IT7611-Creative and Innovative Project

Conversational AI Chatbot

Submitted by

Karthikeyan JM (2018506049)

Dinesh V (2018506028)

Abinaya A (2018506003)

MADRAS INSTITUTE OF TECHNOLOGY

ANNA UNIVERSITY, CHENNAI-600 044.

Submitted to

Dr.M.R. Sumalatha

Dr.P. Lakshmi Harika

DEPARTMENT OF INFORMATION TECHNOLOGY

ANNA UNIVERSITY, CHENNAI 600044

BONAFIDE CERTIFICATE

Certified that this project Report titled "**Conversational AI Chatbot"** is the bonafide work of **Karthikeyan JM (**RegNo.:**2018506049), Dinesh V (**RegNo.:**2018506028), Abinaya A(**RegNo: **2018506003)** who carried out the project work under my supervision. Certified further that to the best of my knowledge the work reported herein does not form part or full of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this to any other candidate.

Submitted for the examination held on 14. 06.2021

SIGNATURE
Dr.M.R. SUMALATHA,
Dr.P. LAKSHMI HARIKA
SUPERVISOR
Department of Information Technology
MIT Campus, Anna University
Chennai - 600 044

ACKNOWLEDGEMENT

It is essential to mention the names of the people, whose guidance and encouragement helped us to complete this project.

We express our gratitude to our project guide, **Dr. M.R. SUMALATHA,** Assistant Professor (Sl grade), Department of Information Technology, Anna University, for guiding and assisting us throughout this project.

Our sincere thanks to **Dr. Dhananjay Kumar**, Head of the Department of Information Technology, MIT Campus, for catering to all our needs and supporting us throughout this project.

We express our gratitude to our respected Dean of MIT Campus, **Dr.T.Thyagarajan**, for providing excellent computing facilities to complete this project.

Karthikeyan JM 2018506049
Abinaya A 2018506003
Dinesh V   2018506028

TABLE OF CONTENTS

ABSTRACT

Conversational AI chatbot using deep learning is an interesting problem in the natural language processing. A chatbot is a software program designed to simulate conversation with human users, especially over the Internet. Chatbots provide the assistance to the required information quickly and efficiently. the first chat bot was made by Joseph Weizenbaum. Each of these commands should be written by the developer separately using string analysis. Smart chatbots rely on AI when they are communicating with users. Instead of pre-prepared answers, the bot responds with best possible suggestions on the topic. In the past, chatbot architectures was constructed based on hand-written rules and templates or simple statistical methods. With the rise of deep learning, the chatbot architecture were replaced by end-to-end trainable neural networks. More specifically, there current encoder-decoder model dominates the task of conversational modelling. Among current chatbots, many are developed using rule-based architectures, simple machine learning algorithms or retrieval-based techniques which generates bad results. In this project, we have developed aSeq2Seq AI Chatbot using modern-day techniques. For developing Seq2Seq Model, we have implemented attention mechanism architecture. This encoder-decoder model uses Recurrent Neural Network with LSTM (Long-Short-Term-Memory) cells. These conversation agents are mostly used by businesses, government organizations and non-profit organizations. They are also frequently used by financial organizations like bank, credit card companies, businesses like online retail stores and start-ups. Their functioning types can range from customer service to personal assistant. Recently, there has been a major increase in interest in the use and deployment of conversational chatbot systems. Though they are primarily been question-answer systems, their adoption by major corporations has peaked interesting customers and looks promising for more advanced conversational agent system in research and development.

LIST OF TABLES

LIST OF ABBREVIATION

CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory cells
BRNN	Bidirectional Recurrent Neural Network
NLP	Natural Language Processing
Seq2Seq	Sequence to Sequence
SMT	Statistical Machine Translation

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

Artificial Intelligence integrates our daily lives with the creation and analysis of intelligent software and hardware, called intelligent models. Intelligent models can do a variety of tasks from labor work to sophisticated operations. A chatbot is a good example of an Artificial Intelligence system and one of the most elementary examples of intelligent Human-Computer Interaction. It is a software program, which responds like a smart entity when conversed with through text or voice and understands human languages easily by Natural Language Processing (NLP). A chatbot is defined as "A computer program designed to simulate conversation with human users, especially over the Internet". Chatbots are also called as smart bots, interactive agents, digital assistants, or artificial conversation entities.

Important to mention is that chatbots lacks understanding the meaning and that they are not as capable as humans of understanding conversational undertones. Though progress has been made in this field, machines will not understand what is the feeling of what he is saying.

Chatbots can be categorized into different parameters: the knowledge domain, the service provided, the goals, the input processing and response generation method, the human-aid, and 21the build method.

Among current chatbots, many are developed using rule-based architectures, simple machine learning algorithms or retrieval-based techniques which do not generate good results. In this project, we have developed aSeq2Seq AI Chatbot using modern-day techniques. For developing Seq2Seq AI Chatbot, we have implemented encoder-decoder model with attention mechanism architecture. This encoder-decoder model uses Recurrent Neural Network with LSTM (Long-Short-Term-Memory) cells.

An Encoder-Decoder model architecture was developed where an input sequence was read and encoded to a fixed-length internal representation of vectors. Then a decoder network used this internal representation to output words until the end of sequence token was reached.

