THE UNIVERSITY OF MELBOURNE

Mobile Chatting AutoBot

COMP90055 Computing Project

Software Development Project

25 Credit Points

submitted to

The University of Melbourne

Supervisor

Prof. Richard Sinnott

Sahaj Dhingra 960448

Hetal Shah 965690

Swapnil Shailee 952247

Semester 2, 2019

We certify that

- this thesis does not incorporate without acknowledgment any material previously submitted for a degree or diploma in any university, and that to the best of our knowledge and belief it does not contain any material previously published or written by another person where due reference is not made in the text.

- where necessary we have received clearance for this research from the University's Ethics Committee and have submitted all required data to the Department.

- the thesis is 8541 words in length (excluding text in images, table, bibliographies, and appendices).

Acknowledgement

We would like to express our special appreciation and thanks to our supervisor Professor Richard Sinnott for his guidance, encouragement, and advice throughout the project.

We would also like to thank all contributors to the open-source projects that assisted us for the completion of this task.

Abstract

Chatbots have been gaining popularity recently due to their help in reducing human effort. With the current technological advancements and the available datasets, it is now possible to create chatbots that can simulate human conversation. We have used NLP and deep learning techniques for creating a chatbot across three categories of datasets such as travel and bookings, customer support, and general chat. We have analysed the performance of the chatbot across the three datasets. The trained models of the datasets are then loaded on the server-side and integrated with a mobile chat application on the client-side. The mobile application has been developed using open source libraries and React Native platform. There is a storage component involved as well which stores the chats between the user and the chatbot to enable the corpus of conversation data to be used again and help train our model.

Keywords: Deep Learning, RNN, Chatbot, Mobile Application, MongoDB, React Native, Seq2Seq

Table of Contents

Table of Figures

1. Introduction

Since their introduction, chatbots are bringing about a revolution and changing customer's interaction with brands. Alongside this, they are vital in saving the companies time, money and effort. They have also been helping improve customer experiences. An Oracle survey mentioned that by 2020, 80% of brands intend to use a chatbot [1].

This report describes the authors' attempts for the development of an independent mobile application integrated with a chatbot. The chatbot spans three categories namely travel and bookings, customer support, and general chat. After training the datasets, the trained models are stored on the server-side and queried through the client i.e. the mobile application to generate a response.

In this chapter, we provide a high-level introduction to the concept of a chatbot and its background and basic development steps involved. In chapter 2, the datasets along with the data pre-processing steps are discussed. In chapter 3, we are describing the concepts and models related to the chatbot, the training process along with providing some results and analysing the trained models. In chapter 4, the implementation process which involves architecture of the mobile application along with its integration with the trained models using the various technologies and their deployment on the server and mobile client. In chapter 5, we look at some of the challenges we faced and the limitations of our application. In chapter 6, we are discussing some of the future works that can be implemented with this project and in chapter 7, we conclude the report.

1.1. What is a Chatbot?

In terms of defining a chatbot, we can define it "*as a computer program that impersonates human conversations in its natural format, which may include text (since the advent of bots) or spoken language using artificial intelligence (AI) techniques such as Natural Language Processing (NLP) and audio analysis*" [2].

Short for chat robot, a chatbot is a software or program that has a conversation with humans through the medium of a chat. Through its live chat interface, a chatbot can scan the keywords in the request of the customers and responds accordingly [3].

Chatbots work through either Artificial Intelligence or rule-based commands [4]. The chatbot can be used to respond to simple or complex conversations.

A typical use of a chatbot is to communicate like a human with other humans. Nowadays, there is an increasing number of chatbots that interact with other chatbots [5].

1.2. How do the Chatbots function?

The primary piece of technology running behind a chatbot is Natural Language Processing and Machine Learning.

Whenever a chatbot encounters a question, a set of rules or some range of complex algorithms process the input received, understand the context of what the user is trying to ask and accordingly present an answer which may be a suitable response to the query.

Chatbots are dependent on an algorithm's ability to identify the text complexity and also the complexity of speech which are spoken words. A lot of chatbots recently perform a conversation on par with a human hence making it difficult for the user to make the distinction between a chatbot and a human.

An AI-based bot is primarily dynamic. Such bots learn from their historical interactions and accordingly become more intelligent to handle complex and intricate conversations. Another challenge is how to handle conversations where multiple complex emotions, various figures of speech are used which are very difficult to understand for the machine [6].

1.3. Why are Chatbots rising in popularity?

A prediction by Gartner says that by 2020, people on average will end up conversing more with a chatbot compared to their partners [1]. Chatbots promise a smart and intelligent 24x7 digital support. These features make them attractive to companies who can use them for engagement with customers apart from the traditional channels of email, phone, and social media. Over 50% of service organizations are moving towards implementation of a chatbot in the next 18 months which predicts a 136% rate of growth, thus foreshadowing a bigger role for technology in the coming future [7].

Productivity and efficiency are boosted in a number of ways in the workplace by implementing the use of chatbots by businesses. Meetings and reminders are set by the workers with the help of chatbots, chatbots also assist in asking the questions without interrupting the ongoing work. These are some of the scenarios where chatbots are used. Along with it, consumers make use of certain virtual assistants like Amazon's Alexa, Apple's Siri and Google Assistant and also prefer these interfaces for brand engagement across every industry [2].

As a tool of establishing interaction between business houses and their customers, so as to solve their problems, chatbots were introduced. It was noticed that repetitive questions were asked by most of the customers. So as to save the valuable time of the executive, who works on solving the same issues faced by the different customers, businesses and other machine learning consulting companies decided to come up with the idea of chatbots; an automated care process for the customer. Resolving the issues of the customers at an increasing speed has been possible with the help of chatbots [8].

Reducing the efforts of the customer support team to apply their emotional intelligence by giving instant responses when needed to many other complicates questions is also an essential feature of chatbots [9].

1.4. Comparison of Chatbot Models

There are two main types of chatbots. A chatbot is either based on pre-programmed responses (Rule-Based) or Artificial Intelligence (AI) to answer a user's questions without the need for a human agent.

1.4.1. Rule-based chatbots

The first and maybe the most basic bots are rule-based chatbots, otherwise called decision tree bots. These bots are the most widely recognized, and a large number of us have likely collaborated with one on social media, online shopping sites or live customer chat support [10].

Rule-based chatbots can hold fundamental discussions dependent on "if/then" rationale. These chatbots don't understand the context or intent of a conversation. Human operators guide out discussions through a flowchart, envisioning what a client may ask, and program how the chatbot should react. We utilize sensible logical stages and clear commands to construct rule-based chatbots discussions. Organizations configure rule-based chatbots to respond to basic inquiries and regularly bring web guests to a live operator to facilitate the discussion. They are not intended to learn and get more brilliant after some time. We can manufacture a standard rule-based chatbot with straightforward or increasingly complex guidelines. They can't, be that as it may, answer any inquiries outside of the commands.

Rule-based chatbots don't learn through collaborations and just perform and work inside situations for which they are programmed for. As the name recommends, they are powered through a progression of characterized rules. These guidelines are the reason for the sorts of issues the chatbot knows about and can convey answers for [4].

Discussions are mapped out like a flowchart to foresee what a client may ask, and how the chatbot should react. To process the voice input, rule-based chatbots scan for recently characterized catchphrases or keywords [11]. Contingent upon which keywords are discovered, the proper answer is given.

The more intricate a framework turns into, the more troublesome it will be to cover every single important situation and make suitable standards or rules. Particularly if information changes inside a brief span, the manual change takes a massive amount of time. That is the reason, rule-based chatbots are helpful for a restricted amount of tasks, as long as the essential information doesn't change routinely.

Some different points of interest of a rule-based chatbot are that they:

- are by and large quicker to prepare and train (more affordable)
- integrate effectively with legacy frameworks and systems
- streamline the handover to a human specialist or agent
- are profoundly responsible and secure
- can incorporate intelligent components and media which are interactive
- are not limited to text communications

1.4.2. AI-Powered Chatbots:

Artificial intelligence-based chatbots are increasingly complex and modified bots dependent on Normal Language Processing (NLP) and AI (ML). These chatbots learn on the go, unlike rule-based chatbots. Artificial intelligence-based chatbots that utilize machine learning are built to comprehend the unique situation and purpose of an inquiry before defining a reaction [4]. These chatbots create their own responses to increasingly entangled inquiries utilizing regular language reactions. The more these bots are utilized and prepared, the more they learn and the better they work with the client.

In the initial step, they train with an example dataset [10]. The chatbot utilizes this test dataset to identify regularities and get further answers from them. When the bot is being used, each new information is utilized to keep learning.

Be that as it may, on the grounds that a chatbot depends on AI doesn't imply that it is appropriate for every theme or issue. There is a vertical and a horizontal AI spectrum. Chatbots with a vertical computer-based intelligence are arranged for one undertaking. They do it very well

indeed, so they can nearly be mistaken for an individual. Conversely, chatbots that have a horizontal AI can help with different undertakings.

AI Bots are considered to be more sophisticated. Clients can converse with the bot about pretty much anything. They function admirably well for organizations that will have a great deal of information. In spite of the fact that they take more time to prepare at first, these intelligent chatbots spare a ton of time over the long haul [10].

Some of the advantages of AI chatbots are:

- learn from the previously collected information
- routine improvement in responses as more relevant data comes in
- understand and store behaviour patterns
- able to answer a broader range of questions
- allowance for multiple language support (requires special programming)

1.4.3. Comparison

Model	Benefits	Limitations
Rule Based Model	Very Fast	Works only for predefined set of commands
Convolutional neural network (CNN)	Works well for extraction local and position-invariant features. Eg: searching for angry terms, sadness, abuses, named entities etc	Information about the general order of words is usually lost
Recurrent Neural Network (RNN)	Classification works based on a long range semantic dependency rather than some local key-phrases. Eg: language modeling or machine translation or image captioning	RNN is 5x slower than CNN based on computation time

Table 1: Comparison of different types of chatbots [12]

1.5. Chatbot Development

For our purposes, we have selected a machine learning and NLP based approach. The basic steps involved in the creation of our chatbot [5] were as follows:

- **Identification of Domain** – This involved decisions based on whether our chatbot would address generic queries or be specific to a domain. We have selected a specific category such as travel and bookings alongside a general conversation as well.

- **Identification of Datasets** – After identifying the domain, it is important to start collecting or identifying the datasets upon which the model can be trained. This is a very crucial step since the model is only as good as its data.

- **Data pre-processing and cleaning** – Even after obtaining the datasets, it is imperative to format the data as per the requirements of the training functionality. This involves changing the format, removing additional and irrelevant information from the datasets and much more.

- **Training the model** – Selecting the model and training the datasets is done before the trained file can be used for querying purposes. Care must be taken not to overfit or underfit the model to the data [13].

- **Testing the model** – A very significant part of the chatbot development is testing or analysing the results to ensure that the chatbot is at least providing some decent responses [6].

These steps are further explained in detail in the later chapters.

