)

ABSTRACT

A College enquiry bot is very essential these days for providing information to non-college/college students about the college. Many students do not have any information regarding the college and the information that is present online is very less and hence, less people know about the working of college and students become very uncertain whether to take admissions or not. A college enquiry chatbot understands the user's message, what type of information is being enquired and then it answers the query accordingly. The user can ask queries and the chatbot analyses the questions, interprets the meaning using a tag and intent and provides an answer. There is also the option of having a conversation about anything with the user. The Chatbot uses the built-in intelligent brain which is constructed using deep NLP to converse with the user. The answers are appropriate to what the user asks. The benefit having such type of chatbot is that the user does not have to personally visit the college to enquire about the details and also can have a conversation about any topic if the user prefers. The chatbot answers the questions as if it were answered by a human because the dataset it gets trained on contains many human-to-human conversations making it easier for the chatbot to understand the way humans converse. The user can also ask many college related activities such as date and timing of annual day, sports day, academic calendar and other cultural activities. This system helps the student to be updated about the college activities and the non-students to gain information about various details about the college and its working.

Keywords: Deep NLP, Training, Brain, Intelligent, Queries, Interpretation power, Combination, Retrieval, Generative

CHAPTER	TITLE	PAGE No.

CHAPTER 1 INTRODUCTION

1.1 INTRODUCTION

The fields of data science and artificial intelligence have progressed rapidly over the past decade. The ability to think creatively like a human is what makes the systems developed with AI more beneficial from other systems. Over 73 percent of the businesses around the world prefer to use AI to assist them if it can help them to gain profit and sustain themselves. More than three quarters of the consumers worldwide use chatbots as technical support for their customers as these chatbots are able to efficiently solve the customers' problems eliminating the need for human workers thereby cutting the expenditure of a large number of companies.

CHATBOT:

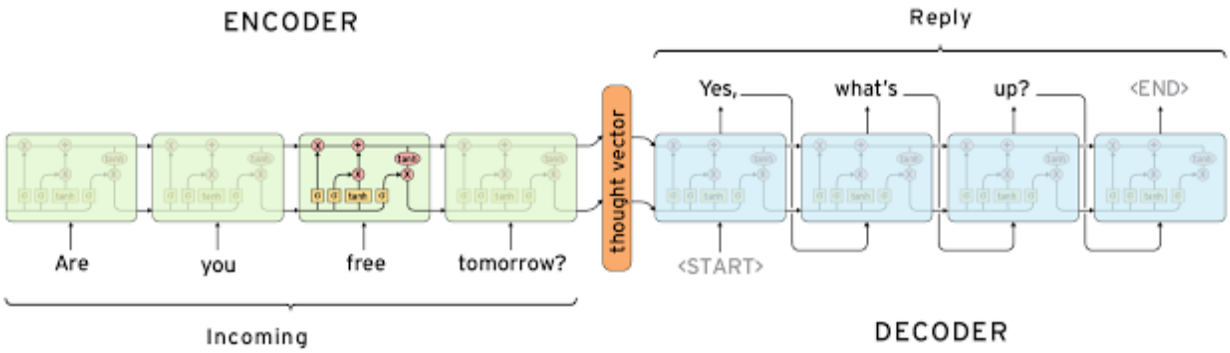
A chatbot is an intelligent software that can communicate and perform actions similar to a human. Chatbots are used in a variety of fields such as virtual assistant, customer and technical support, and for company promotions on many social media platforms.

There are two types of chatbots that are classified using the purpose of their design and the type of information they will provide to the user.

- Retrieval Based Chatbots – These chatbots fetch the input question and its answer directly from the database where the question's context is already predefined. These chatbots are very easy to build using simple NLP libraries of Python.
- Generative Based Chatbots – These chatbots are very hard to build as they require hundreds of conversations between users based on which they train themselves to predict their own answers.

GENERATIVE BASED CHATBOT:

The project is based on a combination of generative and retrieval based chatbot. The generative chatbot is not built using predefined responses and instead, is based on a number of human conversations about college related information. They are open domain, meaning they are not confined to any goal. They are also able to make small talk with the user. They require large conversational data to train themselves. A generative chatbot focuses to perform machine translation methods and translate from an input to an output response based on an LSTM learning approach to predict the answers. The conversations which are very lengthy are more difficult to process and create answers.



RETRIEVAL BASED CHATBOT:

The retrieval based chatbot has a set of predefined questions and responses which it fetches back to the output and displays. These chatbots are very accurate and display the correct response always. These chatbots are very useful for specific enquiries i.e. college related enquiries. They follow a heuristic approach in selecting an output answer from the set of answers in the database. It also checks the required input and context to find that particular answer set in the database to answer the input query put forward by the user.

USING A COMBINATION OF BOTH THE MODELS:

Both the models have their list of advantages and disadvantages. Since the retrieval model has the answers already present in the database and the answers only have to get selected, it has very less chance of making any grammatical errors. Although, the answers they display as output are very accurate and correct, they are not able to display any answer for the query which is not present in the database.

On the other hand, generative models are very intelligent. They refer to previous answers to understand the context and accordingly answer the next question. They infer from previous inputs and give the user and impression that they are conversing to a human. The downside of using generative models is that since they create their own answers, they are prone to lot of grammatical errors in especially lengthy sentences and require millions of conversations of data of human conversations to train themselves.

Deep learning methods are efficient for both of these models but are more beneficial and efficient in generative models.

LENGTHY AND SHORT CONVERSATIONS:

Inputs and outputs with long sentences are very complicated for the chatbot to understand and process. The easiest conversations that can be processed by the generative model are very short(one-word) conversations like Hello- Hi. These short conversations with only one-word input and output do not take large training data. Only when training long conversations where a sentence contains more than 5-6 words, or more, then the data required for training it increases exponentially.

CLOSED DOMAIN AND OPEN DOMAIN CHATBOTS:

In a closed domain chatbot, the inputs and outputs are predefined so the spectrum of question and answers is very restricted. For example, during technical or customer support provided to customers by many businesses use such type of closed domain bots since their goals is very specific i.e. to provide assistance to the end user. The users can ask questions of any context, but the system would not have the capability to understand and answer those questions and will often give an incorrect answer.

In an open domain chatbot, the chatbot is trained and can hence handle any sort of question thrown at it by the user. There is no defined goal or any intent or context. Conversations can go anywhere and for that reason, knowledge of a vast number of topics is required by the chatbot to be able to talk like a human.

PROBLEMS THAT ARE FACED BY CHATBOTS:

LOGICAL STRUCTURE AND PERSONALITY:

When the outputs are predicted, the outputs must sound logical to the type of input provided. The outputs may be correct grammatically and according to the language, but often fail to provide correct semantic answers.

PROVIDING A VARIETY OF ANSWERS:

A lot of chatbots often tend to produce normal answers like “I do not know”, “yes, you?”, “ I think so” and there isn’t very variety in the type of answers produced. Since the generative models are not having any defined goal, they often face this problem to provide variety in their answers.

HOW EFFICIENT CAN OVERCOMING THESE PROBLEMS HELP:

A chatbot of the retrieval model having an open domain is very difficult to create since there has to be a dataset with at least a million conversations relating only to the particular topic. Such dataset can easily take many months to create and hence they are deemed almost impossible and not worth the effort.

Overcoming these problems can assist in making the chatbots more efficient and able to predict the answers more accurately. Using a combination of generative and retrieval models can reduce the problems faced by both the models and makes the chatbot better equipped to answer all sorts of queries/input questions asked by the user.

LEVEL OF INTELLIGENCE OF GENERATIVE CHATBOT:

The chatbot does not fetch any answer that is already present in the database for the required query. Instead, it constructs its own sentences similar to a human being, using the context obtained by training of the chatbot, by analyzing many conversations between different human users.

The chatbot consists of 3 layers which make it able to easily analyze and interpret the answer very quickly within minimal time. The chatbot is constructed using the python version 3.5 which makes it compatible to be used on any windows computer even with very low system configuration.

1.2 DOMAIN DESCRIPTION

1.2.1 DATA SCIENCE

Data Science is a field that uses a number of processes, algorithms, and scientific methods to understand, analyze and process the structured and unstructured data. The ability to take large amounts of data and understand the data, process the data and extract valuable information from it is the main essence of data science. Data science allows artificial intelligences to create solutions to a variety of problems by using the previous data for use in the future. Data science assists AIs to find and sort information from huge pools of data very efficiently.

IMPORTANCE OF DATA SCIENCE IN HUMAN LIFE:

The world is advancing technologically at a very rapid rate. Organizations are using yottabytes to zettabytes of data. Humans have become depended on data to be stored on various digital platforms such as cloud storages since handwritten data can get destroyed or lost over time but digital data stays forever and can never get deleted on its own so it's a much safer way to save crucial data.

The field of Internet of Things accounts for more than 95% of the data that has been created by the users all across the world. These huge chunks of data of more than a trillion bytes is called as big data. This data comes in many formats such as unstructured and structured.

PREPARING THE DATA:

When we have identified what type of data we have, the next step is to prepare it for processing and analyzing. The procedure is to convert all the data into a format which is easily accessible and understandable to the system. This is called as cleaning the data. Once this is done, an inference has to be gained from this dataset and this along with the dataset is sent for further analysis. Many actions are done such as combining similar information, separating different information in different places, etc.

MATHEMATICAL MODEL:

Every data science model runs on a mathematical model that is created only for that model. Every mathematical model is unique for that data science model to suit the needs of that project. Various mathematical tools are applied to analyze the project and see what it requires. Along with it, various computer languages such as python are used to process it further.

IMPLEMENTATION OF THE MODELS:

When the mathematical models are created and the data is prepared, these models are run based on the prepared data to get the results wanted by the user. If the results provided by the model are incorrect, that would mean either the model has not been trained correctly or there are faults in the data prepared.

ADVANTAGES OF DATA SCIENCE:

- Data science is very flexible and used in a lot of places such as the banking sector, healthcare, IT companies, etc.
- Data science makes huge chunks of data more understandable and easier to understand. The data is easily processed after which, using it can be done in a much faster way.
- Data science takes over the monotone boring tasks which were previously performed by humans
- Health sector used data science to save a lot of human lives by early detection of diseases which previously took a long time to diagnose only at the critical. But now, such diseases can easily be diagnosed at their early stages and treated effectively.
- Artificial intelligence helps various businesses to save money that they normally would have invested to hire workforce to solve the physical labor that is now accomplished by intelligent machines.
- Artificial intelligence is also being used a lot in the field of public entertainment as well. Ais are helpful to create music, help create new kitchen dishes based on user suggestions, and the pay on demand platforms like Amazon prime and Netflix, use the data from the user to show different suggestions of movies the user will likely watch.

1.2.1.1 ARTIFICIAL INTELLIGENCE

Artificial Intelligence is the ability of a machine to mimic human intelligence and perform the tasks that require human intelligence. Despite so many advances in computer technology, there has been no computer made yet, that can compete against the versatility of humans over a number of topics that require everyday knowledge. Computers have the ability to perform very complicated tasks and high-level calculations which would otherwise be extremely difficult for a human to solve. But they generally lack the ability of critical thinking and problem solving at which humans are very efficient. Systems are being developed that are able to imitate human intelligence on a basic level. There is still a long way to go before a virtual brain can be developed that is as good as a human brain.