In RNNs, the recurrent layers (or) hidden layers consist of recurrent cells whose states were both affected by past states and current input with feedback connections. The recurrent layers can also be organized in various architectures to form different RNNs. Therefore, RNNs are mainly distinguished by the recurrent cell and network architecture. Different cells and inner connections enable RNNs to possess different capacities. In order to explore the development of LSTM networks, this section first gives a brief review of the LSTM cell and its variants.

1.2 CHALLENGES:

The main challenges in the chatbot are:

1.Chatbot Security

2. Understanding the Emotional and Sentiment of the user.

1.Chatbot Security

In recent years, a lot happened regarding data privacy and user's security. Users are very sensitive and demands high level security when it comes to their personal data. Hence, it's crucial that you create chatbots which can assure data privacy for users.AI chatbots need to collect only information and data which are relevant and necessary to transmit it over the internet securely. If you are going to use chatbots, then you must need to absolutely make sure that it's safe to share information with the chatbots.

Additionally, you need to make sure that the chatbot is hack-proof and no hacker can get access to your chats data. That's because user's data is sensitive and could be easily misused or mishandled, and it can destroy your company's reputation.

2.Understanding the emotional and Sentiment of the user

We can also design conversational AI platforms with Voice enabled Bots. They can be able to recognize human emotions. If they misinterpret human emotion, it can have a huge negative impact on your business. Having said that, it's really challenging to identify the emotion from the user's voice and respond to it accordingly. To overcome this issue and create the best AI chatbot, need a lot of time into training. This way, it can easily identify the correct emotions of a human voice and respond to it in the right tone.

1.3 PROBLEM STATEMENT

Among current chatbots, many are developed using rule-based architectures, simple machine learning algorithms or retrieval-based techniques which do not generate good results. Especially when dealing with real world data the sentences may belong enough where it may lead to the failure of the existing models. As a solution to this problem we propose Seq2Seq model with attention mechanism using modern-day techniques. For developing Seq2Seq AI Chatbot, we propose a encoder-decoder attention mechanism architecture. This encoder-decoder will be using Recurrent Neural Network with LSTM (Long-Short-Term-Memory) cells.

1.4 OBJECTIVES:

The main objective of chatbot framework is specially designed to interact, talk, listen, and communicate with your customers. There are many objectives for chatbots, depending on the sector that you work in. The main purpose of chatbots is to support business teams in their relations with Users(customers), by offering precision, personalization, efficiency and scalability. And the objective of the Chatbots is meant to help and deliver immediate actions where humans can't reach due to timing or budget. To engage with users where the users are hanging out instead of asking users to download an app or visit a website to engage with your service/business. Currently, one of the main potentials of chatbots is their ease of use and that they' are platform agnostic, meaning that they can be deployed in whatever platform the target group to which they're directed prefers. It also means that they can be used internally for companies that are interested in upgrading their internal communications with a chatbot. Depending on the use case of the AI chatbots, they are often used to find information about the services a company provides, consult their products availability, make reservations or bookings, and even assess the customer's experience with the service the company has provided.

1.5 SCOPE:

Among current chatbots, many are developed using rule-based techniques, simple machine learning algorithms or retrieval-based architectures which do not generate good results. Especially when dealing with real world data the sentences may belong enough where it may lead to the failure of the existing models. To overcome these limitations, we try to develop a generative based conversational chatbot using Seq2Seq with Attention mechanism for society and enterprise applications. We propose Seq2Seq model with attention mechanism using modern-day techniques. For developing Seq2Seq AI Chatbot, we propose a encoder-decoder attention mechanism architecture. This encoder-decoder will be using Recurrent Neural Network with LSTM (Long-Short-Term-Memory) cells.

1.6 ATTENTION MECHANISM:

The main novel motivational idea is to use attention mechanism to improve model performance by greeting the encoder-decoder architecture from the fixed-length internal representation which is achieved by keeping the intermediate outputs from the encoder LSTM model from each steps of the input sequence and training the model to learn to pay selective attention to these inputs and relate them to items in the output sequence. Each time the proposed model generates a word while translation, it searches for a set of positions in a source sentence where the most relevant information is concentrated. The model then predicts a target word based on the context vectors and all the previously predicted target words. It encodes the input sentence into fixed-length vectors and chooses a subset of these vectors adaptively while decoding the translation. This relieves a neural AI conversation translation model from having to squash all the information of a source sentence, into a fixed-length vector.

1.7 Seq2Seq Model:

Sequence to Sequence model become the popular model for conversational Systems and Machine Translation. It consists of an Encoder and a Decoder. The encoder takes a sequence of information as input and processes one word at each time step. Its objective is to convert a sequence of information(words) into a fixed size feature vector that encodes only the important information in the sequence while losing the unnecessary information. You can visualize the data flow graph in the encoder along the time axis, as the flow of local information from one end of the sequence to another.