2. Dataset

Dialogue based or conversational datasets is needed to build a chatbot. This section describes the process of obtaining the datasets and various tasks that were followed. The chatbot is built for the following use-cases:

- General
- Bookings
- Customer Support

2.1. Data Collection

Sr. No	Dataset	Number of Dialogues
1	General	1,88,378
2	Bookings	1,43,048
3	Customer Support	30,03,124

Table 2: Number of dialogues in each category

2.1.1. General

The Amazon Alexa Topical Chat Dataset has been used for the implementation of the chatbot for General use-case. It was recently released on September 17, 2019. Topical Chat Dataset, a text-based collection of dialogues that will help support high-quality, repeatable research in the field of dialogue systems [14].

The objective of Topical Chat is to empower innovation and research in learning grounded neural response generation frameworks by handling limitations and challenges that are not tended to by other freely accessible datasets. Both the discussions themselves and the

explanations connecting them to specific information sources were given by laborers selected through Mechanical Turk.

2.1.2. Bookings

The Multi-Domain Wizard-of-Oz dataset (MultiWOZ), a collection of human-human written conversations spanning over multiple domains and topics released by the University of Cambridge on July 10th, 2019. The dataset was collected based on the Wizard of Oz experiment on Amazon MTurk. Every dialogue comprises of a goal label with multiple exchanges between the system and the visitor and the system. Every turn of the system includes labels from within a set comprising on slot and value pairs. These represent a state of dialogue in a coarse format for the system as well as the user. This dataset consists of booking conversations related to hotels, restaurants, trains, attractions, hospitals, police stations and cabs [15].

2.1.3. Customer Support

The Customer Support dataset on Twitter is an enormous, present-day corpus of tweets and answers to help advancement in understanding natural language and conversation models, and for research and study of current customer support care practices and their impact.

The Customer Support on Twitter dataset offers a large corpus of modern English (mostly) conversations between consumers and customer support agents on Twitter and has the following important advantages over other conversational text datasets [16]. Firstly, it is more focused as the consumers contact customer support to have a specific problem solved and the problems to be discussed are relatively small when compared to Reddit Corpus. Lastly, the responses are more natural rather than scripted. The dataset is convenient for training recurrent neural networks [16].

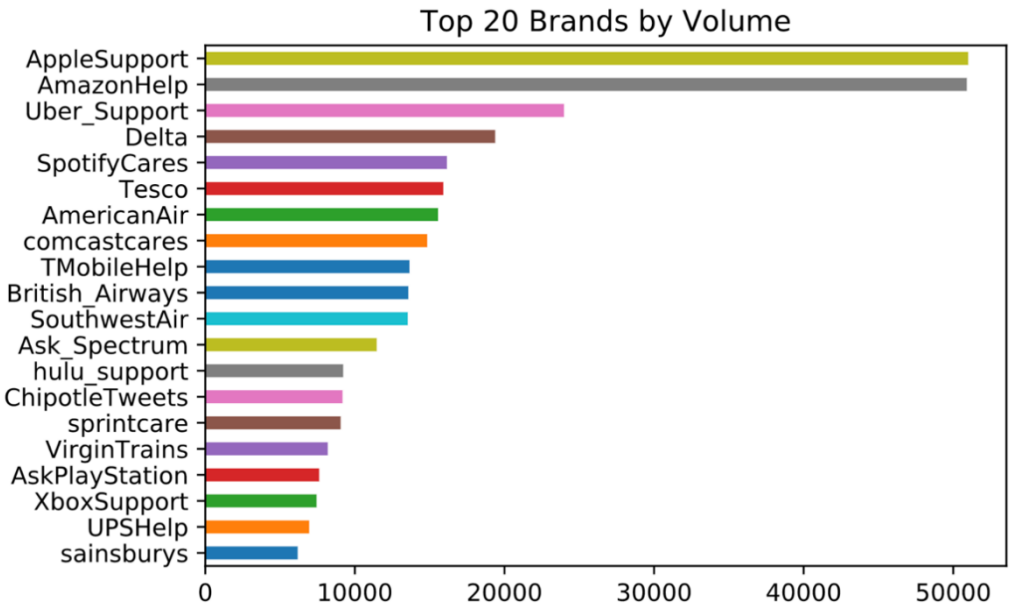

Figure 1: Company based data in Customer Support Dataset [16]

2.2. Data Pre-processing

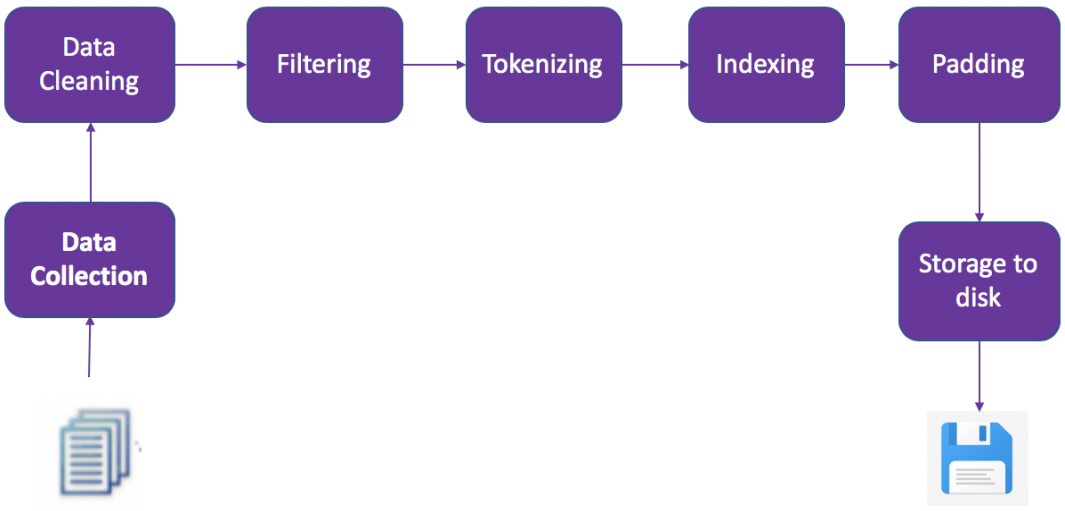

Figure 2: Data Pre-processing flow diagram

The dataset obtained is converted to the required format of conversations or dialogues for pre-processing by using various data cleaning techniques using python libraries like json, pandas, regex, etc.

For example, the twitter handles and URLs in tweets have been removed from the Customer Support twitter dataset while conversion. The following steps are performed in order to pre-process the dataset [16].

Text before cleaning	Text after cleaning
@115712 I understand. I would like to assist you. We would need to get you into a private secured link to further assist.	I understand. I would like to assist you. We would need to get you into a private secured link to further assist

Table 3: Example of data after cleaning.

Step 1:

Reading:

The dataset containing conversations are read and converted to lowercase.

Step 2:

Filtering:

The dataset is filtered in two stages. These are described below:

a. **First Stage Filtering:**

Irrelevant characters are removed from the dataset. Only alphanumeric characters are allowed to be present in the dataset. For example: In the Customer Support dataset, there were non-English conversations that can be removed after this stage of filtering. Also, in this stage, the emojis and hashtags were removed from the data.

b. **Second Stage Filtering:**

In this stage, only certain allowable length of sentences are allowed to be present in the dataset. Extremely short and long sequences are removed from the dataset. The minimum allowable length of sentences is set to 3 and the maximum has been set to 20.

Type of texts filtered	Example
Hashtags	We like to go before the flight for a civilized coffee and newspaper. #travel #Entrepreneur #crossculture #coach #speaker
Foreign language	アマゾンって支払いの期限あるっけ？
Emojis	Shame it doesn't do lost baggage reporting, my bag has been left in Miami 😕

Table 4: Example of data after the first stage filtering

Step 3:

Tokenising:

The conversations are present in the form of a list of lines that are tokenized into words and stored as tokenised questions and tokenised answers in a list.

Step 4:

Indexing:

In this process, the list of words from the dataset is read and converted to the index to word, word to index dictionaries. Word Vocabulary is implemented based on the frequency distribution of most common words for a limit of 6000 using the NLP package – NLTK [17]. The words not in vocabulary are represented by Unk token.

Step 5:

Padding:

The padding of zeroes is added to sentences in order to keep the sequences of equal length.

Step 6:

Storage to disk:

The indices to questions, indices to answers of the dataset are stored to the disk. Dictionaries like a word to index, index to word, limit and vocabulary are stored as metadata to the disk using pickle.

Step 7:

Splitting of Dataset:

The dataset is then into training, test and validation sets in the ratio of 0.70, 0.15 and 0.15.

2.3. Dataset Analysis

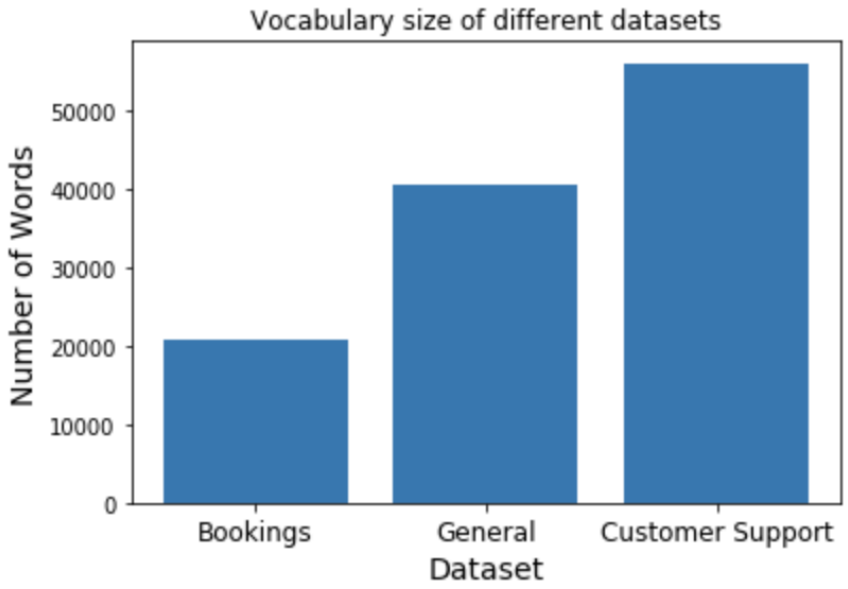

Figure 3: Plot depicting the number of words in each datasets

The number of words present in the different datasets has been calculated which depicts that the bookings dataset contains 20,889 words, the general dataset contains 40,532 words and the Customer Support dataset contains 56,081 words. The vocabulary is calculated by using NLP packages. The vocabulary size of the Bookings dataset is close to the limit of vocabulary size that has been selected for training the model.