IMPORTANCE OF ARTIFICIAL INTELLIGENCE:

- Artificial intelligence takes over the tasks that are monotone and take up lots of time for humans.
- Artificial intelligence achieves high level of intelligence through deep learning of its neural networks by understanding lots of data required to create intelligent systems capable of making their own decisions.
- Artificial intelligence thoroughly processes the data and extracts every bit of information from it achieving very low losses on data extraction.
- Artificial Intelligence is very required in medical field to locate tumors inside the human body more accurately than a human can.
- Artificial intelligence adapts itself to make new algorithms everyday so that it can do the required programming itself freeing up a lot of time for users since humans are prone to making a lot of errors during programming.

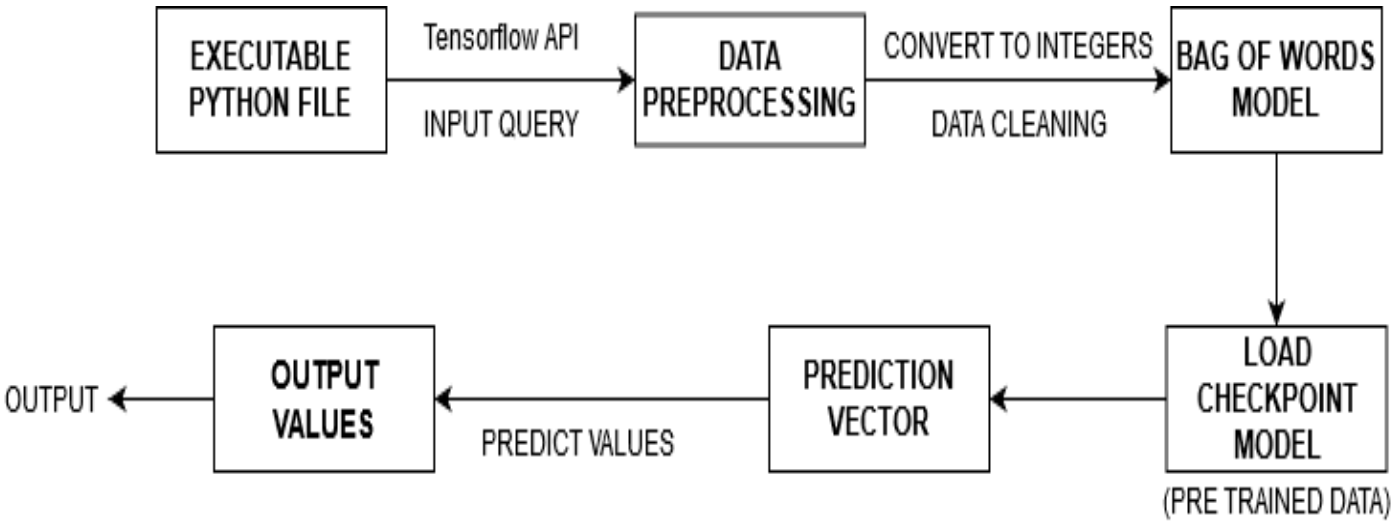
THE LINK

Machine Learning is the link between Data Science and Artificial Intelligence since it is a process of processing and understanding data, and learning from it over a period of time. There are many other things that connect AI and Data Science but Machine Learning works better and more efficiently with Data Science.

1.3 APPLICATION

The chatbot will be trained by a number of queries of different users. It will contain different information about various activities being held in the college and their dates as well as information about the college, faculty and the campus itself rendering it useful for both students and the non-college people. The chatbot can be shared or put up online on any of the college platforms such as the official college website where everyone is given access to download the chatbot. Anyone who has python installed on their personal computers or laptops can execute the chatbot directly which is pre trained and converse with it asking all their queries. As new and new information will be needed to be provided to the people over the course of time, that information can be added to the database, get the chatbot trained again and its updated version can be posted on the website.

1.4 INPUT AND OUTPUT DESIGN



CHAPTER 2 SYSTEM ANALYSIS

2.1 OBJECTIVE OF THE PROJECT

Objective of the project is to develop an intelligent interactive chatbot that is capable of interacting with the user and giving information about the various enquiries made by the user about the college. In addition to answering queries for the user, the chatbot should be able to make a talk with the user making the user feel as if conversing with a human.

2.2 PROBLEM STATEMENT

This project is aimed to develop a python based intelligent chatbot using Natural Language Processing libraries in Python so that the chatbot can interact with the user. It is very difficult for the students who live very far away from the college or other non-college people who have to come to the college to ask basic enquiries. People have to travel to long distances to enquire about information that is not available on the college website. This chatbot will allow them to enquire about various details like upcoming events, faculty information, etc., eliminating the need to go to the college specifically to enquire or calling up the staff to ask them. Along with it, the user will also have the option to converse with a conversation chatbot on any topic if they desire. The chatbots are a combination of both retrieval and generative models, so they are able to answer any specific college enquiry queries as well as have a conversation with the user about almost anything.

2.3 FEASIBILITY STUDY:

STATEMENT OF PROJECT REQUIREMENT:

There are a number of colleges in India where significant amount of students study and get their degree. Before taking any admission into a professional college, they enquire about the college, the faculty, the infrastructure and the college ranking. Most of the times, the colleges require a lot of land space and hence are located far outside the city. This causes difficulty to the people who have to go to the university/college for asking for even basic information. There is an immense need for a software that can answer people's queries and eliminating their need to go to faraway places.

ECONOMIC FEASIBILITY:

The project is very economic friendly. The application used for developing the chatbot is python which is open-source and available for anyone to use. Any institution can easily get the required software tools to develop this project free of cost. The dataset can be easily created by institution in a few days and can be updated and implemented easily. Very less workforce is required to maintain and update the chatbot as it only requires maintenance when updating and training it. It can be used by software companies also with their own defined datasets and used for information broadcasting. The hardware costs are meagre as the construction of chatbot requires a laptop with minimal system requirements. Although there are a few constraints that should be considered when creating the chatbot on an institutional/business level:

- When the project is implemented, at an immediate level there would be a need to hire a few programmers and bug testers to implement the chatbot according to the business'/institution's needs. In the long run, when the chatbot has been effectively trained, there would be no further need of additional programming workforce, just a single bugs tester would be enough to find and remove the bugs from time to time according to the new updates in the chatbot information over time.
- There would be certain costs to do a complete study of the system and creating a plan before a chatbot is created and that will involve significant costs depending on the level of complexity of the chatbot required by the stakeholder.
- There would be considerate amount of savings as there would be lesser requirement for receptionists to be hired to cater to the queries of public which the chatbot can answer itself.

- The resources are enough for the project to be created as of now, but if the chatbot is to be used by bigger MNCs, then the costs to create and regulate the chatbot will also increase simultaneously.

OPERATIONAL FEASIBILITY:

The chatbots have a good response time when they've been trained effectively. When training the chatbot, the retrieval chatbot takes less time as the responses are predefined. On the other hand, the generative chatbot takes a marginal amount of time to train itself. Because the datasets required to train those chatbots are very huge, they take much time to train effectively. The parameters have to be tweaked often as the best parameters can only be defined by trial and error method. But once the training is completed, the generative chatbots tend to be even more effective than the retrieval chatbots as they have the ability to construct answers for the questions not even mentioned in the database. Both the models have their pros and cons, but the effectiveness of using a combination of retrieval and generative chatbot is much higher. The chatbot is very flexible and can be adapted to the consumers' needs.

According to a survey conducted by Spiceworks in 2017 showed that over forty percent of educational institutions all over the world preferred using specialized chatbots as it resulted in less expenditure and hiring of fewer workforce to answer the users' queries. This shows that consumer based chatbots are very essential and are in a huge demand. The users can be encouraged to provide input for the chatbots which can create a user friendly chatbot with the mindset of questions typically enquired by users.

The crucial factors that are important to check the operational feasibility of this project are:

- The current designing of this project helps to project correct and accurate information to the end user. Since the models are trained, they do not show any incorrect information.
- The current project would not allow any breach of information of any form because the data that is only going to be shared by the institution will be put up in the dataset of the chatbot so there is no possible way through which any fraud or abuse of data can take place.
- The project is created to ease the way the users obtain information about the college queries and in doing so, the rate of acceptance of the software by the public will be very high due to the comfort it offers the users to obtain the information.
- The college management will find the project useful as it will cut back the costs to hire more workforce to answer the user's queries.
- The end users, i.e. the people who have various enquiries will save on time and petrol costs and so this project can be seen as a positive change for them.

TECHNICAL FEASIBILITY:

The project is based on python 3.5 version heavily dependent on the API Tensorflow version 1.5. Both of them are stable versions of respective software and API and hence, do not cause any difficulties in the implementation of the program. The manpower required to make the chatbot is very less. A group of 2-3 programmers are sufficient to create the required source code according to specific needs. Testing and implementation are very easy and does not require any workforce. There is a need of debuggers as the project shows huge scope of improvements that can be made and equipped better to suit users' needs. The chatbot can be upgraded using a Tensorflow GPU considerably reducing the training time for the chatbot although it requires a system with high end configuration to process the data quickly. The current technical resources required for the project are sufficient for the implementation of the project and any further resources are not required. They can be implemented to further develop the model based on peoples' needs further in the future. The software and hardware requirements are minimal and can be implemented on almost every present-day laptops and PCs. This specific chatbot is created using particular versions of modules and APIs of python. These modules and APIs are readily available on the python website which can be easily downloaded and installed.

The essential factors required to test the technical feasibility of this project are:

- In India, many people possess the required technical experience required for this project and so this can be created anywhere under minimal costs.
- The project only requires developers to having a knowledge of python language along with concepts of deep learning and natural language processing. An additional knowledge in sequential to sequential chatbot creation would be an added advantage.
- Alternately, a new system can be created by using the updated versions of the python packages to create a much more efficient and faster chatbot which takes less training time.
- The models when being trained should be continuously powered on and any loss of electricity to the system during the training can lead to incorrect training and so therefore, careful monitoring in training is recommended.