1.8 Organization of Report:

The organization of report is as follows:

Chapter 1 gives a brief introduction about Seq2Seq Chatbot. Chapter 2 analyses and gets various notions from international conference and journal papers. Chapter 3 describes modules to be implemented, the system architecture and system design of the proposed work. Chapter 4 describes the execution details of the modules proposed and screenshots of modules. Chapter5 presents the evaluation results. Chapter 6 combines conclusion of project and future enhancements that can be deployed in existing work. Finally, a reference section is stacked where details of various papers related to the project are added. The development environment details are mentioned in appendix I. The output of the system is inserted as snapshots under the topic appendix II.

CHAPTER 2

LITERATURE SURVEY

2.1 CONVERSATIONAL MODELS

1. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation:

Literature search was carried out using Google scholar, PubMed, Research gate, arXiv, IEEE journals, Elsevier journals and Scihub. The literature review in 2014, K. Cho, B. van Merrienboer, et.al described "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation"[1]. The authors proposed a novel neural network architecture that can be used as a part of the conventional phrase-based SMT system called RNN Encoder-Decoder consisting of two recurrent neural networks. They qualitatively analyzed the trained RNN Encoder–Decoder by comparing its phrase scores with those given by the existing translation model. The qualitative analysis shows that the RNN Encoder–Decoder is better at capturing the linguistic regularities in the phrase table, indirectly explaining the quantitative improvements in the overall translation performance. The qualitative analyzation the trained RNN Encoder–Decoder by comparing its phrase scores with those given by the existing translation model. The qualitative analysis shows that the RNN Encoder–Decoder is better at capturing the linguistic regularities in the phrase table, indirectly explaining the quantitative improvements in the overall translation performance. The further analysis of the model reveals that the RNN Encoder–Decoder learns a continuous space representation of a phrase that preserves both the semantic and syntactic structure of the phrase.The BLEU score and training accuracy is low when compared to other models. The goal of this literature review aims at better capturing the linguistic regularities using RNN Encoder-Decoder model.

2. Neural Machine Translation by Jointly Learning to Align and Translate:

Literature search was carried out using Google scholar, PubMed, Research gate, arXiv, IEEE journals, Elsevier journals and Scihub. The literature review in 2014, D. Bahdanau, K. Cho, and Y. Bengio described "Neural Machine Translation by Jointly Learning to Align and Translate"[2]. The authors showed that their proposed approach of jointly learning to align and translate achieves significantly improved translation performance over the basic encoder-decoder approach. The improvement is more apparent with longer sentences, but can be observed with sentences of any length. On the task of English-to-French translation, their proposed approach achieves, with a single model, a translation performance comparable, or close, to the conventional phrase-based system. Furthermore, qualitative analysis reveals that their proposed model finds a linguistically plausible (soft-)alignment between a source sentence and the corresponding target sentence. The proposed RNN search outperforms the conventional encoder–decoder model (RNN encdec) significantly, regardless of the sentence length and that it is much more robust to the length of a source sentence. The qualitative analysis concludes that the model can correctly align each target word with the relevant words, or their annotations, in the source sentence as it generated a correct translation. Perhaps more importantly, the proposed approach achieved a translation performance comparable to the existing phrase-based statistical machine translation. The proposed approach is not good at handling unknown or rare words. Also, it doesn't match the performance of current state-of-the-art machine translation systems. The goal of this literature review aims at better linguistically plausible (soft-)alignment between a source sentence and the corresponding target sentence with their proposed model.

3. Sequence to Sequence Learning with Neural Networks:

Literature search was carried out using Google scholar, PubMed, Research gate, arXiv, IEEE journals, Elsevier journals and Scihub. The literature review in 2014, Sutskever, O. Vinyals, and Q. Le. CoRR 3 described "Sequence to Sequence Learning with Neural Networks"[3]. The authors present a general end-to-end approach to sequence learning that makes minimal assumptions on the sequence structure. They used a multilayered Long Short-TermMemory (LSTM) to map the input sequence to a vector of a fixed dimensionality, and then another deep LSTM to decode the target sequence from the vector. They obtained a BLEU score of 34.81 by directly extracting translations from an ensemble of 5 deep LSTMs (with 384M parameters and 8,000 dimensional state each) using a simple left-toright beam-search decoder on WMT'14 English to French translation task. This is by far the best result achieved by direct translation with large neural networks. For comparison, the BLEU score of an SMT baseline on this dataset is 33.30. The 34.81 BLEU score was achieved by an LSTM with a vocabulary of 80k words, so the score was penalized whenever the reference translation contained a word not covered by these 80k. Their result shows that a relatively unoptimized small-vocabulary neural network architecture which has much room for improvement outperforms a phrase-based SMT system. Finally, they used the LSTM to rescore the publicly available 1000-best lists of the SMT baseline on the same task. By doing so, they obtained a BLEU score of 36.5, which improves the baseline by 3.2 BLEU points and is close to the previous best published result on this task (which is 37.0). Surprisingly, the LSTM did not suffer on very long sentences, despite the recent experience of other researchers with related architectures. The authors were able to do well on long sentences because they reversed the order of words in the source sentence but not the target sentences in the training and test set. They introduced many short term dependencies that made the optimization problem much simpler. As a result, SGD could learn LSTMs that had no trouble with long sentences. Their simple trick of reversing the words in the source sentence is one of the key technical contributions of their work. The proposed LSTM-based approach does well on sequence learning problems. It correctly translates very long sentences. It also outperforms SMT systems. This approach has only a limited memory and it can't perform well if it doesn't have enough training data.The goal of this literature review aims at developing a better model for translation between a source sentence and the corresponding target sentence.