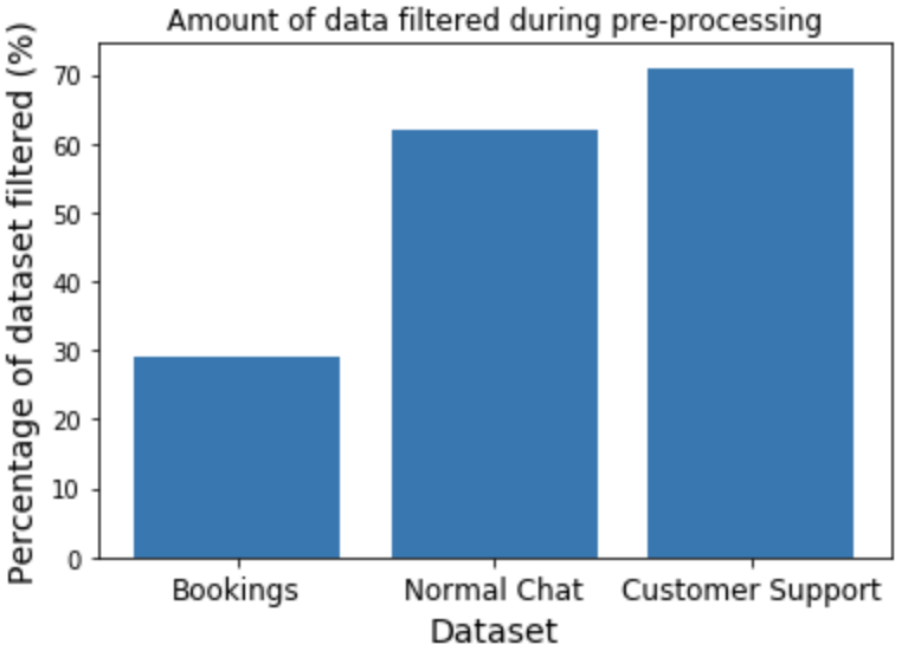

Figure 4: Plot depicting the percentage of dataset filtered

It is observed that the Customer support dataset has the highest amount of filtering because of the presence of foreign language, hashtags, emojis, website links, etc. Bookings dataset is the cleanest among the three datasets.

3. Chatbot Implementation

3.1. Natural Language Processing

Natural language processing is a field in machine learning where computers are able to comprehend human dialects [17]. Most NLP methods depend on AI to get importance from human dialects. Natural language processing is the crucial guiding force behind applications like language translation applications, word processors that check for grammatical errors in texts, speech recognition applications, and chatbots.

Chatbots have become more and more prevalent over the past years and have mostly functioned as rule-based conversations where the chat output is fixed. For example, in rule-based chatbots, greetings and end of conversation messages are predetermined. NLP helps in deriving meaning from user text inputs and would help in responding to complex user queries. NLP adds human essence to the chatbot.

The various undertakings in NLP are syntactic and semantic analysis. Syntactic analysis techniques incorporate lemmatization, morphological segmentation, word segmentation, part of speech tagging, parsing, sentence breaking and stemming. Semantic analysis techniques incorporate named entity recognition (NER), word sense disambiguation and natural language generation [18].

3.1.1. Word Embedding

Word embeddings are texts converted into numbers as machine learning or deep learning architectures are unequipped of processing text data in their native form. There is a requirement of numbers as input to perform any kind of deep learning or machine learning. It maps a word using a dictionary to a vector. It can be further grouped into Frequency based Embedding and Prediction based Embedding [19].

Embedding layer in the neural network for textual data is offered by deep learning libraries like tensorflow, keras, etc. The necessity of the embedding layer is that the input data needs to be integer encoded so that each word is determined by a unique integer. The embedding layer is initialized with random weights which will learn an embedding for all the words in the training dataset. The layer can be used to learn a word embedding that can be saved and used in another model later. The layer can also be used as part of the deep learning model where the embedding is learned along with the model itself. It can be also used for loading a pre-trained word embedding model which is also called transfer learning [19].

The vocabulary of words created for the different datasets is used in the embedding layer of the deep learning model.

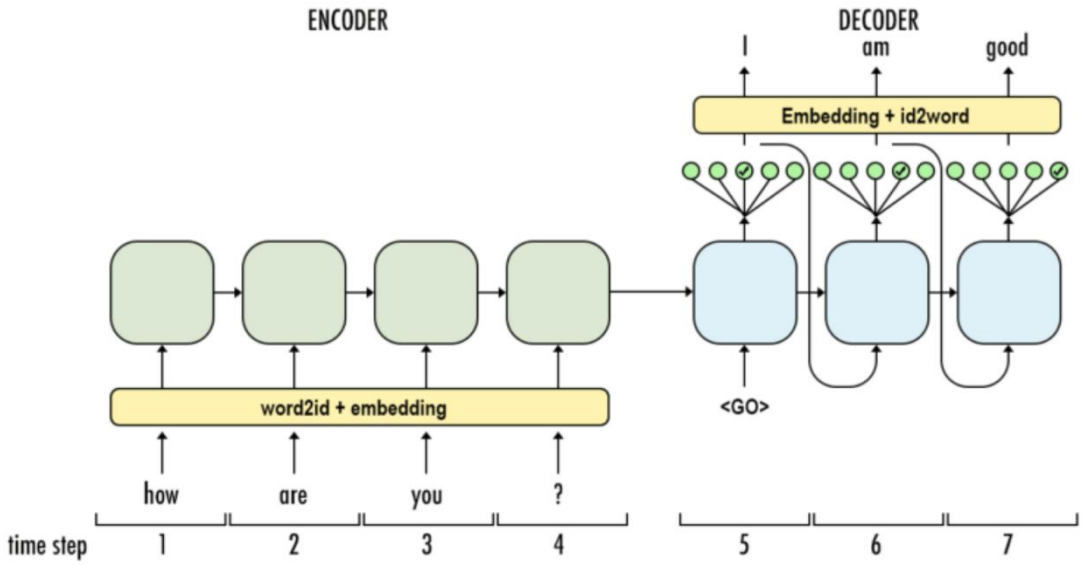

Figure 5: Embedding layer in Seq2Seq model [20]

3.2. Deep Learning

Deep Learning is an exceptionally imaginative innovation with tremendous potential. It is a subfield of machine learning concerned with algorithms motivated by the structure and function of the brain called artificial neural networks. Neural networks are inspired by our understanding of the biology of our brains – all those interconnections between the neurons.

The artificial neural networks have discrete layers, connections, and directions of data propagation. Virtual Assistants, speech recognition, face recognition, language translation are some of the important applications of Deep learning applications [21].

3.2.1. Recurrent Neural Network

Recurrent neural networks also known as RNNs are a type of neural networks, which are used in the cases where saving the previous states is necessary. These are mainly used in text analysis where the order of the words is necessary like in chatbot, translating text to speech, predicting next word and speech to text [22].

RNNs are mainly used where the next word is generated based on the values of weights of the previous words. The working of the RNN is based on the any size input for which weights are calculated. The only condition for the input is that the size of the input should not be more than the model size [23].

RNN mainly uses cyclic hidden states and work better for more complex task. Its uses the concept of storing the previous sequences into the memory so as to predict the new state [22]. This data is stored in the form of chains which is used to generate the best possible sequence based on the previous word [22].

3.2.2. Memory Structures in RNN

Short term memory has always been a problem in implementing of RNN. Since RNN stores the previous sequences using the backpropagation and updates the weights and biases accordingly [22]. At each iteration the derivatives of the loss function are calculated to adjust the parameters. This causes a problem when multiple iterations are run because, by using the same parameters the values can become too large or too small. This results in undefined weights and biases when the values are too large [22]. Also, in case of small values there is no learning as there is nothing to update.

A solution to this problem is using GRU (also called as Gated Recurrent Unit) and LSTM (Long Short Term Memory). These structures can be used to determine when to erase the previous values from the memory or on the other hand when to propagate forward [22].

GRU model:

This structure solves the problem mentioned above by using the update and reset gates in the RNN cell. This can be easily implemented by using these gates. The data that is to be deleted is decided by the update gate. Moreover, it also decides what new data should be added. The job of the reset gate is to determine the life of the past data. It can tell how much previous data can be ignored. This makes GRU model faster than LSTM[22].

LSTM model:

LSTM works with cell state and gates. To transmit the information across the sequence chain maintained by RNN, cell state is used. During the whole cycle of RNN, the cell state is used as a transportation medium, on which data can be added or deleted with the help of the gates. The gates are neural networks that act as a monitor for the information that should be added or deleted from the cell states. These gates can be learn new information, thus solving the problem of short term memory [22].

Steps:

1. The current input is appended to the last hidden state. This operation is called as the combine operation.
2. To remove the irrelevant data, the result of the combine operation is fed as an input to the “forget layer”
3. After this, a layer is created that holds the values for the cell state to work. This layer is called as the candidate layer [22].
4. Also, the result of the combine layer is also fed into the input layer. This is done so that the data to be added to the new cell state can be determined by this layer [22].
5. Finally, when the three layers are calculated namely input layer, candidate layer and the forget layer, using the previous cell state and the vectors from these 3 layers, the cell state is calculated. Also, the new hidden state is calculated by doing pointwise multiplication of the output and this new cell state [22].

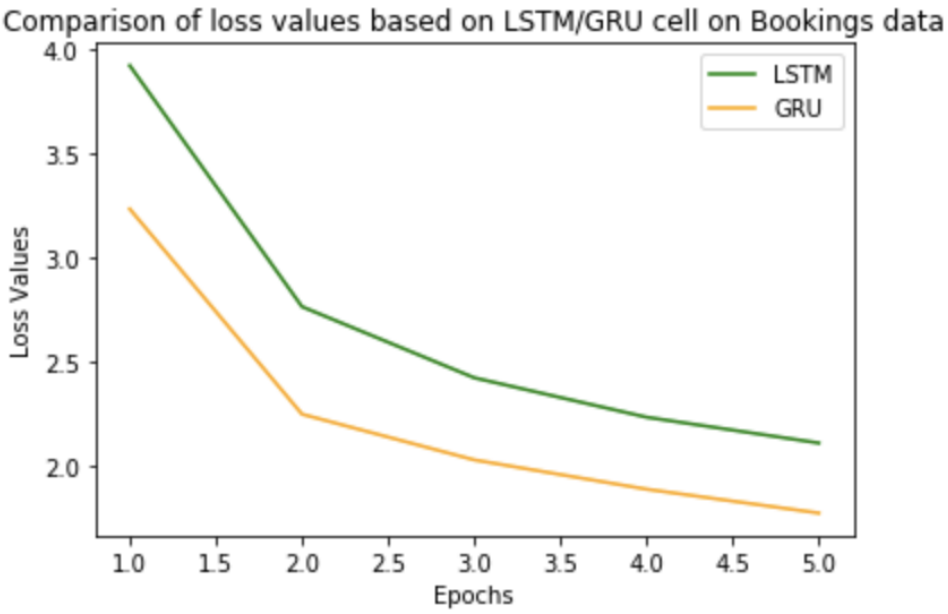

Figure 6: Comparison of loss values based on LSTM/GRU cell on Bookings data

GRU cell has been implemented in the model as it provides better results as compared to the LSTM cell.

GRU	LSTM
Trains faster and simple to implement	Trains slower than GRU and more complex than GRU
Performs better for short sequences.	Remembers longer sequences hence , would perform better if user types long sentences.
Loss value after 5 epochs : 1.77 (Bookings dataset)	Loss value after 5 epochs : 2.11 (Bookings dataset)

Table 5: GRU v/s LSTM comparison

3.2.3. Optimizers

Optimizers combine the loss function and model parameters together and help in improving the performance of the model based on the output of the loss function [24].

Adam Optimizer and Stochastic Gradient Descent Optimizer have been tested in the model where Adam Optimizer has provided better results [25].