2.4 SYSTEM REQUIREMENTS

HARDWARE REQUIREMENTS:

A PC/laptop with

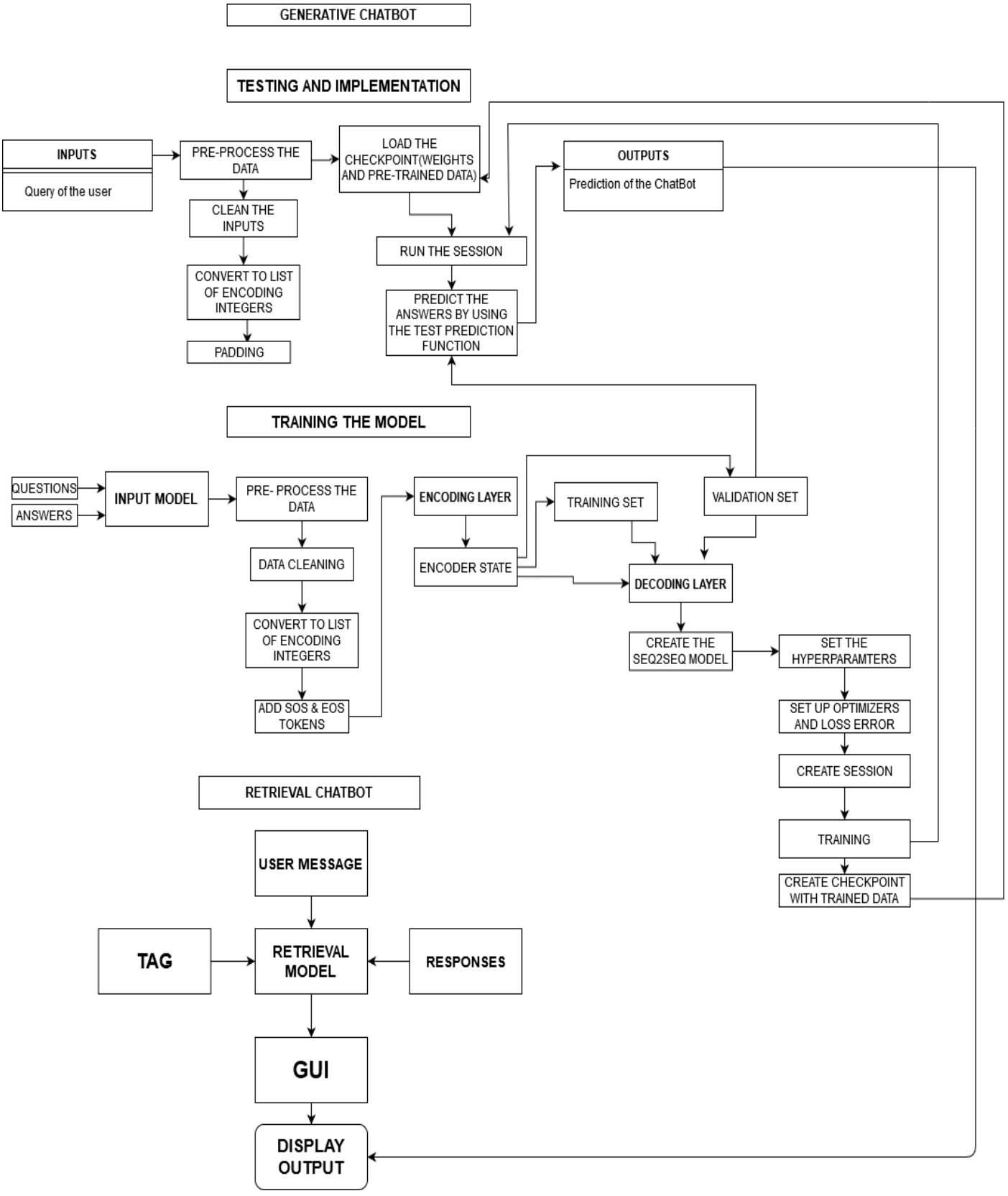
- 2 GB RAM
- 2 GB Internal storage memory

SOFTWARE REQUIREMENTS:

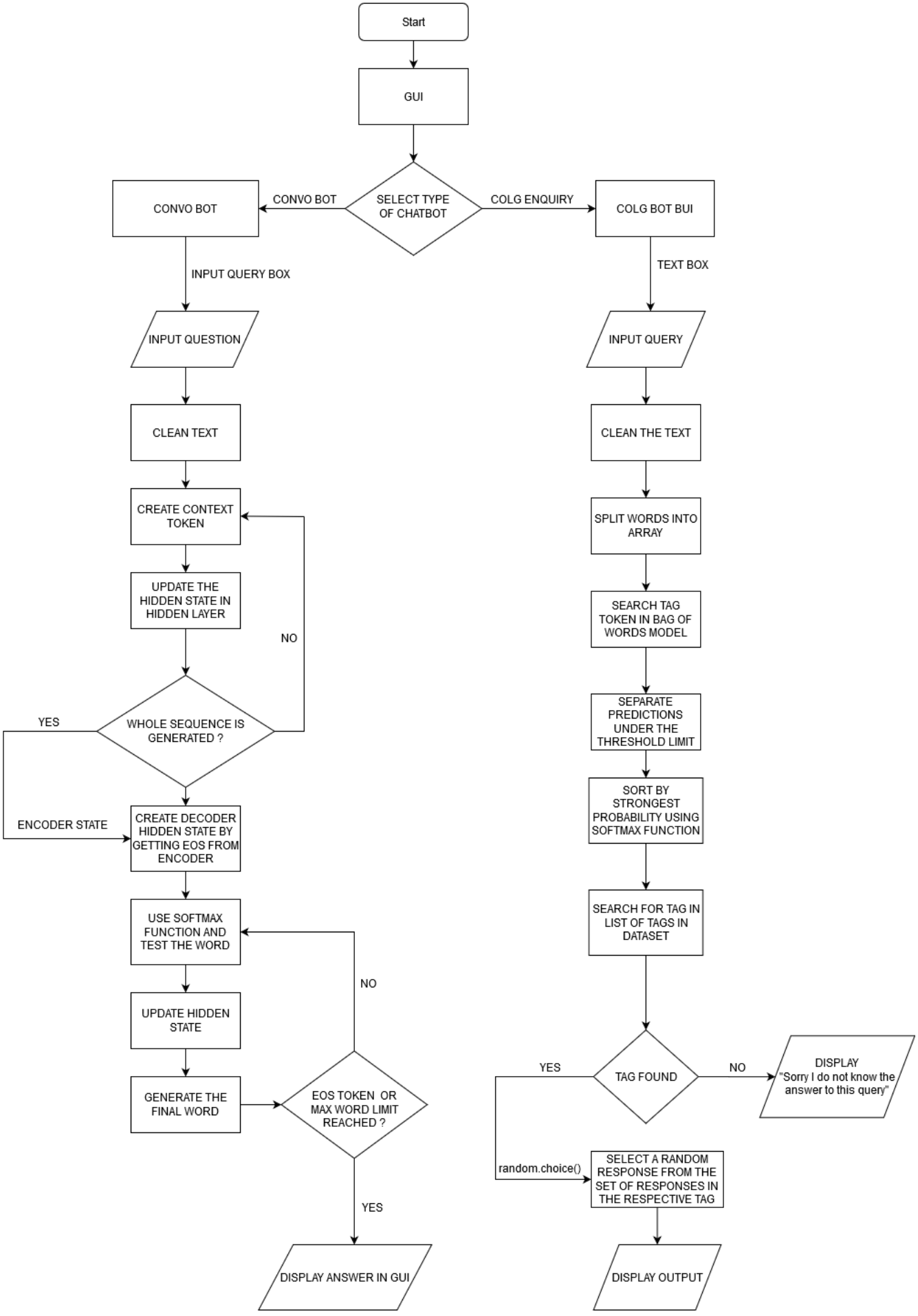
- OS: Windows 7 and above
- Python IDLE: 3.6 with packages:
    1. NLTK Version 1.4.3
    2. Keras Version 2.1.5
    3. Tensorflow Version 1.5.0
    4. JSONPickle Version 1.2.0
    5. Numpy Version 1.16.4
    6. Protobuf Buffer version 3.6.0

2.5 SCHEMATIC DIAGRAMS

2.5.1 BLOCK DIAGRAM DEPICTING THE PROCESS FLOW



2.5.2 FLOWCHART DEPICTING THE EXECUTION PROCESS

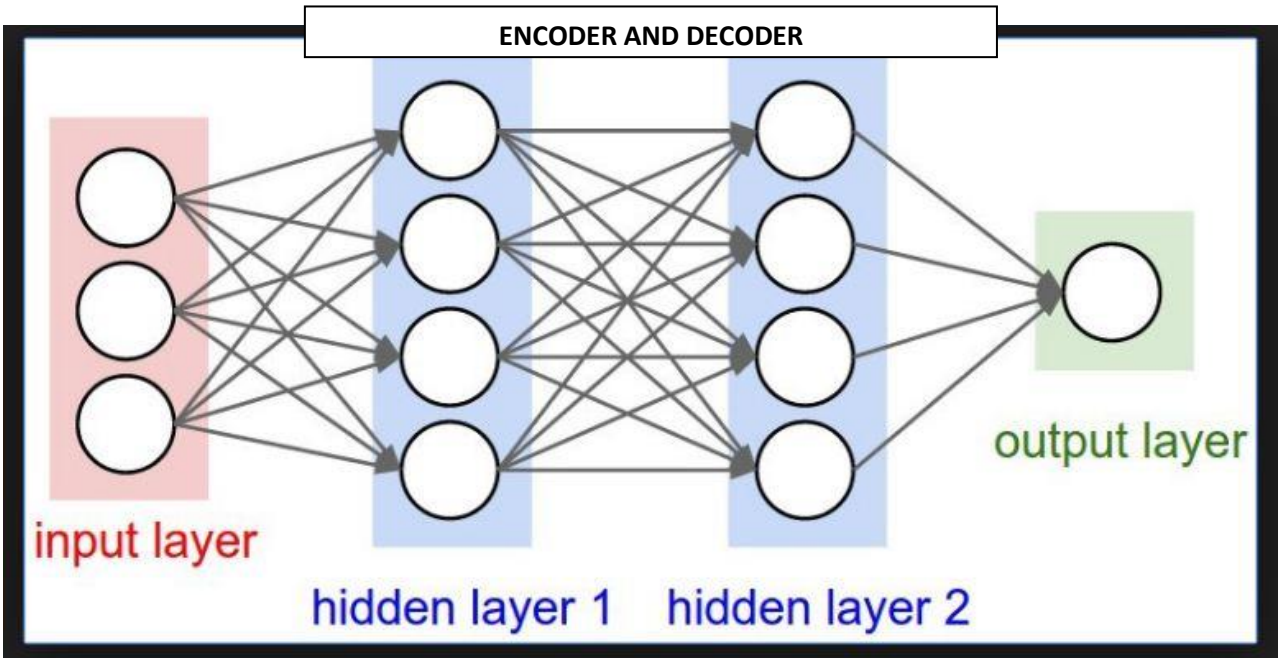


CHAPTER 3 IMPLEMENTATION

3.1 ARCHITECTURE

There are several layers that make up the architecture of the chatbot model. They are:

1.) INPUT LAYER
2.) ENCODE LAYER
3.) ATTENTION MECHANISM → REQUIRED NUMBER
4.) DECODE LAYER OF HIDDEN LAYERS
5.) OUTPUT LAYER



INPUT LAYER:

The input layer consists of input data which is inputted by the user. It consists of artificial neurons which are responsible for data transfer to the next layer. This is the first layer of the artificial neural network. It transfers the input data in the form of integers to the next hidden layer where the data is multiplied by the initially set low weights. It receives the data backpropagated by the output layer back to this layer where it again propagates it forward with updated weights.

In the input layer, the input nodes provide the data to the hidden layer from the input given by the user. The multi-layer perceptron will be used to construct the chatbot. This will consist of few hidden layers that will allow the chatbot to understand the data better and improve the accuracy of the model.

ENCODE LAYER:

The encoding and decoding of the architecture are a part of the RNN through which the data is translated, processed and given a meaning which can be later used to create vectors to predict answers based on user queries. This is the usual way through which the sequence to sequence predictions are generally made. This model used the seq2seq learning method using the mechanism of attention. Long Short-Term Memory networks is used to read the input data and understand the data. The data is then converted into vector form called prediction vector exactly like a bag of words model where every word is converted into an encoding integer. There can be any number of hidden layers in an encoder-decoder model based on the complexity of the model. The more complex the model, the more hidden layers are present and more time is required to process the data sequences.