4. A Neural Conversational Model:

Literature search was carried out using Google scholar, PubMed, Research gate, arXiv, IEEE journals, Elsevier journals and Scihub. The literature review in 2015, O. Vinyals, and Q. Le described "A Neural Conversational Model"[4].The authors found that Conversational modeling approach did surprisingly well on generating fluent and accurate replies to conversations. They tested the model on chat sessions from an IT helpdesk dataset of conversations, and found that their model can sometimes track the problem and provide a useful answer to the user. They also experimented with conversations obtained from a noisy dataset of movie subtitles, and found that their model can hold a natural conversation and sometimes perform simple forms of common-sense reasoning. In both cases, the recurrent nets obtain better perplexity compared to the n-gram model and capture important long-range correlations. From a qualitative point of view, their model is sometimes able to produce natural conversations. This purely data driven approach without any rules can produce rather proper answers to many types of questions. This approach shows that even a simple language model based on the seq2seq framework can be used to train a conversational engine. The model doesn't deliver any realistic conversations. The lack of a coherent personality makes it difficult for the system to pass the Turing test.The goal of this literature review aims at developing a better model for producing natural conversations.

5. Neural Network Approach to Context-Sensitive Generation of Conversational Responses:

Literature search was carried out using Google scholar, PubMed, Research gate, arXiv, IEEE journals, Elsevier journals and Scihub. The literature review in 2015, A. Sordoni, M. Galley, et.al described "Neural Network Approach to Context-Sensitive Generation of Conversational Responses"[5].The authors proposed to address the challenge of context-sensitive response generation by using continuous representations or embeddings of words and phrases to compactly encode semantic and syntactic similarity. They argued that embedding-based models afford flexibility to model the transitions between consecutive utterances and to capture long-span dependencies in a domain where traditional word and phrase alignment is difficult. To the end, they present two simple, context-sensitive response-generation models utilizing the Recurrent Neural Network Language Model (RLM) architecture. Their models first encode past information in a hidden continuous representation, which is then decoded by the RLM to promote plausible responses that are simultaneously fluent and contextually relevant. Unlike typical complex task-oriented multi-modular dialog systems their architecture is completely data-driven and can easily be trained end-to-end using unstructured data without requiring human annotation, scripting, or automatic parsing. The proposed novel multi-reference extraction technique allows robust automated evaluation using standard SMT metrics such as BLEU and METEOR. The context-sensitive models consistently outperforms both context-independent and context-sensitive baselines by up to 11% relative improvement in BLEU in the MT setting and 24% in the IR setting, albeit using a minimal number of features. As the models are completely data-driven and self-contained, they hold the potential to improve fluency and contextual relevance in other types of dialog systems. The model lacks mechanisms both for reflecting agent intent in the response and for maintaining consistency with respect to sentiment polarity.The goal of this literature review aims at developing a context sensitive model for producing conversational responses.

6. Attention with Intention for a Neural Network Conversation Model:

Literature search was carried out using Google scholar, PubMed, Research gate, arXiv, IEEE journals, Elsevier journals and Scihub. The literature review in 2015, K. Yao, G. Zweig, and B. Peng described "Attention with Intention for a Neural Network Conversation Model"[6]. The authors developed a model that consists of three recurrent neural networks (RNNs). The source side RNN, or encoder network, encodes the source side inputs. The target side RNN, or decoder network, uses an attention mechanism to attend to particular words in the source side, when predicting a symbol in its response to the source side. Importantly, this attention in the target side is conditioned on the output from an intention RNN. Their model, which has the structural knowledge of the conversation process, is trained end-to-end without labels. They experimented with this model and observed that it generates natural responses to user inputs. This model that incorporates attention and intention processes in a neural network generates natural responses to user inputs. It also generates responses for long sentences. The attention mechanism adds more weight to the model, which increases the training time.The goal of this literature review aims at developing a attention based conversational model for producing natural conversational responses.

2.2 SUMMARY:

The summary of the literature survey are as follows:

Table 1: Summary of literature survey

AUTHOR	TITLE	PROPOSED	PUBLISHED YEAR
K. Cho, B. van Merrienboer, et.al	Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation	Phrase-based SMT system called RNN Encoder-Decoder	2014
D. Bahdanau, K. Cho, Y. Bengio	Neural Machine Translation by Jointly Learning to Align and Translate	Advantages of jointly learning to align and translate	2014
Sutskever, O. Vinyals, Q. Le	Sequence to Sequence Learning with Neural Networks	End-to-end approach to sequence learning	2014
O. Vinyals, Q. Le	A Neural Conversational Model	Advantages of Conversation Modelling on generating fluent and accurate replies	2015
A. Sordoni, M. Galley, et.al	Neural Network Approach to Context-Sensitive Generation of Conversational Responses	The challenge of context-sensitive response generation by using continuous representations	2015
K. Yao, G. Zweig, B. Peng	Attention with Intention for a Neural Network Conversation Model	Conversational Model with three Recurrent Neural Networks	2015

To summarize more further most of the researches suggest that they have used models with static and mathematical approaches and some model have bleu score lesser, so we try to improve the model performance so that it able to understand the sentiment and context and be able to chat with the human easily.