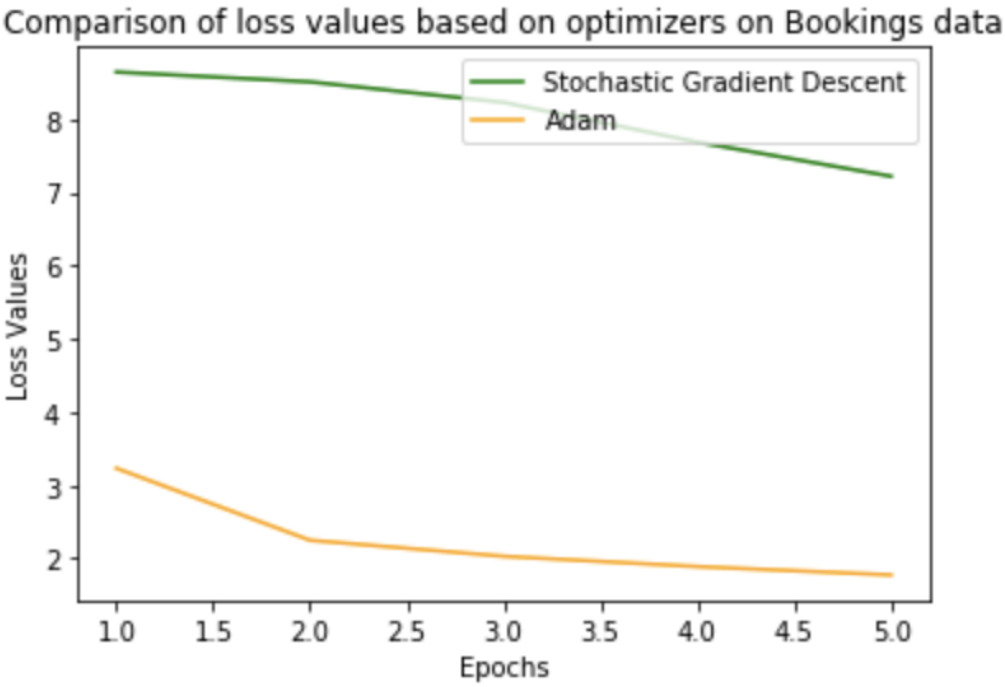

Figure 7: Plot depicting performance of the two optimizers on different datasets

Different learning rates have been tried and finally, Adam Optimizer with a learning rate of 0.001 has been used in the model.

3.3. Deep NLP (SEQ2SEQ Model)

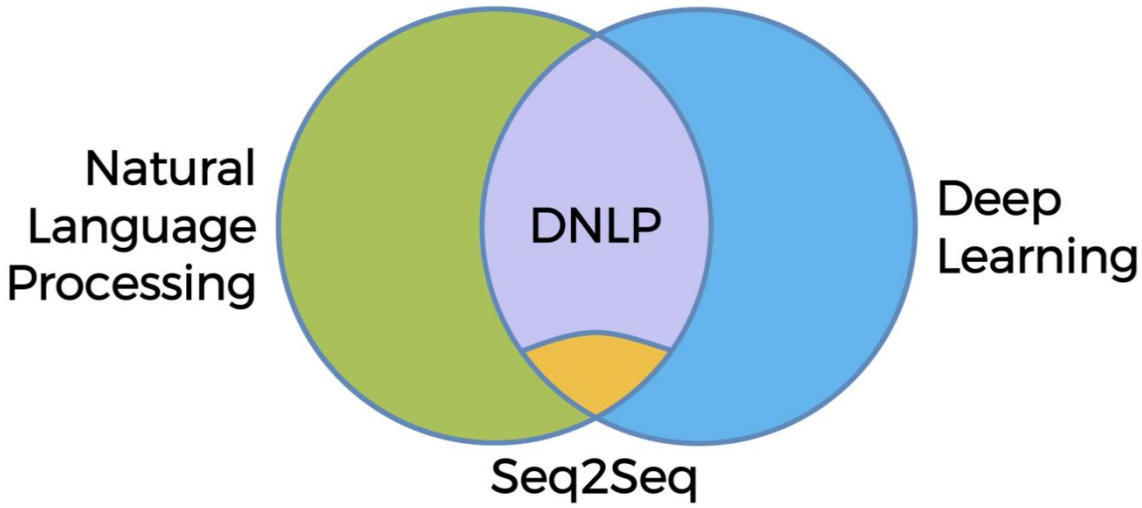

Figure 8: Seq2Seq research area

Sequence to Sequence model has been used for implementing the chatbot. As depicted in the above figure (figure 8), this model lies in the middle of NLP and Deep Leaning. This deep NLP model shares features of both NLP and Deep Learning which makes it suitable for the chatbot.

The Seq2Seq model is made up of 2 RNN one encoder and one decoder, which is connected to by the encoder input [20]. The word sequence is fed to the encoder one word at a time. The encoder foresees the meaning of the input sentence by generating the final hidden state of the encoder called as context. This context is then fed to the decoder as input, which generates the output sequence considering the previous sequences from the context. The context can be an input to the initial state for the RNN decoder or at each timestep it can connect to the hidden units [26].

In seq2seq model, the states are dependent on the previous states. The next hidden state is directly affected by the current hidden state forming a chain of sequences. Since seq2seq cannot work with variable length data, in the pre-processing step, the data padding was done to make the input and output for same length. In this case, three symbols namely <EOS> , <PAD> and <UNK> are added to solve this problem.

The encoder contains various layers namely the embedding and memory layers like GRU or LSTM. The model implemented contains GRU layer. The embedding layer helps in storing the input into a predefined sized dictionary of words. The output of the embedding layer is then passed to the GRU layer. The GRU layer helps in the calculation of hidden state from the preceding layer. It then updates the reset , update and new gates. The decision to erase and propagate the information is controlled by the GRU layer [20].

The decoder of the model consists of three main layers namely embedding layer, GRU layer and a dense layer without activation. The embedding layer has a look up table to generate the output which is fed to the GRU layer as input for calculating the predicted output state. To solve the problems like vanishing gradient, activation like Relu can be used [20].

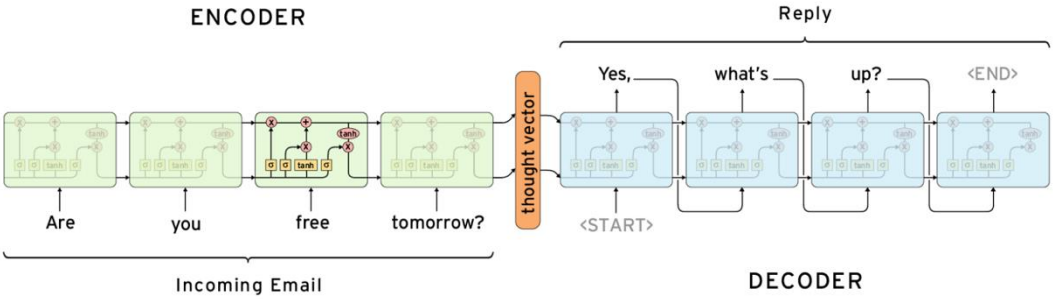

Figure 9: Sequence to Sequence model [26]

3.4. Training

3.4.1. Training Process

After the completion of the data pre-processing step, the metadata containing dictionaries, indices to questions, indices to answers that are stored on the disk loaded to memory. The training, testing and validation dataset are loaded. The model is trained on 70% of training data, 15% test, and 15% validation data. The vocabulary size of set for the embedding layer. The embeddings are in the word to vector format. The model is implemented using the seq2seq of the tensorlayer [27] library. The model is built using the GRU cell for the cell encoder and cell decoder with 3 layers and 256 units. Adam Optimiser has been incorporated for minimising the losses [24].

The model is then trained for 50 epochs for each dataset and the model weights are saved for obtaining predictions for user queries. Training loss and predictions after each epoch for a sample of test data is logged. The loss values and predictions are further used for evaluating the model.

Hyperparameter	Value
Batch Size	32
Embedding dimension	1024
Learning rate	0.001
Number of epochs	50

Table 6: Hyperparameters used for training the model

Dataset	Training Time (in hours)
General	21.3
Bookings	30.91
Customer Support	7.86

Table 7: Training time for different datasets using GPU

3.4.2. Performance Analysis

The model was computationally expensive (could not train) for the Customer Support dataset. To overcome this problem, the method of sampling has been used where the model was trained on a subset of the dataset. After sampling, the model could be successfully trained for the Customer Support dataset.

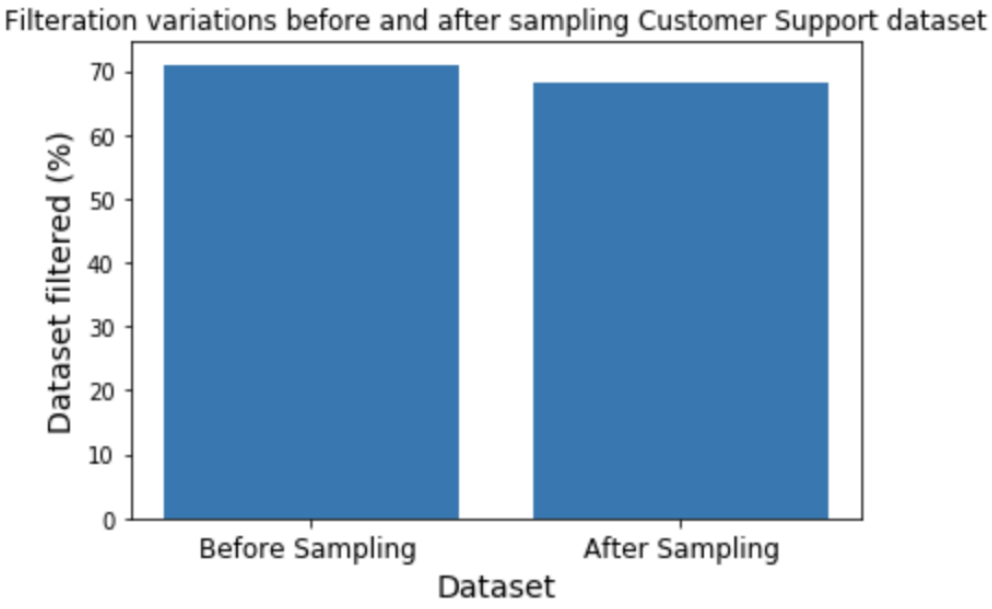

Figure 10: Plot depicting the variations in filtration before and after sampling

Minor variation in filtration can be observed after sampling the dataset.

3.5. Result and Analysis

3.5.1. Bilingual Evaluation Understudy Score (BLEU)

It is a metric for evaluating a generated sentence to a reference sentence. It is a method that has been developed from a precision method that is used for automatic evaluation of machine translation to human translation [28].

A perfect match results in a score of 1.0, whereas a perfect mismatch results in a score of 0.0. Few chatbot responses can attain a score of 1[28].

Multiple chatbot responses can be accurate hence, BLEU is an important measure to evaluate chatbot responses [29]. It correlates highly with human responses. The idea behind BLEU is to count the number of matching n-grams in the sentences. Unigram is a word, bigram is a pair of two words, etc [28] Order of words is not important. BLEU doesn't consider synonyms into consideration. It is a fast and efficient algorithm that highly correlates with human evaluation. For example, in the below chatbot conversation, the human agent reply and chatbot reply is highly similar though the BLEU score is 0.12. BLEU is calculated based on the presence and absence of particular words and their ordering. It is sensitive to word breaks.