The method of predicting an answer, understanding what comes next is called as language modelling. The new output is dependent on the output gained before. A set of LSTM cells are created where every LSTM cell accepts a single input. The input is a word in the encoded integer format. After the input is taken, then the information about that word is taken after which that LSTM cell is propagated forward.

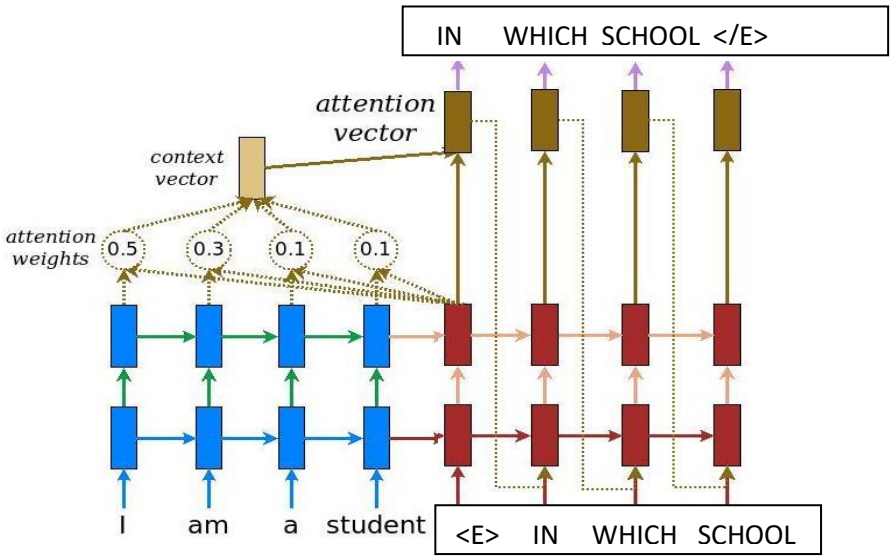
Every word here in the list created is of the order a_b, where a is the word and b is its order. The hidden state of the word which is h_b is then calculated using the formula :

$$h_t = f(W^{(hh)} h_{t-1} + W^{(hx)} x_t)$$

This will allow the encoder vector to get the hidden state, also called as the encoder state. The encoder state is the output that we get from the encoder that contains the information about the input processed in the encoder in accordance with a particular learning rate, dropout rate and the current weights applied. This encoder state then becomes the input of the next part of the neural network i.e. the decoding layer. The decoding layer receives the output of encoding layer (A vector called context/thought vector) and takes it as its first input along with few other parameters such as decoder cell matrix, batch size and RNN size.

ATTENTION MECHANISM:

The attention mechanism is a very essential component of the neural network. The attention mechanism was first created for NMT (Neural Machine Translation) by the use of sequential to sequential networks. These models consist of encoder and decoder layers which form the neural network and create the seq2seq model which predicts the data. There is a huge problem when the context vector is generated at the end of the processing of the encoding layer. The context vector is not designed to remember lengthy sequences. The context vector forgets the already processed input sequence when it processes very long sentences. So, the attention mechanism was introduced to combat this difficulty.



Attention vector is a vector that uses the softmax function which assists in determining the final output. The encoder looks at the entire input sentence and the creates the thought vector. But when the sentence is long, the attention vector makes the encoder ignore the rest of the lengthy sentence and instead, focus on the important keywords of the sentence and then generate an output for the current keyword it is processing. When the keyword has been processed, it proceeds to the next word and generates an output sequence. This attention mechanism is only used when processing longer sentences which can create difficulty in learning for the context vector.

There are certain steps by which the attention mechanism works:

1.) The encoder is fed the information containing the lengthy sequence.
2.) The words are converted into vectors and stored in a list.

3.) The list is propagated through the neural network.
4.) Softmax function is applied to the vectors along with the attention weights.
5.) A context vector is then generated and the output of the softmax function is added to it.
6.) The output generated by the decoder is the next correct word in the sentence.
7.) Same is repeated for the following words of the input sequence.

It is responsible for handling the relations between the input data and the output data. The attention component of the recurrent neural network will check all the words and assign values to them. It will analyze the input query asked by the user and map out the essential words from the question and give it higher values and weights so that they are of more significance and are given more priority. The problem is that often the seq2seq model is not able to process long data sentences accurately since the last part of the hidden layer of the encoding layer i.e. the encoding state is used as an input in the decoding layer. The attention mechanism prevents this and used all the hidden layers to process the data due to which the decoder is able to use data from all the hidden layers and has a greater choice in selecting the data to produce more accurate results.

There are many different types of attention layers that can be created as every attention layer is different and corresponds to the need of the particular neural network it is assigned to. The attention function, optimization and score function is different for every neural network and it is important to create a correct attention mechanism for the neural network to effectively predict the answers.

DECODE LAYER:

The decoding layer is a part of the RNN network that processes the data received in the form of vectors from the encoding layer and produces the output. The output of the encoding layer i.e. the encoder state is used as a main component in the decoding layer where using the prediction vector, the most likely possible prediction is made according to the rate of accuracy. The output is a fixed vector that shows how the input sentence is processed. The loss is obtained which along with the output which is then passed onto the final output layer.

The decoding layer gets the encoding state as its input which decodes a probability of what word is best to be predicted using the softmax function.

Softmax function gives back a probability function of each output word that can be chosen. The result that is achieved in the softmax function is rounded off a value between 0 and 1. Whichever value is the greatest, it will be chosen and that output is predicted.

Along with the softmax function, the bias, the weights and the hidden encoder state are all taken as inputs and then finally the prediction is made based on the previous outputs. The final output is generated and then sent to the output layer for displaying on the screen/GUI.

OUTPUT LAYER:

The output layer displays the output along with the loss generated. This loss generated is a gradient of the loss function generated every epoch. This error/loss is backpropagated back through the hidden layers onto the input layer. The initial weights are updated in accordance with the loss and the whole sequence is repeated again with the new updated weights. This is called as an epoch. These epochs are done until the loss reduces to a very low amount and the accuracy increases to an optimum of 95 percent.

3.2 MODULES DESCRIPTION

There are 3 modules in the project and each have a certain dependency on the other modules. They are:

1.) Building, Importing and preprocessing the dataset
2.) Building the deep learning RNN brain model, the retrieval model and creating the GUI
3.) Training both the models to achieve 100 percent accuracy

BUILDING, IMPORTING AND PRE-PROCESSING THE DATASET:

Preprocessing the data mean bringing the data into a form which can be predicted, analyzed and understood. The text is processed in different ways in the model.

Lemmatization: The process of combining similar words. Every word is mapped to a root word and put into the same category for easy retrieval

Stop Words: Words like the articles and prepositions are removed from the text as they are information with low importance.

Higher form of English such as the words I'd, it'll, it's are reduced to their normal base form, like I would, it will and it is, to optimize the chatbot model.

The data is cleaned and converted into a bag of words model where every word is given a unique integer id which is used later for data prediction. The questions and answers are separated and mapped to each other and the SOS and EOS tokens are added to the sentences to determine the start and end of the sentences.

However, the processing for both the models is different. For generative models, data has to be cleaned effectively whereas for the retrieval model need not be trained to that extent as the answers are already predefined.

GENERATIVE MODEL:

The preprocessing starts with cleaning the data and sorting them in encoded list form. The steps are:

- The 2 datasets consisting of lines and conversations are imported using "With open" command.
- A dictionary is created that maps every line with its unique ID in the data set.
- A class called Vocabulary is created to manage all the words in the data sequence.

- The number of times a word occurs in a sequence is noted down and a limit called threshold limit is set on the minimum times a word should occur in the dataset. This helps the neural network to ignore the words that occur less than the threshold limit showing the words are not important and can be removed from preprocessing.
- 4 special tokens are created that perform various tasks on the dataset:
    1. PAD – PAD token is created to ensure that the length of questions and the answers remain the same.

       Eg: Question: What is your name? – length - 4

           Answer: It is Abdullah <PAD> - length – 4
    2. SOS – The SOS token indicates the start of a sentence so that the neural network knows a new sentence has begun in the dataset.
    3. EOS – The EOS token indicates the end of a sentence signaling the Neural Network to look for the next sentence by finding the next SOS token.
    4. OUT – If a word is not present among the vocabulary in the dataset, the OUT token is used in place of that word.
- Every sentence in the data set is cleaned by converting higher form of English to its lower form.
- The words are then converted to integer form and stored in a list.
- Questions and answers are separated in the dataset and stored in separate lists.

RETRIEVAL MODEL:

There is very less cleaning required in a retrieval dataset as the answers are already predefined. The steps to preprocess the data in the retrieval model chatbot are:

- Import various python libraries required like numpy, sequential API from keras.models, SGD from keras.optimizers and Dense, Activation and dropout from Keras.layers .
- Lists called words, classes and documents are created for data processing.
- Every word is tokenized using the package WordNetLemmatizer which returns a group of words of similar type together. They are processed as of one type.
- Tags are created and stored in the classes list. These tags are used to identify a keyword in the sentence and if same keyword is found among the tags in the dataset, that set of answers is fetched and an answer selected using the soft max function.
- Every word is then lemmatized and a combination of the intents/context and the patterns are created and stored in the “documents” list.
- The preprocessed data is saved in pickle files using. pkl extension.

BUILDING THE DEEP LEARNING RNN BRAIN MODEL AND THE RETRIEVAL MODEL:

1.) GENERATIVE MODEL:

The model, which is the recurrent neural network is built using the LSTM learning method to get the input data, process it using matrixes and vectors, and get a final output of the predicted answers.

The neural network consists of an input layer, several hidden layers and an output layer. The inputs are passed from the input layer to the output layer via the hidden layers where the weights are multiplied with the inputs. The result obtained is backpropagated the entire network, the weights are updated again and the next epoch begins. This is done until the accuracy of the predictions is well over 95 percent. If the number of hidden layers is increased, the chatbot has a chance of understanding the input data better.