CHAPTER 3

PROPOSED WORK

3.1 INTRODUCTION:

The conversational AI chatbot will be a generative model-based approach where the model generates a response, word by word based on the query. In this model we will be using RNN Encoder Decoder/Seq2Seq LSTM model with Attention mechanism. The model will contain two LSTM-RNNs (Recurrent Neural Network), an Encoder and a Decoder. The encoder takes the sentence as input and processes one word at each time step. Its objective is to convert a sequence of words into a fixed size feature vector that encodes only the important information in the sentence while losing the unnecessary information. Each hidden state of the LSTM influences the next hidden state and the final hidden state can be seen as the summary of the sentence which is called the context. From the context, the decoder generates another sequence, one word at a time. Here, at each time step, the decoder is influenced by the context and the previously generated words. In the first step the decoder keeps the top K words with the highest probabilities. Then at each time-step this list is expanded by calculating the joint probability of the partial sequences in the list and the words in the current time-step and retaining the N most probable partial sequences until the end of the output sentence is reached. This model will be trained on Cornell Movie Dialogs Corpus. As the length of the sequence gets larger, we start losing considerable amount of information so we try to add the attention mechanism, allowing the decoder to selectively look at the input sequence while decoding. This takes the burden from the encoder to encode every useful information from the input. The project that we build will be made into a web application using flask web framework in python. So, with this idea behind we will try to build an intelligent conversational chatbot that will be useful to the society.

3.2 System Design:

The proposed architecture involves feeding the input dataset to the next stage that is preprocessing and after preprocessing is done it is proceeded to the model where it gives a generated response for the input sentence given and then it is tested using human input request and the Seq2Seq Chatbot generates the response and it is evaluated using bleu score. So this is basically the system design of our application.

3.3 System Architecture:

The following figure 3.1 shows us the architecture of our Seq2Seq Chatbot Application:

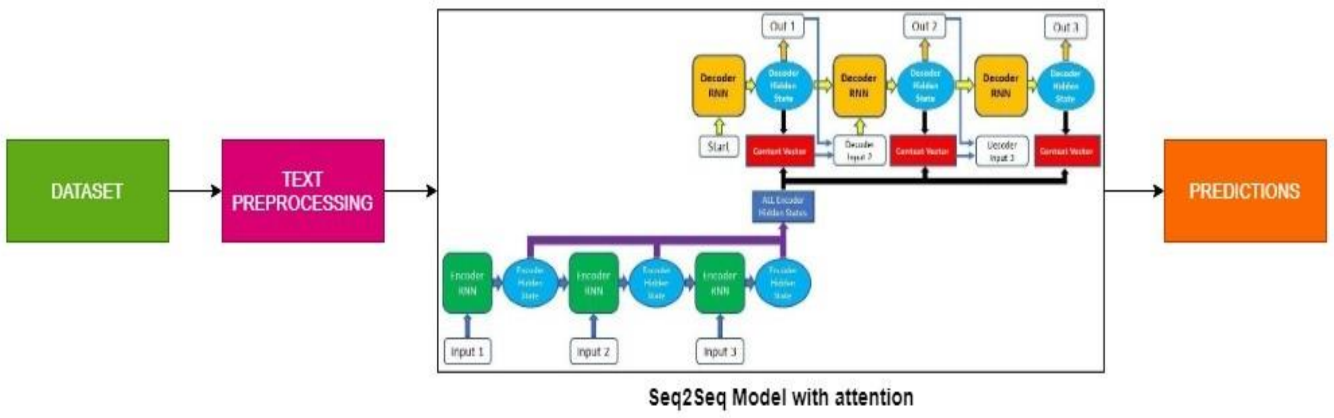


Fig 3.1 Architecture of the Seq2Seq Chatbot application

3.4 SEQUENCE-SEQUENCE:

A **sequence** is an ordered list of words. For Ex:

- A sequence of websites visited by a user, ordered by the time of access.
- A sequence of words or characters typed on a mobile by a user, or in a text such as a book.
- A sequence of products bought by a in a retail store. customer
- A sequence of symptoms observed from a patient at a hospital.

The task of **sequence prediction** consists of predicting the next word of a sentence based on the previously observed symbols. For example, if a user has visited some websites A, B, C, in that order, one may want to predict what is the next webpage that will be visited by that user to prefetch the webpage.

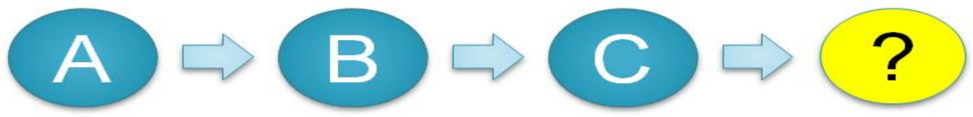


Fig 3.2 Sequence to Sequence

3.5 Encoder-Decoder Model:

The main difference between rule-based and neural network-based approaches is the presence of a learning algorithm. An encoder RNN reads the source sentence and transforms it into a fixed-length vector representation, which in turn in used as the initial hidden state of a decoder RNN that generates the target sentence. It is obvious to use a CNN as an image encoder, by first pre-training it for an image classification task and using the last hidden layer in the model as an input to the RNN decoder that generates sentences.