Google's Multilingual Neural Machine Translation System, achieves a score of around 38.0 (0.38*100) on Spanish-to-English translation which is an excellent translation model [22].

User Input	Agent Reply	Chatbot Reply	BLEU Score
I am looking to book a train that is leaving from Cambridge to Bishops Stortford on Friday.	There are a number of trains leaving throughout the day. What time would you like to travel?	what time will you be traveling	0.12

Table 8: BLEU score of Chatbot calculated with agent and chatbot reply

Actual Conversation	Chatbot Reply 1	Chatbot Reply 2	Chatbot Reply 3
User: am looking for a place to to stay that has cheap price range it should be in a type of hotel	i have 10 cheap places to unk in town is there a specific area youre need a hotel in the	i would recommend express by 1445 how many people would you like	i have 10 hotels that match that description would you like me to search for one hotels or a north
Bookings Agent: Okay, do you have a specific area you want to stay in?			
BLEU Score	0.73	0.55	0.69

Table 9: BLEU Score calculation between the top 3 chatbot replies and the agent reply

It can be observed that Chatbot reply 1 has the highest BLEU score as "specific area" bigram appears in both the sentences.

For a sample of actual and chatbot responses for the three datasets, the BLEU score has been calculated and it is in the range of 20-30% which is fairly good.

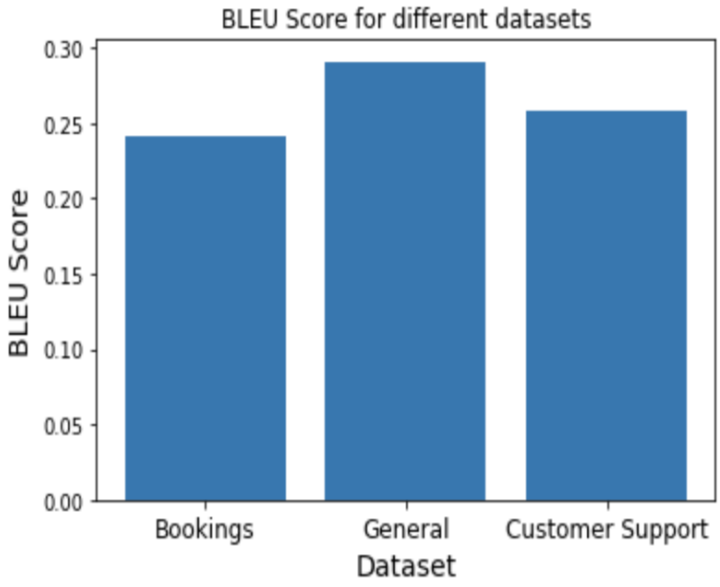

Figure 11: Plot depicting the BLEU scores for the three datasets

3.5.2. Loss

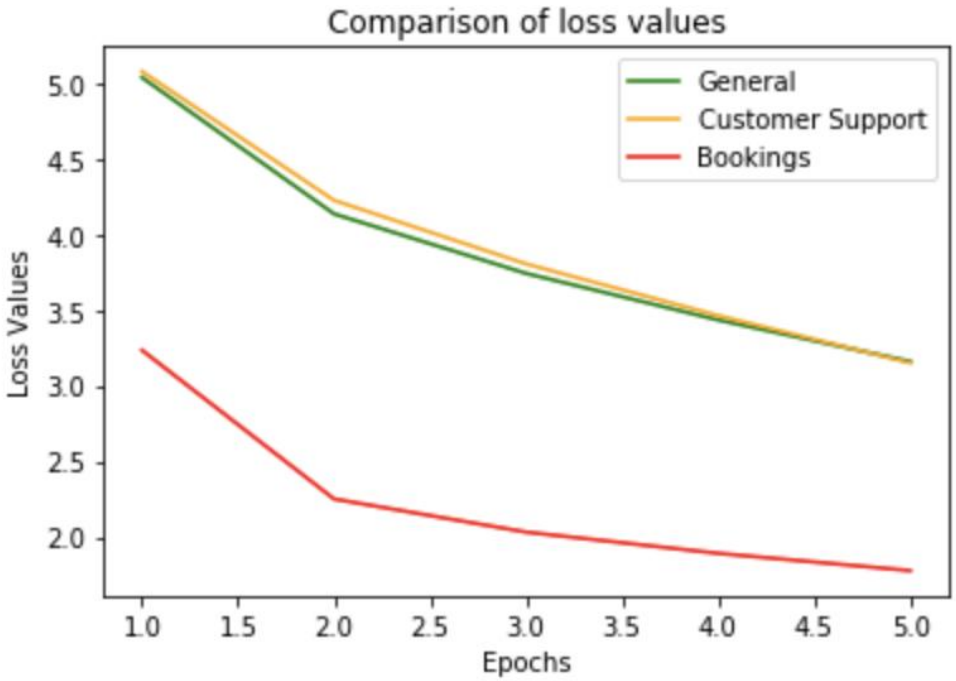

Figure 12: Plot depicting training loss after the first 5 epochs for the three different datasets

It can be observed that after the first 5 epochs, the sequence to sequence model performs best for the bookings dataset. After completion of 50 epochs, the least loss value is for the Customer Support dataset. After 30 epochs, there is not much variation for the General and Bookings dataset. Slight variation can be observed for the Customer Support dataset.

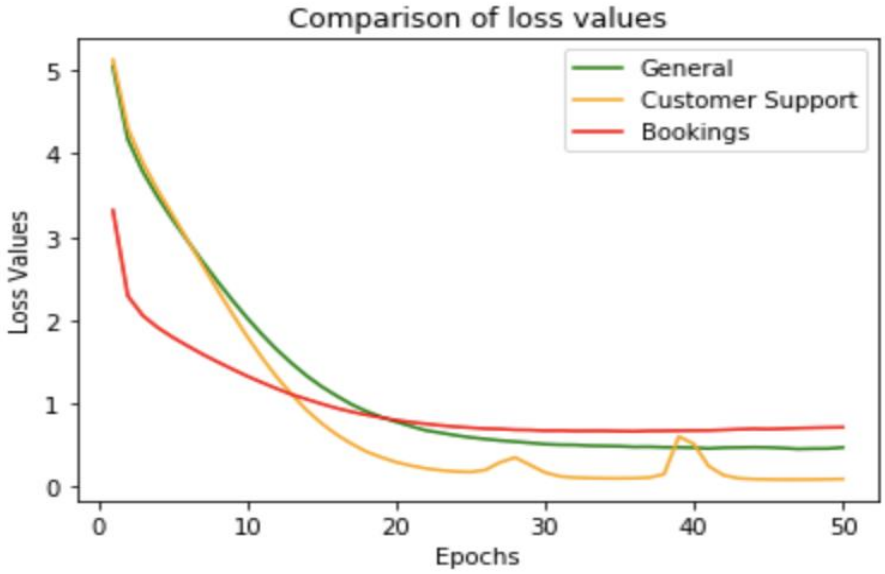

Figure 13: Plot for training loss values after 50 epochs

Validation and training loss values have been calculated for the training and validation sets for all the datasets.

Figure 14: Comparison of loss values for Customer Support dataset

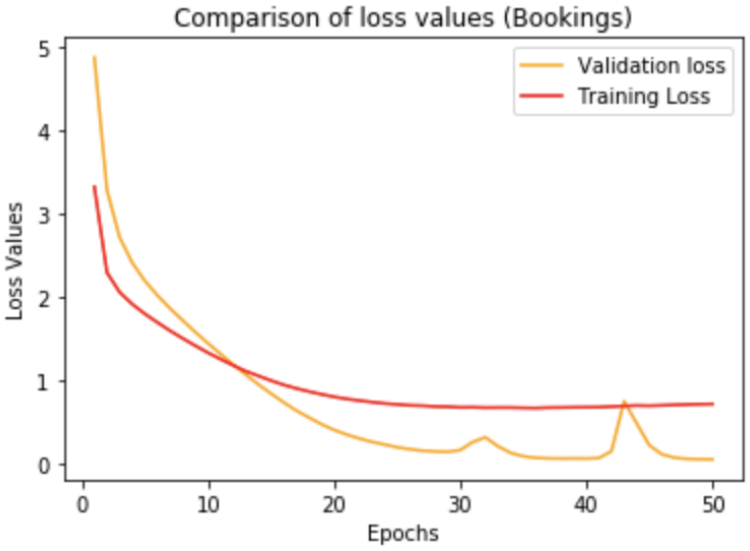

Figure 15: Comparison of loss values for Bookings dataset

It can be observed that the Seq2Seq model trained for the different datasets over 50 epochs is a good fit as there is a minimum gap between the two loss values: training and validation loss. Variation in validation loss can be observed after 30 epochs, hence the optimum epoch value for training is near 30.

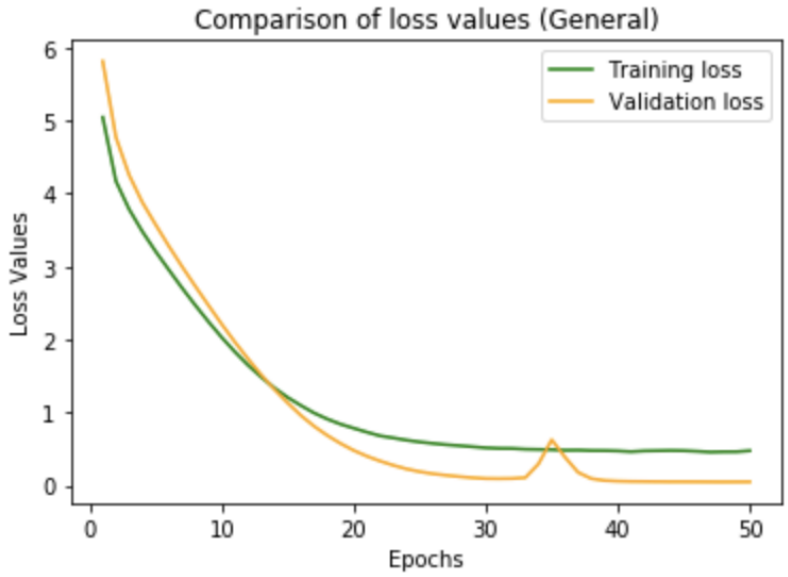

Figure 16: Comparison of loss values for General dataset

3.5.3. Response Time

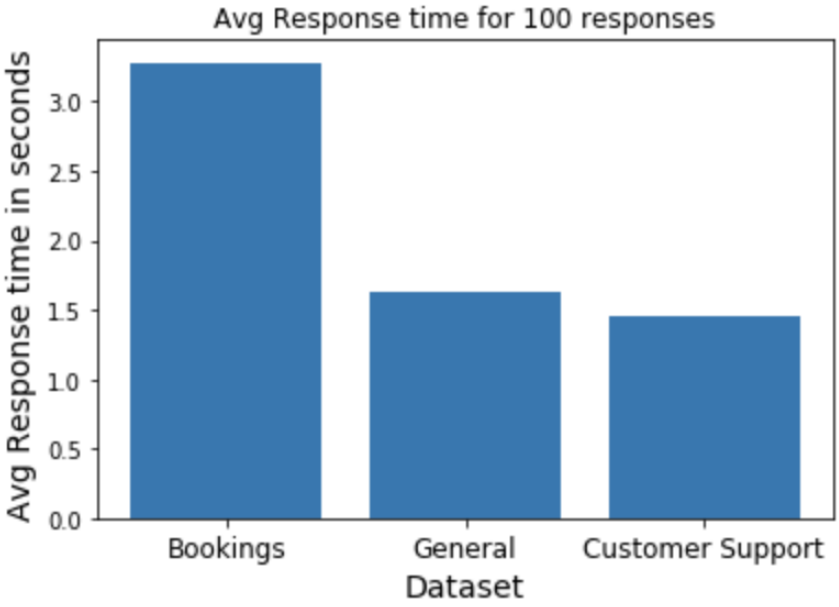

Figure 17: Response time of chatbot for all 3 datasets

The chatbot performance is also measured by calculating the response time of 100 responses on average for all 3 different categories. As can be observed from figure 17 above that the response time of bookings dataset is 3.28 seconds which is the maximum. This is expected as the training time of bookings dataset is maximum. For general and customer support dataset the response time is 1.63 and 1.45 seconds respectively.