The steps required to create a recurrent neural network using Long Short-Term Memory (LSTM) cells are:

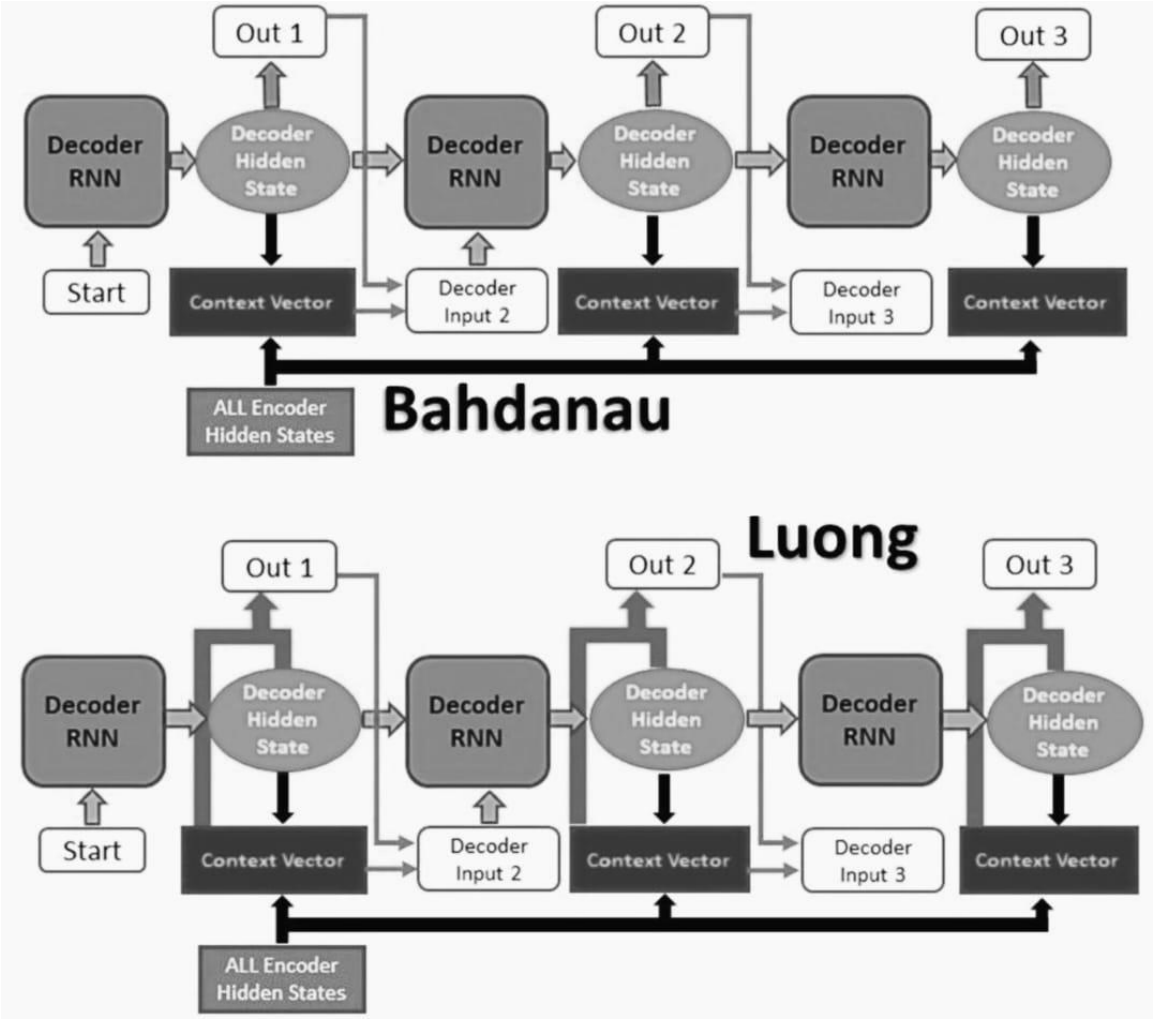
- Firstly, A parent class called ChatbotModel is created which will consist of all the sub classes required to construct the model. This class is heavily dependent on tensorflow.
- The seq2seq chatbot model is created using various parameters such as the hyperparameters class, the input and the output vocabulary and the directory where the model is stored and the checkpoint file is saved with the .ckpt file extension.
- The tensorflow graph has to be reset to a default value after which it can be loaded with the input tensors created in the neural network.
- Placeholders are created for the chatbot model which are two dimensional. These placeholders are used later in the encoder-decoder model.
- The learning rate, keep probability and the length of the sequence is defined and stored in the placeholders.
- Create a session which will save the output to a checkpoint file after training and which will also load the checkpoint for execution of the chatbot.
- Two functions load and save are created for loading and saving the checkpoint file respectively.

- A function for training the model using a batch of conversations is defined. It will train the model on a single batch and give back the output. This function has several important arguments:
    1. Input: Input has to be provided in the form of a matrix. The shape can be two dimensional or three dimensional. We have chosen a two-dimensional input tensor so as to simplify the taking of chatbot inputs.
    2. Target: targets are the answers which are predicted. These are also in the form of a tensor with a particular shape. They are stored in a matrix.
    3. Length of the input sequence: The length of the input sequence is crucial to determine a good batch size during the training of the chatbot model. The sequence lengths can be varied accordingly.
    4. Length of the output sequence: This vector contains the length of the output answers which are predicted by the chatbot model. It is very essential that the length of the input sequence and the length of the output sequence remains the same so as to simplify the learning.
    5. Learning rate: It is used to update the weights at every epoch.
    6. Drop out rate: The dropout rate is a percentage of the neurons in the neural network used to train the neurons. Only when testing the model are all the neurons used. When training the model, usually 50 percent of the neurons are used to training as using all of the neurons will much loss in training and have a high probability to generate incorrect answers.
- The next step is to create the encoder which will perform the first phase of processing the encoded data in integer form. It has an argument called encoder embedded input.
  Here, instead of using a single RNN cell for both forward and backward propagation, we use a bi-directional encoder with a few RNN cells used to read and understand the forward and the backward sequences separately. Then all the cells are joined together and then sent to the decoder after the construction of the thought vector.
- Before creating the decoder, we need to apply the attention mechanism on the thought vector after which only it will be sent to the decoder. A decoder cell is created and the attention mechanism is applied to it. The decoder cell takes several crucial arguments:
    1. The output of the encoder: This is the output obtained after the data has been propagated through the encoder. This is a tensor of a particular shape containing the input data passed through the encoder at every timestep.

2. The encoder state: the encoder state is the combined encoder state of all the RNN cells after their encoding has been completed. This encoder state which is the output of the encoder becomes the initial input to the decoder layer.
3. Group Length/ Batch Size: The number of the conversations taken at a time for reading via the encoder. The batch size should be average and neither too low nor high for effective understanding of the conversations.
4. The attention mechanism is applied to the decoder cell. There are two types of attention:
    a) Luong Attention
    b) Bahdanau Attention

Loung Attention: Luong attention uses a single layer to obtain the hidden state of both the encoder and the decoder.

Bahdanau Attention: Bahdanau attention takes different RNN cells to obtain the forward and the backward hidden states from which the attention score function is calculated. The attention score function is then used to construct the attention context function from which the context vector is derived.



- After the decoder cell has been built, create the decoder layer which uses the decoder cell and encoder state as its arguments in addition to another argument called decoder embedded matrix which consists of the embeddings of the decoder layer.
- Create the decoder layer for training the data using the two functions using Tensorflow core libraries - tf.contrib.seq2seq.BasicDecoder and tf.contrib.seq2seq.dynamic_decode.
- When the decoder layer is created, an optimizer should be set up to optimize the neural network and achieve faster iterations/epochs. This is done through using the tf.train.GradientDescentOptimizer function.
- Create an RNN cell to be used in the neural network and preprocess the predictable answers by adding the <SOS> token to all the answers in the group so that the neural networks know where a new sentence starts.
- Create and initialize the session and proceed towards training the data.

2.) RETRIEVAL MODEL:

The retrieval chatbot model is comparatively easier to build as it does not use the approach of building and training the brain using deep NLP. The retrieval is made on the principle of directed flows. The model searches the input data sequence for the context words, also called as tag, which will allow the model to determine what type of input is being asked by the user. It will then fetch the answer from the group of answers mentioned under that tag from the already predefined dataset. Since the chatbot gets the output from the list of predefined responses, there is no chance of the chatbot showing grammatical errors when compared to the generative chatbot which shows a tendency to sometimes generate answers with grammatical mistakes. The retrieval model uses the Keras and NLTK libraries of python to perform the data preprocessing and answer generation.

There are several steps when constructing the neural network using the Keras Deep Learning library of Python 3.5 though the steps are fewer when compared to the generative chatbot model. They are:

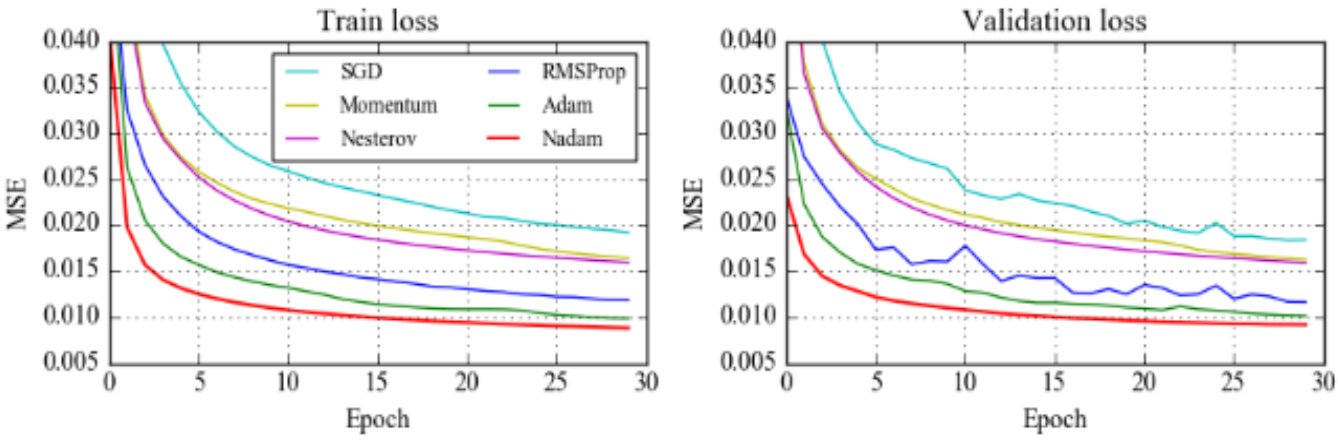
- Create a deep learning model using the Sequential API of the Keras library. The sequential API will allow to create 2-3-layer neural network model.
- The model also uses few functions from keras library which are significant to construct the network:
    1.) The Dense function is used from keras.layers. The Dense implements the output: OP = (dot (input, Kernel) + bias) where kernel is the matrix constructed by the weights and bias is a vector.

2.) Activation is an activation function used in the dense function.

3.) SGD is imported from the keras.optimizers function. SGD is Stochastic Gradient Descent optimizer for optimizing the neural network. SGD is used to overcome the exploding gradient and the vanishing gradient problems usually accompanied in a neural network. The loss gradient becomes dangerously close to zero or more than one respectively which is solved by SGD optimizer. It replaces the gradient created by the dataset by an approximate gradient estimate which is calculated by a random group of data in the dataset.

4.) Dropout is taken from keras.layers to withhold a certain percentage of neurons for training the neural network.

- Create a model of 3 layers. The first layer contains 128 neurons, the 2nd one contains half the neurons and the final layer which is the output layer contains the number of neurons which are equal to the intents which are equal to the outputs which are created using the function called Soft Max function.
- The soft max function is used in the output final layer which is used to predict the probability of the output to be chosen. The outputs are rounded off to a value between 0 and 1 and the value which is the greatest will be chosen among the answer set to be displayed as output.
- The model is then compiled by using SGD optimizer along with nesterov accelerated gradient used to conjugation to achieve better results.



As it is represented in the above diagram, the training loss and the validation loss are depicted when used various optimizers and SGD optimizer takes the value further from zero and hence, trains the data better than the optimizers shown in the diagram.

CREATING GUI FOR RETRIEVAL AND CHAT MODEL FOR GENERATIVE MODELS:

A GUI is very essential to display the output in a user-friendly manner. A GUI can be created in python using the TKinter library used to create windows.

RETRIEVAL MODEL:

For the retrieval model, since the model is already trained, only the checkpoint file with the extension ".ckpt" has to be loaded and the model to be executed. The faster way is to execute the program using the operating system rather than an interpreter as OS execution is much faster.