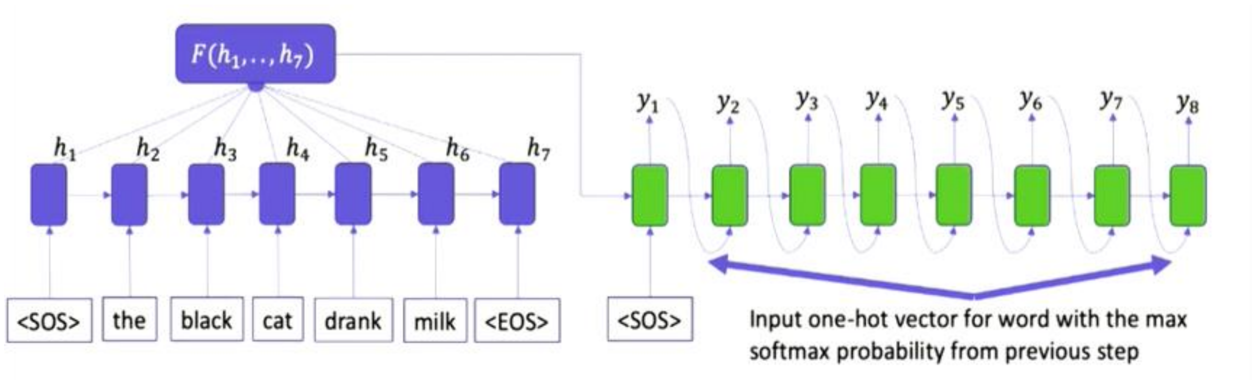

$F(h_1,..,h_7)$
h_1 h_2 h_3 h_4 h_5 h_6 h_7
y_1 y_2 y_3 y_4 y_5 y_6 y_7 y_8
<SOS> the black cat drank milk <EOS>
<SOS>
Input one-hot vector for word with the max softmax probability from previous step

Fig 3.3 Encoder Decoder Model

3.6 Recurrent Neural Network (RNN):

A recurrent neural network (RNN) is a neural network that can take as input a variable length sequence x = (x1,...,xn) and produce a sequence of hidden states h = (h1,...,hn), by using recurrence. At each step the network takes as input xi and hi−1 and generates a hidden state hi. At each step i, the hidden state hi is updated by

$$\mathbf{hi = f(Whi-1 + Uxi)} \qquad \mathbf{(3.1)}$$

where W and U are matrices containing the weights (parameters) of the network. f is a nonlinear activation function which can be the hyperbolic tangent function for example. The original implementation of an RNN is rarely used, because it suffers from the vanishing gradient problem which makes it very hard to train. Usually long short-term memory (LSTM) are used for the activation function. LSTMs were developed to combat the problem of long-term dependencies that original RNNs face. As the number of steps of the unrolling increase it becomes hard for a simple RNN to learn to remember information seen multiple steps ago.

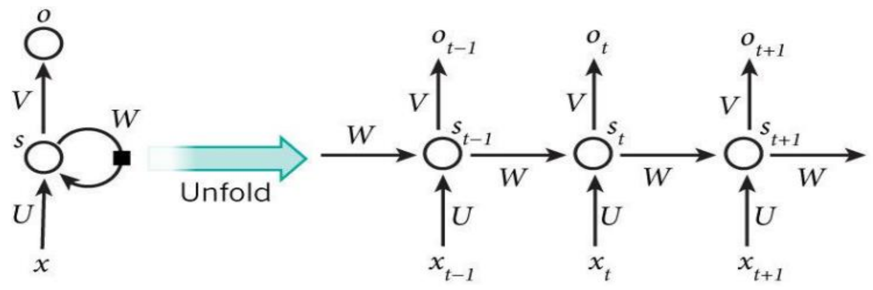


Fig 3.4 RNN

3.7 Seq2Seq Model:

Sequence To Sequence model become the Go-To model for Dialogue Systems and Machine Translation. It consists of two RNNs, an Encoder and a Decoder. The encoder takes a sentence as input and processes one word at each time step. Its objective is to convert a sequence of words into a fixed size feature vector that encodes only the important information in the sequence while losing the unnecessary information.

Each hidden state of the LSTM influences the next hidden state of the LSTM and the final hidden state can be seen as the summary of the sentence. This state is called the context or thought vector, as it represents the intention of the sequence. From the context, the decoder generates another sentence, one word at a time. Here, at each time step, the decoder is influenced by the context and the previously generated words.