4. Chatbot Integration

Various different components have been developed during the implementation of this project. These include the following -

a. Client-Side Mobile Application
b. Flask Socket Server
c. MongoDB Database

These models are described in the following sections.

4.1. Implementation of the Flask Socket Server

4.1.1. Technology

The flask socket server is built various different python libraries. These include the following

a. **Socket IO Library**

Socket.IO enables bi-directional and event based communication in real time. This library works on every device, browser or platform and is really focused on being reliable and speedy. This is a JavaScript based library and works on real time applications based on the web between web clients and servers [30].
There are two main components: a server side library for Node.js and a client side library that runs in the browser. These components have an almost similar API and it is event driven like Node.js. The WebSocket protocol is used by Socket.io alongside polling as their backup alternative and also provides the same user interface. Socket.io provides multiple features such as broadcasts to multiple sockets, data storage for every associated client, asynchronous I/O. It can also be used simply as a wrapper for WebSocket.

b. **Tensor Flow**

TensorFlow is an open source software library that helps for data flow and differential programming. This library is available to the public for free and allows for various tasks to be done. It is basically a math library and is widely used for machine learning projects including but not limited to that of neural networks. Google uses this extensively for research and production.

TensorFlow was developed for use internally in Google by their team Google Brain. This was further released on the 9th of November in 2015 under the Apache Licence 2.0 [31].

c. Numpy

The Python programming language has a library called NumPy which provides support for large and multi dimensional matrices and arrays. They also have a wide assortment of high level mathematical functions and operations which work with these arrays. Numeric, which is an ancestor of NumPy, has been created by Jim Hugunin. Multiple other developers also contributed to this cause. In 2005, features of Numeric were integrated with a competing library called Numarray. Alongisde, several modifications in this step by Travis Oliphant, resulted in the final creation of NumPy. NumPy currently has a lot of contributors and is an open source software [32].

d. Tensor Layer

TensorLayer is an extended library from Google TensorFlow. This is a Reinforcement Learning (RL) and Deep Learning (DL) library. A lot of popular Deep Learning and Reinforcement Learning modules are provided by TensorLayer. These are easily customisable and provides easy assembly to tackle real world based machine learning issues [27].

The flask socket server runs on port 5000 and is open to all the socket emit requests from the client site. The flask socket server uses the socket.on method of the socket, library to listen to the new messages.

Message Type	Message Format	Description
login	{'type': 'login', 'mobileNumber': '61452641992', 'password': 'sahaj', 'user_name': 'Sahaj'}	This message is sent to the server when a new client wants to login.
old_messages	{'type': 'old_messages', 'category': 'bookings', 'mobileNumber': '61452641992'}	This message is sent to the server when the client wants to query his existing messages from the server.
message	{'type': 'message', 'user_name': 'Sahaj', 'category': 'bookings', 'message': 'I would like to book a hotel for a cheap price', 'mobileNumber': '61452641992'}	This message is sent to the server when the user types in a new message in the mobile chat application.

Table 10: Format of messages received by the server

4.1.2. The server communication

The server communication is explained in detail below:

Handling Connection Request:

When a new client is connected and a "connection" request is sent to the server, the server accepts a request and opens a socket to the communicate with the client. Once the connection is established the client server communication starts.

Logging In:

When a client connects with the server it sends a "login" message to the server. This message is used for authenticating the client. Once the server receives this message, it queries the MongoDB database and authenticates the client. If the client is authenticated successfully, a success response is sent by the server to the client.

Old Private Messages:

When the server receives an "old_messages" request from the existing client, the server queries the MongoDB database to get all the messages of the particular client based on the unique mobile number of the user.

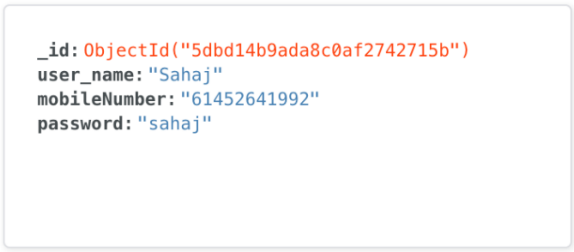

Figure 18: MongoDB users' schema

Server-side Message Communication:

If the client sends a message, the server determines the category of the message (bookings, support, general) and sends the message to the appropriate pre-trained model and waits for a response.

The server also simultaneously stores the message in the MongoDB database. Once the response is received from the model the server again saves this response in the MongoDB database and sends the message to the mobile client via the "socket. emit" method.

4.2. Implementation of the Mobile Application

4.2.1. Technology

The mobile client is developed using react native as the main technology with expo toolchain. The client also uses various libraries and SDKs that are listed below.

React Native

React Native allows the building of a mobile application with only JavaScript. The apps use the same UI blocks for building as that of a native Android or iOS application. Apps built with React Native are similar to apps built using Java, Swift or Objective C. However, React Native enables building a mobile app faster and cheaper. It is an open source framework [33].

Expo

Expo is a platform for making universally native apps for iOS, Android and the Web. These use React with JavaScript. Expo is open source [34].

Message Type	Message Format	Description
login	{'type': 'login', 'mobileNumber': '61452641992', 'password': 'sahaj', 'user_name': 'Sahaj'}	This message is sent to the client after logging in.
old_messages	[{"user_name":"Sahaj", "senderMobileNumber":"61452641992", "receiverMobileNumber":"chatbot", "category":"bookings","message": "I want to book a room for 1 night" ,"timestamp": "2019-11-02 06:46:58.390413"}, {"user_name":"Sahaj", "senderMobileNumber":"chatbot", "receiverMobileNumber":"61452641992", "category":"bookings","message": "What day would you like to book for?", "timestamp":"2019-11-02 06:47:01.360811"}]	This message is sent to the client which contains all the previous message of the client with the chatbot.
message	{"message": "I can book it for Tuesday ","user_name": "ChatBot"}	This is the chatbot response message that is sent from the server to the client.

Table 11: Format of messages received by the client

Handling Connection Response: Once a connection response is received by the client, the client becomes ready to send the messages to the server.

4.2.2. Client-Side Communication

Client-side Message Communication: When user types in a message, the message is sent to the server using the socket.emit method from the socket IO library. After the message has been sent, the client waits for the appropriate response. The format of the message sent to the server is same as the format of message sent by the server as is described in the above section. Once the client receives the message, it is displayed to the user.

4.2.3. Data Flow

Figure 19 shows the data flow of the application. This is described in more details in the following steps:-

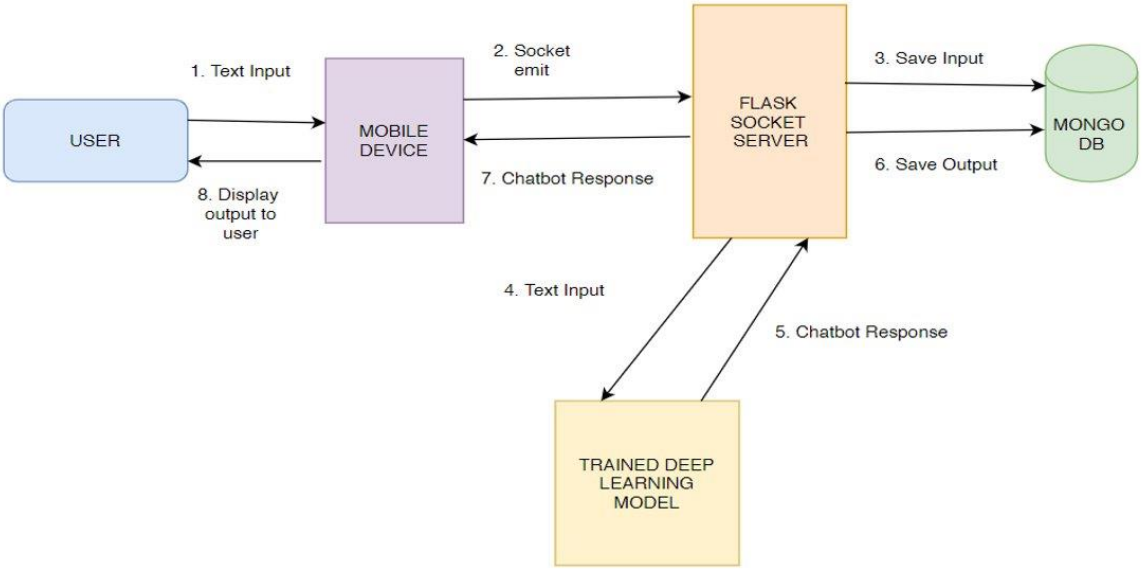

Figure 19: Data Flow of the Application

1. **Text Input:** The authenticated user can select the category from the list of 3 categories namely Travel and Bookings, Customer Support and General as displayed in the mobile application.

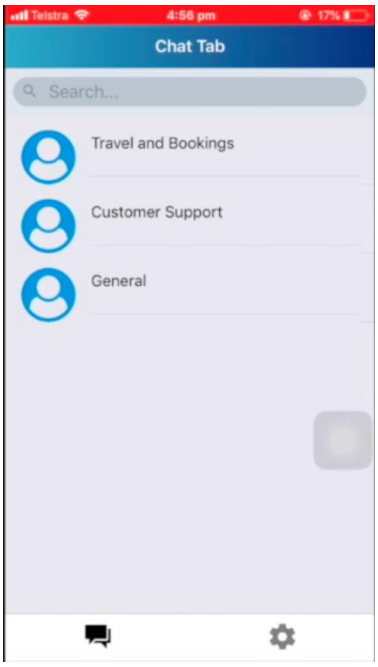

Figure 20: List of categories available for the user on the mobile application

Once the user selects the category, he or she can type in the text input on the app.

2. **Socket emit:** The mobile application sends this text input to the flask socket server using the socket.emit method of the SocketIO library [35].