The method to create a GUI for the retrieval chatbot involves a series of steps:

- Import the tkinter library.
- Import "os" and "sys" from the predefined python libraries.
- Define two functions and insert the chatbot execution command in the function using os.system.
- Choose a window size for the GUI by using the geometry function of the tkinter library.
- Create a frame in which the options and buttons are put in.
- Create two buttons using the tkinter.button option and select the function to be loaded when the respective buttons are pressed. Give the buttons a background colour, font colour, the height, the width and font text.
- Arrange the buttons vertically in the center of the window and exit the GUI by using function.mainloop().

TRAINING THE MODELS:

GENERATIVE MODEL:

Both the models have to be trained with the data given in the dataset. For the generative chatbot, the dataset has to be very large of at least one million conversations to effectively train the chatbot. The dataset is constantly updated with conversations between people which the model will use to predict answers for input put forward by the user and trained upon previous conversations, based on which the responses to the user are generated. The encoder-decoder component of the LSTM works to analyze and get the prediction vector required for data prediction. The encoder processes the vector of input words, by checking the importance of every word likely to be in the answer and passes this information onto the decoder as encoder state. The decoder takes the output of encoder as the input and

makes the prediction using the test prediction vector. The output is then passed onto the output layer to be displayed. This epoch is done a number of times to increase and accuracy and reduce the training error loss.

The procedure for training the data set involves a number of different operations to be performed:

- Firstly, A hyperparameters class is created where all the important hyperparameters are specified. Each hyperparameter has its own significance:
    1. RNN Cell type – The type of the learning which is being used by the neural network, like LSTM.
    2. Bi-Directional Encoder: A bi directional encoder used 2 LSTM cells to process and analyze the data. It is different from a single LSTM cell commonly used to forward and backpropagate the encoded data in integer form. Here since 2 different cells are used, the data is transmitted from one layer to another very fast and the training time reduces.
    3. Number of layers in encoder: The quantity of RNN cells used in the encoder layer.
    4. Number of layers in decoder: The quantity of RNN cells used in decoder layer.
    5. Size of the embeddings matrix in encoder: The size of the matrix in encoder which will create a vector of particular shape.
    6. Size of the embeddings matrix in decoder: The size of the matrix in decoder which will create a vector of particular shape.
    7. Type of the attention mechanism: BAHDANAU OR LUONG.
    8. Width of the beam: The quantity of beams which are created by the function called Beam Search Decoder.
    9. To enable sampling or not: According to the value set, Different type of decoder such as greedy or sampling decoder is utilized for training. Both the decoders have different methods of learning and each should be tested by trial and error method to determine which decoder learns the best way.
    10. The maximum value of Gradient: The max value of gradient which is to be clipped.
- Read the hyperparameters class in a new program and import the dataset along with it.
- The dataset has to fed into the validation and the training set. The training set and the validation sets are 2 different sets used in different contexts. The training set will be used to train the data with the hyperparameters applied along with a keep probability rate which has the default value of 0.5. 50% of the total neurons will be used during the training. In case of the testing set, it is only used to check the predictive power of the chatbot and hence, all the neurons are used to test the neural network. So, the keep probability is set to 1.

- Create the sequential-to-sequential model class by combining many functions created beforehand like the encoder class, decoder class, validation and training class.
- Resume the training if already the checkpoint file exists or begin new training if there is no existing checkpoint file.
- Note down the time taken for training one batch of data and for one epoch. This can be done using the time library available in python.
- Calculate the total loss obtained when training a single batch. The error must be close to zero and the accuracy should be close to 1.
- A minimum of 100-200 Epochs have to be done to train the data effectively with a batch size of 32 or 64 as the value accepted for the batch size should be exponential powers of 2.
- Validation set must be run to test the predictive power of the chatbot and the parameters have to be tweaked repeatedly and validation set run again until the answers being generated are logically coherent.
- The learning rate has to be applied to the neural network which will control the way the weights are being adjusted after every iteration in accordance with the loss gradient being generated at every iteration. At the beginning, the learning rate should be a small positive value close to zero. The value should be modified according to the answers being generated and can be fixed when decent answers are being generated.
- Create a checkpoint where the trained data will be stored after every batch is trained until all the epochs are completed. The training can be stopped if desired and resumed again later as effective training requires 24-48 hours which depends on the system processor speed.
- The training loss as well as the validation loss is improved at every epoch until the accuracy is close to 1. If the accuracy is reached before all the epochs, the training stops itself mid-way.
- Save the final trained data in the checkpoint file.
- Create another python file called Training Stats which does a deeper processing during the training:
    1. Compare Train Loss: The function compares the new training loss generated after an epoch with the training loss generated at the previous epoch. If the new value is better than the previous value, it replaces the older value and if not, then rejects the new value and moves to the next iteration.
    2. Decay learning rate: The learning rate is changing at every epoch and the weights are adjusted accordingly. If the learning rate becomes very low and falls below the minimum value, the error generated will become greater with each epoch. To overcome

this problem, the decay learning rate is used which will set a minimum value for the learning rate below which it would not go.

3. Save: The save function saves the Training Stats to the system.
4. Load: The load function loads the Training Stats from the system during execution.
5. Compare Metric Value: The new value obtained will be compared with the old metric value. Whichever value is best will be selected among the two. There are 2 types of metrics:
    a) Lower is better: The loss generated
    b) Higher is better: The accuracy of the training

   The type of metric is selected and the values are compared. Then the next type is selected and values are compared.

RETRIEVAL MODEL:

For the retrieval chatbot, the dataset has to contain the dataset with the set of questions and answers which the chatbot will directly get when the particular question is asked. The chatbot is trained after importing the dataset to improve the efficiency of selecting correct answers. The chatbot will check the input query for the tag words through which it will find the answer set in the dataset and choose the best ranking answer among the given answers and display the output.

For the retrieval model, the training is very simple and uses the Keras deep learning library to train the data:

- Create an empty array for the output and a list for training data.
- Create a bag of words model and a training set in the form of a array.
- The bag of words model will contain the words which are lemmatized before putting them into the array. Lemmatization is the method of grouping together words which have a similar meaning. For example, the words good, better, nice can be grouped under one category since they map to the same root word: GOOD.
- Lemmatization is done and the bag of words model is created and each word is given is value of 0 or 1. If the word that is being searched is found in the bag of words model under the pattern tag, that word is given an increment of 1, otherwise it remains 0.
- Random.shuffle() function is used to shuffle the list of words in the bag of words model and generate a new order of the list.
- The list is then turned into a numpy array using the pre-defined numpy python library.

- Two lists called test and training are created. The training list is used during training and the test list is used only when testing the predictive power of the data.
- The model is compiled with Stochastic Gradient Descent to get the training done faster. The Nesterov Accelerated Gradient is used along with SGD for good results.
- The learning rate, decay learning rate is set and Nesterov is set to true.
- The model is compiled using model.compile function with the parameters:
    1. Loss: The loss function used here is the 'Categorical Cross Entropy" loss function. Categorical cross entropy is the loss function that is used for the categorization of a single label. The labels are always mutually exclusive and every instance of a label is handled by only one category. This function is used when only one option is to be used among many. If the softmax activation function is being used, then only the categorical cross entropy loss function can be applied to the model.
    2. Optimizer: The optimizer is Stochastic Gradient Descent which helps to get faster iterations. It will replace the real gradient created by the dataset with an approximate value of the gradient which is calculated by using small subsets/batches of data from the data set.
    3. Metrics: The metrics used here is "Accuracy". The metrics are of 2 types: Accuracy and Loss. Whichever is suitable to the user, that metric can be used to analyze how well the training has been done. If accuracy metric is chosen, it's value should be close to 1 and if loss metric is chosen, it's value should be close to 0.
- The model is compiled and the training data is fit into the model using fit() function of the SciPy python library. For supervised learning, there are 2 arguments and for unsupervised learning, there is only one argument.
- The epochs and batch size is set accordingly and the model is saved using the save() function.
- The model can be trained by executing the train_chatbot.py file. The training in this retrieval chatbot is done very quickly since the data set is very small and does not require thorough cleaning as is in the case of a generative chatbot model.

3.3 INPUT AND OUTPUT ANALYSIS

INPUT ANALYSIS:

GENERATIVE CHATBOT:

The input for the generative model can be any sentence. The model has been trained over one million conversations and has the capability to construct answers even when there is no similar question present in the database. The model can only predict accurate answers if the training has been done correctly otherwise there would be high probability of the chatbot generating incorrect answers. The chatbot though is only capable of providing the answers to the questions the user asks. The chatbot does not have the ability to ask the user a question, i.e. simultaneous conversation between the two. It works only on the principle of question and answer.

The input is constructed using a context/thought vector that generates the data based upon the previous conversations. The data can be input in any form or structure. Any question when inputted, the question is cleaned by the “clean_text” function in the chatbot program and removes unnecessary symbols and stop words. That cleaned question is sent to the encoder for further processing to generate an answer.

The inputs can be of any form and any structure. However, if a question pertaining to a specific topic is asked, the chatbot will provide incorrect answers as the generative chatbot supports only general everyday conversations and not discrete conversations. For instance, if questions like,” What is the width of a human brain?”, “How many people live in Masab Tank?”, “Does pluto have humans living on it?” cannot be answered because they are discrete topics and not something people converse about every day. Such specific enquiries should be inserted in the retrieval chatbot dataset as that chatbot can answer specific questions easily if they are pre-defined.

RETRIEVAL CHATBOT:

For the retrieval model, the inputs have to be provided in a certain way. The inputs for the college queries have to be asked in the same way as present in the dataset or the chatbot won’t be able to get an answer if the input differs from the questions present in the dataset. The tag words are essential to be present in the query as the absence of them can mislead the chatbot to predict incorrect answers.

For a retrieval chatbot, the questions have to be specific enough with their answers present in the database:

- The greeting questions, “Hi”, “hello”, “Hi There” are asked, they come under a single category called greetings. Their intents are same and hence the answer to these questions will be selected from under the same tag.
- A separate intent tag is created for blank space, so a blank space can also be inputted and the answer to it will be fetched from a distinct pattern tag.
- The input can either be blank or a specific question or a similar question with the same intent.
- Any question, if not present in the dataset and does not come under any tag, will have a common answer.

OUTPUT ANALYSIS:

GENERATIVE CHATBOT:

For the generative model with which the user can have any conversation, the output is predicted using the test prediction function with which a learning rate and a keep probability rate is associated. The words present in the integer form having higher weights are given more priority and selected for prediction. The output is more accurate when the training loss error is less and accuracy is more. The prediction is made on the most likely word that comes after the previous word and if it is correct, the chatbot predicts the next word and checks the loss if it is being reduced. If not, then it will move on to the next word and make the prediction again.

The output varies every time a question is asked. For example, if a question,” what is going on in your life presently?” is asked, the answer to that question can be, “I am fine”, “It is going good”, “I feel excellent during these days”, “I am sick since a long time” etc. because there are many ways in which that question can be answered. The dataset contains many conversations of the same type with different responses and hence, the answer varies even if the same question is asked a few times.