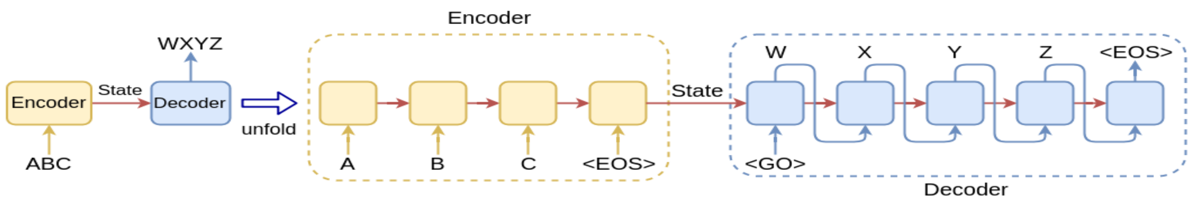


Fig 3.5 Seq2Seq model

3.8 Long Short Term Memory (LSTM):

Long Short Term Memory networks are a kind of RNN, capable of learning long-term dependencies. LSTMs are created to avoid the long-term dependency problem. Remembering information for long periods of time is their behavior, not something they struggle to learn. All RNN have the form of a chain of repeating functions of neural network. In standard RNNs, this repeating functions will have a very simple structure, such as a single tanh layer. LSTMs also have this chain like structure, but the repeating functions has a different structure. Instead of having a single neural network layer, there are four, interacting in a very special way as shown below

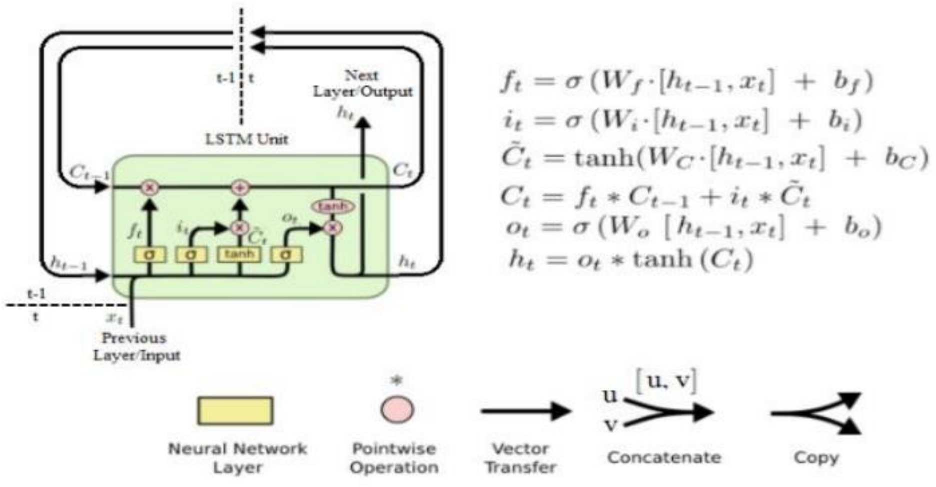


Fig 3.6 LSTM

3.9 Bidirectional Recurrent Neural Network (BRNN):

Bidirectional Recurrent Neural Networks (BRNN) connect two hidden layers of LSTM from opposite directions to the same output. With this form of generative deep learning, the output layer can get information from past (backwards) hidden states and future (forward) hidden states simultaneously.

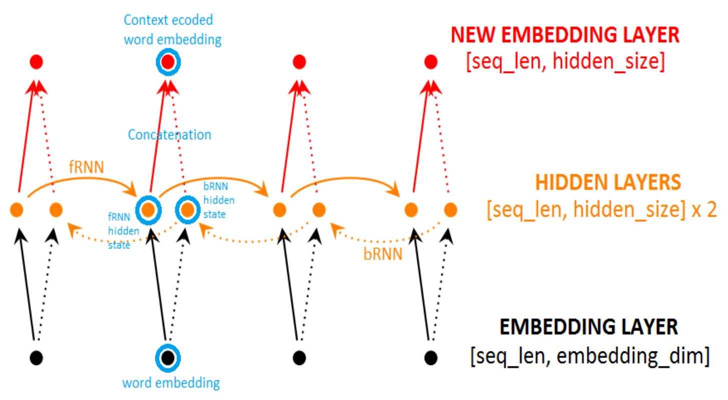


Fig 3.7 Bi-Directional RNN/LSTM

3.10 Attention Mechanism (Bahdanau Attention):

The attention mechanism, allows the decoder to selectively look at the input sentence while decoding. This takes the pressure off the encoder to encode every useful information from the input.

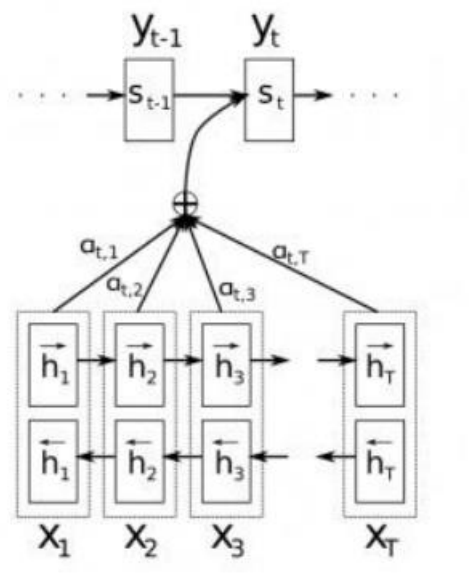

y_{t-1}
y_t
s_{t-1}
s_t
$a_{t,1}$
$a_{t,2}$
$a_{t,3}$
$a_{t,T}$
$\overrightarrow{h}_1$ $\overrightarrow{h}_2$ $\overrightarrow{h}_3$ $\overrightarrow{h}_T$
$\overleftarrow{h}_1$ $\overleftarrow{h}_2$ $\overleftarrow{h}_3$ $\overleftarrow{h}_T$
x_1 x_2 x_3 x_T