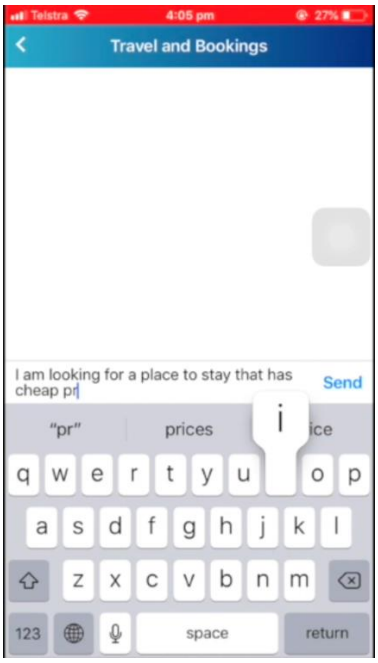

Figure 21: User typing input in Travel and Bookings category

3. **Save Input:** The flask socket server saves this input into the MongoDB database.

```
_id: ObjectId("5dbd2662cdc59db1c2443d14")
message: "I am looking for a place to stay that has
        cheap price."
timestamp: "2019-10-29 16:05:58.390413"
user_name: "Sahaj"
senderMobileNumber: "61452641992"
receiverMobileNumber: "chatbot"
category: "bookings"
```

Figure 22: MongoDB database schema for storing the messages

4. **Text Input to trained Model:** The text input is also sent to the brain of the system that is trained deep learning model.
5. **Chatbot Response:** The trained deep learning model generates an appropriate response based on the category and sends it back to the flask socket server.
6. **Save Chatbot Response:** The Flask Socket server saves this response to the MongoDB database.
7. **Chatbot Response to Mobile Device:** The flask socket server sends the chatbot response to the mobile device.

8. **Display Output to User:** This chatbot output is finally displayed to the user.

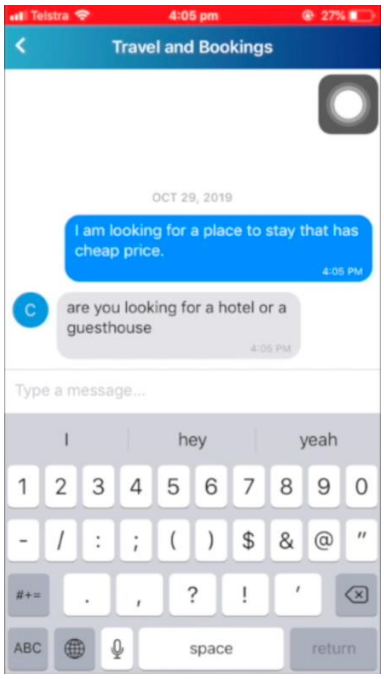

Figure 23: Chatbot Response based on the Travel and Booking Category

5. Challenges and Limitations

Building a chatbot comes with its own set of challenges and limitations which are mentioned below :

5.1. Limited training time

The project encompassed one semester which was enough time to train our model. However in the case of seasoned models, usually data collection takes the longest and is done specific to the domain of the chatbot [22].

5.2. Scope of Testing

Testing a chatbot depends of various factors such as the chatbot domain, its targeted users, type of chat (casual or business), rule based or AI based. With these possibilities, there are multiple combinations to think of for testing. There is no straightforward test case for a chatbot performance [36].

5.3. Free form of text

There is no control over the input which the user may enter. Accordingly the chatbot must still respond [37].

5.4. API integration

In case of third party API integration for categories such as bookings it is imperative to test if the bookings work. Moreover, the bookings must be correct and for the particular user [36].

5.5. UI Elements

The entire app UI needs to be self explanatory such that the human can converse without much guidance. Elements such as personality, grammar and images also need to be taken care of. If the response rate is slow, typing elements such as a loading or typing symbol need to be present [36].

5.6. Lack of context

The chatbot is unable to understand the query within a context. The queries need to be explicit. Another significant limitation is based on what dialects and language the chatbot is trained in. Our chatbot currently performs in English and does not account for slangs or acronyms [37].

5.7. Lack of emotions

Chatbots are unable to think and feel like humans even though they can try to converse like one. This lack of empathy may cause the conversation to fall short in particular scenarios [38].

6. Future Work

6.1. Improved processing time

As per the suggestions received, we would process our model on TPU instead of GPU for faster training time. This will help make our model and save on precious time just waiting for the model to train before testing it [39].

6.2. Model building on Transformer

Due to limited knowledge and lack of capabilities around this area, we decided to build our model on Tensorflow and Tensorlayer. Transformer model like BERT is known to give better results. Maybe in the future we could do a comparison on our existing encoder-decoder model and working transformer model like BERT [40].

6.3. Better programming and datasets

In the future, it is essential to get better quality data which has been collected for this sole purpose to get better results. Also use newer technologies that come up in the area. There are are some commonly available datasets which are currently free [41] and they suit their domain. However for better results, a dedicated data collection process needs to be carried out.

6.4. Integration of various models

We have currently used Seq2Seq model for training our database. We may need to integrate multiple models to get a better performing chatbot. A potential combination which has generated better results is a combination of rule based and artificial intelligence based models [42].

6.5. Human involvement

A lot of AI powered chatbots allow a human resource to take charge of the conversation if the confidence level of the bots falls below a particular threshold [43]. They can also take over if the bot fails to respond.

Some organisations have integration solutions where after a pre determined touchpoint the human takes over the conversation and leads from there. Thus saving valuable time and improving the quality of conversation as well.

Highly automated bots can also fail to understand a user's issue. At these times, to guarantee the continuous and proficient rendering of client support, it is better to have a fall back option in the sense of live chat agents or customer specialists. These humans can flawlessly assume control over the discussion utilizing the instrument of the chatbot.

Every enterprise should endeavour towards automation as their goal. However the human component stays vital and cannot be ignored. An advanced version of a chatbot with its AI component, can assist clients across multitude of services [43]. In such cases having a human agent or operator on the side will make the service 100% foolproof. There are some conversations that are best dealt with a human and in these cases even the most advanced chatbots could use some help.

6.6. Integration of speech recognition

A lot of the chat applications have now integrated speech recognition. They allow text based chatting alongside voice based chatting. A future work would definitely involve more along this line [11].

7. Conclusion

It is very easy to build a simple chatbot using Node.js and Socket.io [44]. However for a great app, effort and money needs to be spent in terms of developing a quality UI and collecting data.

A chatbot's ability to comprehend and react to client's inquiries make them a trendy and a ground-breaking application. This has urged the organisations to incorporate chatbots as an essential part of their operations.

Chatbots help to boost the operational competency of the organisations compressing the use of resources in backend support. Organisations can enhance their human resources by utilizing this chatbot advancements.

Health, banking, e-commerce and insurance are some areas where the solid presence of chatbots can be expected, as companies seem to favour them [45] [46] [47]. Cost savings, a lean team to undertake the complaints of the clients are some benefit. Companies will be in the position to shift the manpower for more effective activities. As of now, there are various regions where chatbots are amazingly useful – and will continue for more.

Companies all over the world are taking efforts to come up with new and different ideas and also taking initiatives to convert the working environment more beneficial. Chatbots can be of great help here too. Mechanisation can give rise to a lot of free time to increase the efficiency or give allowance to people to devote that spare time for other imaginative work.

With growing and progressing in AI technology, chatbots will soon become an essential part in company's operations and activities. The rundown of increasing benefits that chatbots has for companies is undoubtedly to develop in the coming years and can also be calibrated when need emerges. There is a little uncertainty that the number of complex activities that will be managed and undertaken with the help of chatbots will also increase exponentially in the later

period. There will come a time when it will be difficult to know if you are talking to a chatbot or a fellow human being [48].

8. Bibliography

[1] S. Moore, “Gartner Says 25 Percent of Customer Service Operations Will Use Virtual Customer Assistants by 2020,” Gartner, 2018. [Online]. Available: https://www.gartner.com/en/newsroom/press-releases/2018-02-19-gartner-says-25-percent-of-customer-service-operations-will-use-virtual-customer-assistants-by-2020. [Accessed: 05-Nov-2019].

[2] D. McGinnis, “What Is a Chatbot and How Is It Changing Customer Experience?,” Salesforce Blog, 2019. [Online]. Available: https://www.salesforce.com/blog/2019/04/what-is-a-chatbot.html. [Accessed: 04-Nov-2019].

[3] Kommunicate, “The Complete Beginner’s Guide to Chatbots.”

[4] Amit Dua, “Rule-Based Bots or AI Bots? Which One Do You Prefer? - Dataconomy,” Dataconomy, 29-May-2019. [Online]. Available: https://dataconomy.com/2019/05/rule-based-bots-or-ai-bots-which-one-do-you-prefer/. [Accessed: 04-Nov-2019].

[5] G. Hald, “Chatbot 101: Everything you ever wanted to know about Chatbots,” Medium, 15-Nov-2017. [Online]. Available: https://medium.com/botsupply/chatbot-101-everything-you-ever-wanted-to-know-about-chatbots-478c0b825dd0.

[6] J. Jia, “CSIEC: A computer assisted English learning chatbot based on textual knowledge and reasoning,” Knowledge-Based Systems, vol. 22, no. 4, pp. 249–255, May 2009.

[7] L. Goasduff, “Smarter With Gartner,” Gartner.com, 31-Jul-2019. [Online]. Available: https://www.gartner.com/smarterwithgartner/chatbots-will-appeal-to-modern-workers/. [Accessed: 05-Nov-2019].

[8] R. Harman, “Chatbot and Customer Service; How Technology Is Helping To Increase Engagement,” Medium, 14-May-2019. [Online]. Available: https://chatbotsmagazine.com/chatbot-and-customer-service-how-technology-is-helping-to-increase-engagement-91c1bcb03c61. [Accessed: 05-Nov-2019].

[9] M. Mason, “3 types of business chatbots you can build - Watson,” Watson, 12-Dec-2017. [Online]. Available: https://www.ibm.com/blogs/watson/2017/12/3-types-of-business-chatbots-you-can-build/. [Accessed: 04-Nov-2019].

[10] Nilima Shah, “Which Is Best for You: Rule-Based Bots or AI Bots?,” Medium, 02-Feb-2017. [Online]. Available: https://chatbotsmagazine.com/which-is-best-for-you-rule-based-bots-or-ai-bots-298b9106c81d. [Accessed: 04-Nov-2019].

[11] S. Abdul-Kader and J. Woods, “Survey on Chatbot Design Techniques in Speech Conversation Systems - Research Repository,” Essex.ac.uk, vol. 6, no. 7, 2015.

[12] Shreya Ghelani, “Text Classification—RNN’s or CNN’s?,” Medium, 02-Jun-2019. [Online]. Available: https://towardsdatascience.com/text-classification-rnns-or-cnn-s-98c86a0dd361. [Accessed: 04-Nov-2019].

[13] M. Schlicht, “The Complete Beginner’s Guide To Chatbots,” Chatbots Magazine, 20-Apr-2016. [Online]. Available: https://chatbotsmagazine.com/the-complete-beginner-s-guide-to-chatbots-8280b7b906ca. [Accessed: 04-May-2019].

[14] alexa, “alexa/alexa-prize-topical-chat-dataset,” GitHub, 2019. [Online]. Available: https://github.com/alexa/alexa-prize-topical-chat-dataset/tree/master/conversations.

[15] P. Budzianowski et al., “Research data supporting ‘MultiWOZ - A Large-Scale Multi-Domain Wizard-of-Oz Dataset for Task-Oriented Dialogue Modelling,’” Cam.ac.uk, 2019.