The inputs can be of any form and any structure. However, if a question pertaining to a specific topic is asked, the chatbot will provide incorrect answers as the generative chatbot supports only general everyday conversations and not discrete conversations. For instance, if questions like,” What is the width of a human brain?”, “How many people live in Kukatpally area?”, “Are there glaciers on Pluto?” cannot be answered because they are discrete topics and not something people converse about every

day. Such specific enquiries should be inserted in the retrieval chatbot dataset as that chatbot can answer specific questions easily if they are pre-defined.

If another dataset is used, the answers generated will be entirely different. The dataset taken here is the Cornell Movie Corpus that contains over 6 lakh conversations between people from different movies. There are other datasets like Twitter Corpus, Reddit Corpus, etc. which if used, will generate different answers.

RETRIEVAL CHATBOT:

For the retrieval model for college enquiries, the output is directly taken from the dataset and displayed as it was. There are no grammatical errors since the answers are already predefined in the dataset and show 100 percent accuracy. The outputs cannot differ from the answers in dataset because the model does not predict answers on its own but rather chooses the best answers from the answer set and displays it on the GUI screen.

For the retrieval chatbot, the answers are specific as they are defined in the dataset already:

- For the questions like “Hello”, “Hi”, “Hello my friend”, the answer will be selected from a single pattern tag. The answers from that tag will be selected at random. The answers can be the respective greeting back like “Oh hello”, “Hey”, etc.
- If a blank space is given as input, the answer will be chosen from the blank pattern tag category where the answer will be chosen the set of responses like “Sorry, can you say that again?”, “I didn’t get you” etc.
- If a question is asked which is not present in the dataset or under any tag, a specified answer is given to all questions of this type which will respond, “I do not know the answer to that query”. The if-else condition is used in the python program where if a tag occurs, then to search the answer from the list of tags and if tag not found, then print the aforementioned statement.
- All the questions should be simple and not contain any form of commas or quotations since the chatbot is only capable of cleaning the query of question marks and exclamation marks. This is because the chatbot can train faster if the question and answer formats remain simple. If lot of cleaning is done, the chatbot will take more time to train and since the datasets are small, they can be created with simple sentences and less symbols.

CHAPTER 4 CONCLUSION AND FUTURE ENHANCEMENT

4.1 CONCLUSION

The chatbot thus created will solve the problem of the students who live far away to enquire about various details like upcoming events, faculty information, etc. eliminating the need to go to the college specifically to enquire.

The users also will have the opportunity to chat with the other chatbot with any conversation the user would like to have as the chatbot talks very similar to a human.

4.2 FUTURE ENHANCEMENTS

There are a number of future enhancements that can be made on the present project model.

1.) A special dataset can be created with at least 1 million conversations of college related enquiries and that dataset be trained with the present neural network model creating an intelligent chatbot which predicts the answers on its own rather than fetching the answers from the database.
2.) The chatbot can be converted into a windows program and uploaded online so that it can be used on any system without having the need to install python language.
3.) More datasets can be added to the present generative chatbot model to make it more efficient at human conversations.
4.) The training time for the generative can be greatly reduced by using the updated version of the python packages.
5.) It can be further updated and edited to include the compatibility of use on Macintosh Systems.

REFERENCES

1.) Ilya Sutskever, Oriol Vinyals, Quoc V. Le, 2014, Sequence to Sequence Learning with Neural Networks, Dec 2014, published in ArXiv, by Cornell University.

2.) Oriol Vinyals, Quoc Le, 2015, A Neural Conversational Model, published in ArXiv, by Cornell University.

3.) Dzmitry Bahdanau, Kyunghyun Cho, Yoshua Bengio, 2014, Neural Machine Translation by Jointly Learning to Align and Translate, published in ArXiv, by Cornell University.

4.) Minh-Thang Luong, Hieu Pham, Christopher D. Manning, 2015, Effective Approaches To Attention-Based Neural Machine Translation, publishes in ArXiv, by Cornell University.

5.) Sumit Raj, Dec 2018, Building Chatbots with Python: Using Natural Language Processing and Machine Learning, Book published by Apress,2018.

6.) Jurgita Kapočiūtė-Dzikienė, 2020, A Domain-Specific Generative Chatbot Trained from Little Data, published in *Applied Sciences* (ISSN 2076-3417; CODEN: ASPCC7), by MDPI.

7.) Gundapu Nitish Kumar, Devavarapu Sreenivasarao, Shaik Khasim Saheb, 2019, Chatbot and its Practical Applications in the Materialistic World, published in International Journal of Recent Technology and Engineering (IJRTE)ISSN: 2277-3878, Volume-8 Issue-4

8.) Minghui Qiu, Feng-Lin Li, Siyu Wang, Xing Gao, Yan Chen, Weipeng Zhao, Haiqing Chen, Jun Huang, Wei Chu,2017, AliMe Chat: A Sequence to Sequence and Rerank based Chatbot Engine, published in proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)**.**

9.) Anik Das, Rashid Khan, 2017, Build Better Chatbots: A Complete Guide to Getting Started with Chatbots, published by Apress,2017.

10.) Anik Das, Rashid Khan, 2017, Build Better Chatbots: A Complete Guide to Getting Started with Chatbots, published by Apress,2017

11.) Manisha Biswas, 2018, Beginning AI Bot Frameworks: Getting Started with Bot Development, published by Apress,2018.

12.) Mohammad Nuruzzaman, Omar Khadeer Hussain, 2018, A Survey on Chatbot Implementation in Customer Service Industry through Deep Neural Networks, published by IEEE in 2018 IEEE 15th International Conference on e-Business Engineering (ICEBE), 18366961, 10.1109/ICEBE.2018.00019.

13.) Kyunghyun Cho, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, Yoshua Bengio, 2014, Learning phrase representations using RNN encoder-decoder for statistical machine translation, published in ArXiv, by Cornell University.

14.) Shaojie Jiang, Maarten de Rijke, 2018, Why are sequence-to-sequence Models so dull? Understanding the Low-Diversity problem of Chatbots, publishes in ArXiv, by Cornell University.

15.) Yuening Jia, Attention Mechanism in Machien Translation, 2019, published in Journal of Physics Conference series, 1314:012186, DOI: 10.1088/1742-6596/1314/1/012186.

16.) Kumar Shridhar, May 2017, image on page 9, source: https://medium.com/botsupply/generative-model-chatbots-e422ab08461e

17.) May 2019, Hidden layers in Neural Networks, image on page 23, source: https://www.i2tutorials.com/technology/hidden-layers-in-neural-networks/

18.) Synced, September 2017, A brief overview of attention mechanism, image on page 25, source: https://medium.com/syncedreview/a-brief-overview-of-attention-mechanism-13c578ba9129

19.) Gabriel Loye, September 2019, Attention Mechanism, Image on page 32, source: https://blog.floydhub.com/attention-mechanism/

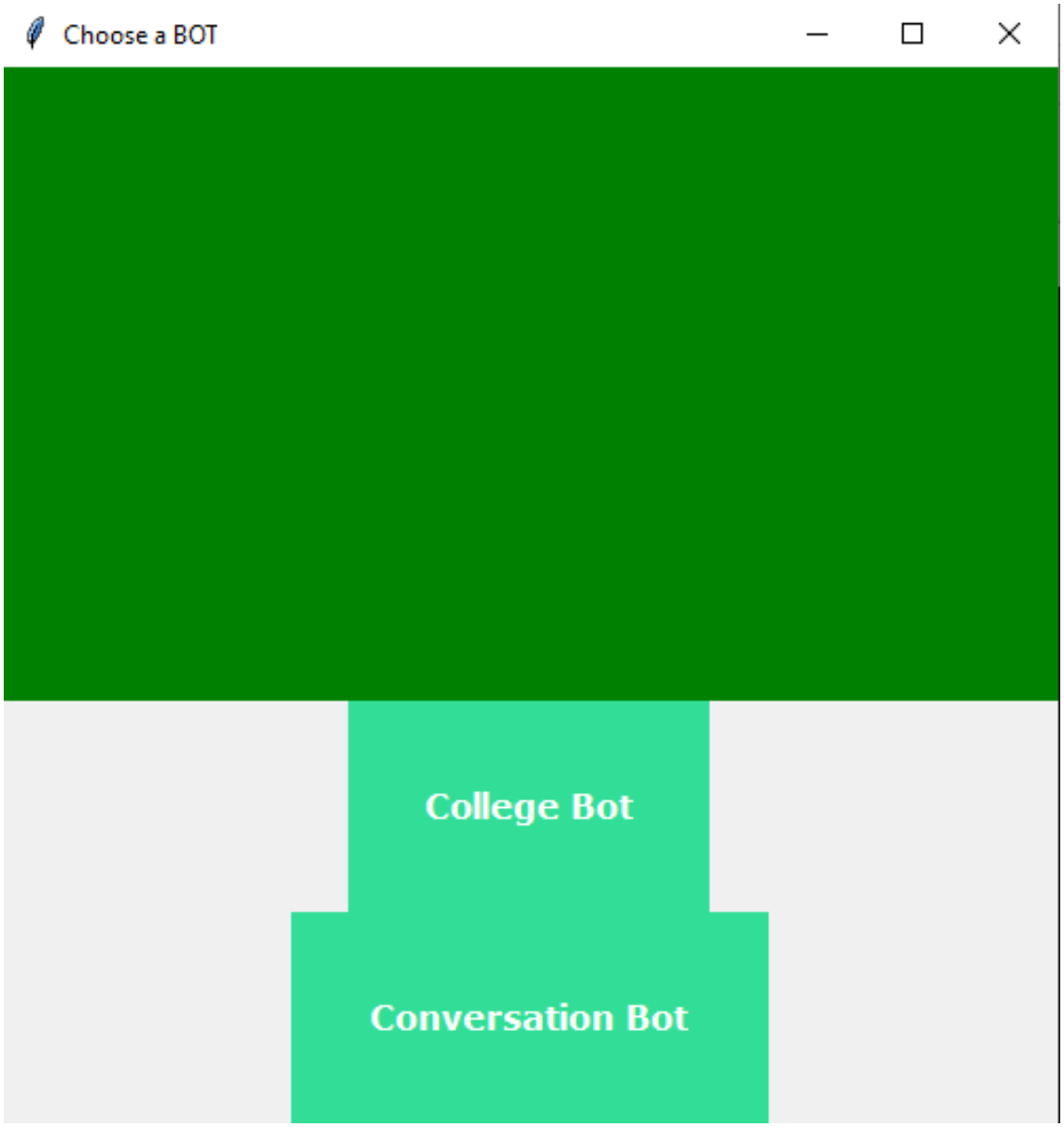
20.) Vitaly Bushaev, October 2018, Adam – latest trends in deep learning optimization, image on page 34, source: https://towardsdatascience.com/adam-latest-trends-in-deep-learning-optimization-6be9a291375c

APPENDICES

I PLAGIARISM REPORT

II SCREENSHOTS

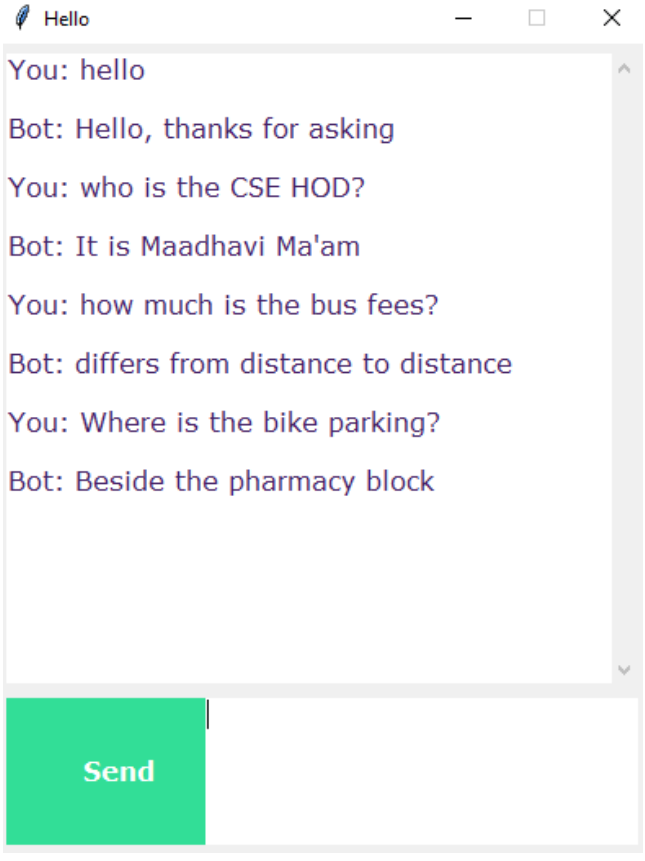
CHOOSE A BOT - GUI – OUTPUT



CONVERSATION BOT OUTPUT – GENERATIVE MODEL

```
You: HELLO
ChatBot: how is it going?

You: its going good. How are you?
ChatBot: I am fine.

You: do you have a name?
ChatBot: no I am just.

You: where do you live?
ChatBot: I told you all night.

You: do you like talking to me?
ChatBot: yeah.
```

COLLEGE ENQUIRY BOT OUTPUT – RETRIEVAL MODEL



GENERATIVE CHATBOT DATASET – CORNELL MOVIE CORPUS

```
L1044 +++$+++ u2 +++$+++ m0 +++$+++ CAMERON +++$+++ They do to!
L985 +++$+++ u0 +++$+++ m0 +++$+++ BIANCA +++$+++ I hope so.
L984 +++$+++ u2 +++$+++ m0 +++$+++ CAMERON +++$+++ She okay?
L925 +++$+++ u0 +++$+++ m0 +++$+++ BIANCA +++$+++ Let's go.
L924 +++$+++ u2 +++$+++ m0 +++$+++ CAMERON +++$+++ Wow
L872 +++$+++ u0 +++$+++ m0 +++$+++ BIANCA +++$+++ Okay -- you're gonna need to learn how to lie.
L871 +++$+++ u2 +++$+++ m0 +++$+++ CAMERON +++$+++ No
L870 +++$+++ u0 +++$+++ m0 +++$+++ BIANCA +++$+++ I'm kidding.  You know how sometimes you just become this "persona"?  And
L869 +++$+++ u0 +++$+++ m0 +++$+++ BIANCA +++$+++ Like my fear of wearing pastels?
L868 +++$+++ u2 +++$+++ m0 +++$+++ CAMERON +++$+++ The "real you".
L867 +++$+++ u0 +++$+++ m0 +++$+++ BIANCA +++$+++ What good stuff?
L866 +++$+++ u2 +++$+++ m0 +++$+++ CAMERON +++$+++ I figured you'd get to the good stuff eventually.
L865 +++$+++ u2 +++$+++ m0 +++$+++ CAMERON +++$+++ Thank God!  If I had to hear one more story about your coiffure...
L864 +++$+++ u0 +++$+++ m0 +++$+++ BIANCA +++$+++ Me.  This endless ...blonde babble. I'm like, boring myself.
L863 +++$+++ u2 +++$+++ m0 +++$+++ CAMERON +++$+++ What crap?
L862 +++$+++ u0 +++$+++ m0 +++$+++ BIANCA +++$+++ do you listen to this crap?
L861 +++$+++ u2 +++$+++ m0 +++$+++ CAMERON +++$+++ No...
L860 +++$+++ u0 +++$+++ m0 +++$+++ BIANCA +++$+++ Then Guillermo says, "If you go any lighter, you're gonna look like an ex
L699 +++$+++ u2 +++$+++ m0 +++$+++ CAMERON +++$+++ You always been this selfish?
L698 +++$+++ u0 +++$+++ m0 +++$+++ BIANCA +++$+++ But
L697 +++$+++ u2 +++$+++ m0 +++$+++ CAMERON +++$+++ Then that's all you had to say.
L696 +++$+++ u0 +++$+++ m0 +++$+++ BIANCA +++$+++ Well, no...
L695 +++$+++ u2 +++$+++ m0 +++$+++ CAMERON +++$+++ You never wanted to go out with 'me, did you?
L694 +++$+++ u0 +++$+++ m0 +++$+++ BIANCA +++$+++ I was?
```

GENERATIVE CHATBOT LIST OF CONVERSATIONS DATASET

- CORNELL MOVIE CORPUS

```
u0 +++$+++ u2 +++$+++ m0 +++$+++ ['L204', 'L205', 'L206']
u0 +++$+++ u2 +++$+++ m0 +++$+++ ['L207', 'L208']
u0 +++$+++ u2 +++$+++ m0 +++$+++ ['L271', 'L272', 'L273', 'L274', 'L275']
u0 +++$+++ u2 +++$+++ m0 +++$+++ ['L276', 'L277']
u0 +++$+++ u2 +++$+++ m0 +++$+++ ['L280', 'L281']
u0 +++$+++ u2 +++$+++ m0 +++$+++ ['L363', 'L364']
u0 +++$+++ u2 +++$+++ m0 +++$+++ ['L365', 'L366']
u0 +++$+++ u2 +++$+++ m0 +++$+++ ['L367', 'L368']
u0 +++$+++ u2 +++$+++ m0 +++$+++ ['L401', 'L402', 'L403']
u0 +++$+++ u2 +++$+++ m0 +++$+++ ['L404', 'L405', 'L406', 'L407']
u0 +++$+++ u2 +++$+++ m0 +++$+++ ['L575', 'L576']
u0 +++$+++ u2 +++$+++ m0 +++$+++ ['L577', 'L578']
u0 +++$+++ u2 +++$+++ m0 +++$+++ ['L662', 'L663']
u0 +++$+++ u2 +++$+++ m0 +++$+++ ['L693', 'L694', 'L695']
u0 +++$+++ u2 +++$+++ m0 +++$+++ ['L696', 'L697', 'L698', 'L699']
u0 +++$+++ u2 +++$+++ m0 +++$+++ ['L860', 'L861']
u0 +++$+++ u2 +++$+++ m0 +++$+++ ['L862', 'L863', 'L864', 'L865']
u0 +++$+++ u2 +++$+++ m0 +++$+++ ['L866', 'L867', 'L868', 'L869']
u0 +++$+++ u2 +++$+++ m0 +++$+++ ['L870', 'L871', 'L872']
u0 +++$+++ u2 +++$+++ m0 +++$+++ ['L924', 'L925']
u0 +++$+++ u2 +++$+++ m0 +++$+++ ['L984', 'L985']
u0 +++$+++ u2 +++$+++ m0 +++$+++ ['L1044', 'L1045']
```

BAG OF WORDS MODEL WITH NUMBER OF INSTANCES OF EVERY WORD IN THE ENTIRE DATASET

```
assimilate      3
assimilated     4
assist  19
assistance      27
assistant       83
assistants      9
assisted        2
assisting       6
associate       32
associated      11
associates      26
association     20
associations    3
assorted        2
assume  116
assumed 32
assumes 3
assuming        31
assumption      9
assumptions     5
assurance       11
assurances      5
assure  67
assured 19
assures 3
```

RETRIEVAL CHATBOT DATASET

```
{"intents": [
        {"tag": "greeting",
         "patterns": ["Hi there", "How are you", "Is anyone there?","Hey","Hola", "Hello", "Good day"],
         "responses": ["Hello, thanks for asking", "Good to see you again", "Hi there, how can I help?"],
         "context": [""]
        },
        {"tag": "goodbye",
         "patterns": ["Bye", "See you later", "Goodbye", "Nice chatting to you, bye", "Till next time"],
         "responses": ["See you!", "Have a nice day", "Bye! Come back again soon."],
         "context": [""]
        },
        {"tag": "thanks",
         "patterns": ["Thanks", "Thank you", "That's helpful", "Awesome, thanks", "Thanks for helping me"],
         "responses": ["Happy to help!", "Any time!", "My pleasure"],
         "context": [""]
        },
        {"tag": "CSE HOD",
         "patterns": ["Who is the CSE HOD?", "Who's CSE HOD?","What is the name of CSE HOD?", "Please give the name of CSE
         "responses": ["It is Maadhavi Ma'am", "It is Dr.K.Maadhavi", "Dr. K. Maadhavi"],
         "context": [""]
        },
        {"tag": "car parking",
         "patterns": ["where is the car parking?", "where is car parking?", "location of car parking?"],
```

III SAMPLE CODE

CODE FOR EXECUTABLE GUI FILE

```
from tkinter import *
import tkinter as tk
import os,sys
def convo():
    os.system("chat.py best_weights_training.ckpt")
def colg():
    os.system("chatgui.py")
root = Tk()
root.geometry('500x500')
root.title('Choose a BOT')
frame1 = Frame(root, bg='green')
frame1.pack(expand=True, fill=BOTH)
button1 = tk.Button(   root,
                        font=("Verdana",20,'italic'),
                        text="College Bot",
                       width="15",
                        height=5,
                       bd=0,
                       bg="#32de96",
                       activebackground="#3c9d9b",
                       fg='#ffffff', command= colg )
```

```
button1.pack(side=TOP)
button2 = tk.Button(   root,
                        font=("Verdana",20,'italic'),
                       text="Conversation Bot",
                       width="20",
                       height=5,
                       bd=0,
                        bg="#32de96",
                        activebackground="#3c9d9b",
                       fg='#ffffff',
                       command= convo)
 button2.pack(side=BOTTOM)
root.mainloop()
```

RETRIEVAL CHATBOT GUI

```
import tkinter
from tkinter import *
def input():
    abcd = EntryBox.get("1.0",'end-1c').strip()
    EntryBox.delete("0.0",END)
    if abcd!= '':
        ChatLog.config(state=NORMAL)
        ChatLog.insert(END, "User: " + abcd + '\n')
```

```
        ChatLog.config(foreground="#442255", font=("Arial", 10 ))
        pqrs = chatbot_response(abcd)
        ChatLog.insert(END, "Query Bot: " + pqrs + '\n\n')
        ChatLog.config(state=DISABLED)
        ChatLog.yview(END)
xyz = Tk()
xyz.title("Chatbot")
xyz.geometry("650x650")
CL = Text(xyz, bd=0, bg="yellow", height="10", width="70", font="Times New Roman")
CL.config(state=DISABLED)
SNDBT = Button(xyz, font=("Arial",10,'bold'), text="Enter", width="15", height=8,
                bg="#32de98", activebackground="#3c9d9b",fg='#ffffff',
                 command= input )
TextBox = Text(base, bg="blue",width="30", height="8", font="Times New Roman")
CtL.place(x=10,y=10, height=400, width=400)
TextBox.place(x=140, y=410, height=100, width=250)
SNDBT.place(x=10, y=410, height=100)
xyz.mainloop()
```