[16] Thought Vector, “Customer Support on Twitter,” Kaggle.com, 2017. [Online]. Available: https://www.kaggle.com/thoughtvector/customer-support-on-twitter.

[17] “Natural Language Toolkit — NLTK 3.4.4 documentation,” Nltk.org, 2009. [Online]. Available: https://www.nltk.org/.

[18] D. Braun, A. Hernandez-Mendez, F. Matthes, and M. Langen, “Evaluating Natural Language Understanding Services for Conversational Question Answering Systems,” Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue, 2017.

[19] Analytics Vidhya, “Understanding Word Embeddings: From Word2Vec to Count Vectors,” Analytics Vidhya, 04-Jun-2017. [Online]. Available: https://www.analyticsvidhya.com/blog/2017/06/word-embeddings-count-word2veec/. [Accessed: 04-Nov-2019].

[20] “seq2seq (Sequence to Sequence) Model for Deep Learning with PyTorch,” Guru99.com, 07-Oct-2019. [Online]. Available: https://www.guru99.com/seq2seq-model.html. [Accessed: 04-Nov-2019].

[21] Serban, Iulian V et al., “A Deep Reinforcement Learning Chatbot,” arXiv.org, 2017. [Online]. Available: https://arxiv.org/abs/1709.02349. [Accessed: 04-Nov-2019].

[22] Unimelb.edu.au, 2019. [Online]. Available: https://learning-oreilly-com.ezp.lib.unimelb.edu.au/library/view/recurrent-neural-networks/9781789132335/7da927a6-293b-4715-b36e-06251c339214.xhtml.

[23] T. Mikolov, M. Karafiát, Lukás Burget, J. Černocký, and Sanjeev Khudanpur, “Recurrent neural network based language model,” undefined, 2010. [Online]. Available: https://www.semanticscholar.org/paper/Recurrent-neural-network-based-language-model-

[24] https://www.facebook.com/jason.brownlee.39, “Gentle Introduction to the Adam Optimization Algorithm for Deep Learning,” Machine Learning Mastery, 02-Jul-2017. [Online]. Available: https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/. [Accessed: 15-Oct-2019].

[25] Renu Khandelwal, “Overview of different Optimizers for neural networks,” Medium, 03-Feb-2019. [Online]. Available: https://medium.com/datadriveninvestor/overview-of-different-optimizers-for-neural-networks-e0ed119440c3. [Accessed: 05-Nov-2019].

[26] Suriyadeepan Ramamoorthy, “Practical seq2seq,” Complx.me, 2016. [Online]. Available: http://complx.me/2016-12-31-practical-seq2seq/. [Accessed: 04-Nov-2019].

[27] “Welcome to TensorLayer — TensorLayer 2.2.0 documentation,” Readthedocs.io, 2019. [Online]. Available: https://tensorlayer.readthedocs.io/en/latest/. [Accessed: 04-Nov-2019].

[28] G. D. Németh, “BLEU-BERT-y: Comparing sentence scores,” Medium, 14-Oct-2019. [Online]. Available: https://towardsdatascience.com/bleu-bert-y-comparing-sentence-scores-307e0975994d. [Accessed: 04-Nov-2019].
Mikolov-Karafi%C3%A1t/9819b600a828a57e1cde047bbe710d3446b30da5. [Accessed: 04-Nov-2019].

[29] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, “Bleu: a Method for Automatic Evaluation of Machine Translation Bleu: a Method for Automatic Evaluation of Machine Translation,” 2001.

[30] “Socket.IO,” Socket.IO, 20-Oct-2019. [Online]. Available: https://socket.io/.

[31] “TensorFlow,” TensorFlow, 2019. [Online]. Available: https://www.tensorflow.org/.

[32] “NumPy — NumPy,” Numpy.org, 2009. [Online]. Available: https://numpy.org/.

[33] “React Native · A framework for building native apps using React,” Github.io, 2019. [Online]. Available: https://facebook.github.io/react-native/.

[34] “Expo,” Expo, 2019. [Online]. Available: https://expo.io/. [Accessed: 04-Nov-2019].

[35] M. Makai, “flask redirect Example Python Code,” Fullstackpython.com, 2014. [Online]. Available: https://www.fullstackpython.com/flask-redirect-examples.html. [Accessed: 05-Nov-2019].

[36] https://www.facebook.com/lajari.patil, “Haptik Tech Blog,” Haptik Tech Blog, 15-Mar-2019. [Online]. Available: https://haptik.ai/tech/automating-bot-testing/. [Accessed: 05-Nov-2019].

[37] J. Zamora, “I’m Sorry, Dave, I’m Afraid I Can’t Do That: Chatbot Perception and Expectations,” Google AI, 2017. [Online]. Available: https://ai.google/research/pubs/pub46429. [Accessed: 04-Nov-2019].

[38] D. Thomas, “Chatbots Sound Great, But are They Really Useful for Businesses?,” Medium, 14-May-2019. [Online]. Available: https://chatbotsmagazine.com/chatbots-sound-great-but-are-they-really-useful-for-businesses-ec46611561dd. [Accessed: 05-Nov-2019].

[39] Ambika Choudhury, “TPU Vs GPU Vs CPU: Which Hardware Should You Choose For Deep Learning,” Analytics India Magazine, 08-Aug-2019. [Online]. Available: https://analyticsindiamag.com/tpu-vs-gpu-vs-cpu-which-hardware-should-you-choose-for-deep-learning/. [Accessed: 05-Nov-2019].

[40] J. Devlin, M.-W. Chang, K. Lee, K. Google, and A. Language, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” 2019.

[41] A. Nguyen, “15 Best Chatbot Datasets for Machine Learning | Lionbridge AI,” Lionbridge AI, 02-Jul-2019. [Online]. Available: https://lionbridge.ai/datasets/15-best-chatbot-datasets-for-machine-learning/. [Accessed: 05-Nov-2019].

[42] An AI-powered, “Which is a better option, an AI-powered or a Rule-based chatbot?,” swivl, 02-Jun-2019. [Online]. Available: https://tryswivl.com/blog/conversational-interface/which-is-the-best-option-rule-based-or-ai-powered-chatbots/. [Accessed: 05-Nov-2019].

[43] Haptik IO, “Live Chat Agent: The Human Element in Conversational AI - Haptik Blog,” Haptik Blog, 28-May-2019. [Online]. Available: https://haptik.ai/blog/live-chat-agent-conversational-ai/. [Accessed: 05-Nov-2019].

[44] Noufel Gouirhate, “Build a simple chat app with node.js and socket.io,” Medium, 24-Dec-2017. [Online]. Available: https://medium.com/@noufel.gouirhate/build-a-simple-chat-app-with-node-js-and-socket-io-ea716c093088. [Accessed: 05-Nov-2019].

[45] L. Wood, “The Global Chatbot Market is Forecast to Reach $5.63 Billion by 2023 -- Asia Pacific to Witness the Highest Growth - ResearchAndMarkets.com,” Businesswire.com, 14-Mar-2019. [Online]. Available: https://www.businesswire.com/news/home/20190314005364/en/Global-Chatbot-Market-Forecast-Reach-5.63-Billion.

[46] J. Young, “Global ecommerce sales grow 18% in 2018 | Digital Commerce 360,” Digital Commerce 360, 2018. [Online]. Available: https://www.digitalcommerce360.com/article/global-ecommerce-sales/. [Accessed: 05-Nov-2019].

[47] Business Insider Intelligence, “80% of businesses want chatbots by 2020 - Business Insider,” Business Insider, 14-Dec-2016. [Online]. Available: https://www.businessinsider.com/80-of-businesses-want-chatbots-by-2020-2016-12/?r=AU&IR=T. [Accessed: 05-Nov-2019].

[48] B. Brn.AI, “Chatbot 2018 — Is It My Competitor or Colleague?,” Medium, 24-Jan-2018. [Online]. Available: https://chatbotsmagazine.com/chatbot-2018-is-it-my-competitor-or-colleague-8e98ba8cd473. [Accessed: 05-Nov-2019].

9. Appendix

Source Code

https://github.com/sahajdhingra26/Computing-Project-Mobile-Chatting-AutoBot

Video Demo

https://youtu.be/EmqE8xKwBe0

Screenshots

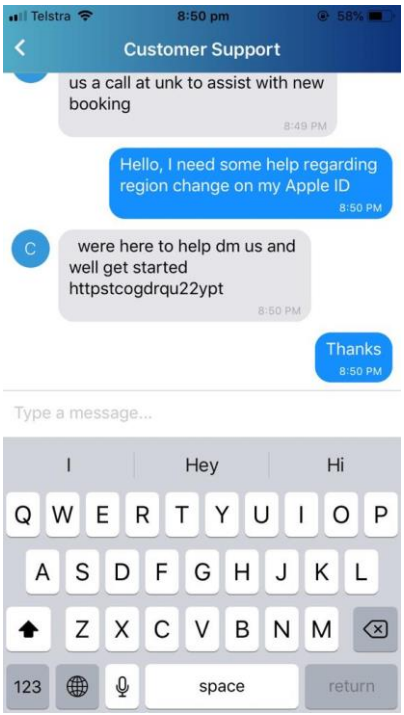

Figure 24: User chatting with Customer Support chatbot

Figure 25: User chatting by selecting the normal category.

Instructions to run

Mobile App

Step 1:

Android App can be directly run on the mobile device by downloading the apk from the apk folder and running it on the android mobile device.

Step 2:

To compile the code on the system and run the android or ios app on the device, please clone the repository.

Step 3:

Install all the necessary packages on the system navigate to the client folder by doing cd client and then by running the command npm install. This will create a node_modules folder in the root directory.

Step 4:

Update the assets/baseUrl file with the server url and then execute the command expo start to start the expo server on the system.

Step 5:

To run the app on the mobile, install the Expo Client app from Play Store for android Click here to download Expo Client on android or App Store for iOS Click here to download Expo Client on iOS.

Step 6:

Open the expo client app on the phone and scan the QR code that was generated in step 4. This will compile the app and run it on your device.

Training the Model

Step 1:

The pretrained model can be used. Its uploaded in the server folder for all 3 categories. For Travel and Bookings the file bookings_model.npz should be used.
For Customer Support the file support_model.npz should be used.
For General the file normal_model.npz should be used.

Step 2:

The model can be trained by navigating to the training folder by executing the command cd training/ and then installing all the necessary python packages by running the command pip3 install -r requirements.txt .

Step 3:

The model will start training by running the command python3 main.py

PLEASE NOTE: THE MODEL SHOULD ONLY BE TRAINED ON A MACHINE WITH GPU FOR BEST PERFORMANCES. GOOGLE CLOUD PLATFORM GPU ENABLED MACHINE WAS USED FOR TRAINING OF THIS MODEL

Running the server

Step 1:

The server can be run by navigating to the server folder by executing the command cd server and installing the flask by executing pip3 install flask then installing all the necessary python packages by running the command pip3 install -r requirements.txt Also, in this case another package called as PyMongo has to be installed by executing the command pip3 install pymongo

Step 2:

The database can be created by creating an account on Mlab and upating the MONGO_URI inside the server/main.py file.

Step 3: Run the server by executing python3 main.py