Fig 3.8 Attention model

During each time step in the decoder, instead of using a fixed context (last hidden state of encoder), a distinct context vector ci is used for generating word yi. This context vector ci is basically the weighted sum of hidden states of the encoder.

$$c_i = \sum_{j=1}^{n} \alpha_{ij} h_j$$

(3.2)

where n is the length of input sequence, hj is the hidden state at time step j.

$$\alpha_{ij} = \exp(e_{ij}) / \sum_{k=1}^{n} \exp(e_{ik})$$

(3.3)

Each hidden state in the encoder encodes information about the local context in that part of the sentence. As data flows from word 0 to word k, this local context information gets flushed. This makes it necessary for the decoder to peek through the encoder, to know the local contexts. Different parts of input sentence contain information necessary for generating different parts of the output sentence. In other words, each word in the output sentence is aligned to different parts of the input sentence. The alignment model gives us a measure of how well the output at position i match with inputs at around position j. Based on which, we take a weighted sum of the input contexts to generate each word in the output sentence.The idea of the attention mechanism is to form direct short-cut connections between the target and the source by paying “attention” to relevant source content.

3.11 SUMMARY

The summary of the proposed works are as follows:

Table 2: Summary of proposed works

MODEL/ MECHANISM	**DEFINITION**
Encoder-Decoder Model	An “encoder” RNN reads the source sentence and transforms it into a rich fixed-length vector representation, which in turn in used as the initial hidden state of a “decoder” RNN that generates the target sentence.
Recurrent Neural Network (RNN)	A **Recurrent Neural Network (RNN)** is a class of Artificial Neural Network in which the connection between different nodes forms a directed graph to give a temporal dynamic behaviour.
Seq2Seq Model	Sequence to Sequence **models** is a special class of Recurrent Neural Network architectures that we typically use (but not restricted) to solve complex Language problems like Machine Translation, Question Answering, creating Chatbots, Text Summarization, etc.
Long Short-Term Memory (LSTM)	Long Short-Term Memory (LSTM) networks are a type of recurrent neural network capable of learning order dependence (long-term dependencies) in sequence prediction problems.
Bidirectional Recurrent Neural Network (BRNN)	BRNN duplicates the RNN processing chain so that inputs are processed in both forward and reverse time order. This allows a BRNN to look at future context as well.
Attention Mechanism	The Attention mechanism in Deep Learning is based on the concept of directing your focus, and it pays greater attention to certain factors when processing the data.

CHAPTER 4
IMPLEMENTATION

4.1 Preprocessing:

4.1.1 Dataset collection:

The dataset that we used is a popular movie subtitle corpus which is “Cornell movie subtitle corpus”. This corpus contains a metadata-rich large collection of conversations extracted from raw movie scripts from popular movies. The following are found in the corpus: - 220,579 conversational exchanges between 10,292 pairs of movie characters. - Involves 9,035 characters from 617 movies - In total 304,713 utterances Other movie meta-data included genres, release year, IMDB rating, number of IMDB votes, IMDB rating.

4.1.2 Data Preprocessing:

Conversation data in the movie corpus contained Movie ID, Character ID, and Movie Line ID was separated by” +++++”. For preprocessing, conversation data were cleaned to remove this meta-data (eg. movie ID, character ID, Line ID). Also, data separators (” +++++”) were removed. Additionally, some of the characters in the data contained an unsupported encoding format by UTF-8 standard and hence was removed. Finally, data were separated into two different lists to replicate with the format of Sequence-to-Sequence model (Seq2Seq) model input format where first list is the questions and the second one was the response to answer. After separating the two lists, data in both is cleaned simultaneously. Everything except alphabetical character and some punctuation (. , ?!’) was removed in conversation. Also, all the text was converted to lowercase. Then, multiple consequent occurrences of this punctuation (. , ?!’) was reduced to one in order to reduce punctuation overhead. Next, all the punctuation except (’) was separated with a single space before and after for better performance in the Sequence-to-Sequence model (Seq2Seq) module. Finally, all the consequent multiple spaces were reduced to single space and each text string was trimmed to remove before and after space. Also, data was cleaned for removing extraneous dialogues. If multiple consequent utterances from a single person were present everything except the last utterance for the person was stored. Filter the question and answers that are too short or long (here I used 2 as my minimum length and 5 as my maximum length) i.e., the sentence with 2 to 5 words is taken into account. The most frequent words in the training data were kept as vocabulary. Additionally, the <PAD> token was used for padding input sequences to same lengths, the <EOS> token was used to signal the end of an utterance and the <UNK> token was used to replace all words not present in the vocabulary. <GO> was given to the start of the sentence. Before training, we work on the dataset to convert the variable length sequences into fixed length sequences, by padding. We use a few special tokens to fill in the sequence. Assuming that we would like our sentences (queries and responses) to be of fixed length, 10. Introduction of padding solve the problem of variable length sequences, but consider the case of large sentences. If the largest sentence in our dataset, we need to encode all our sentences to be of that length in order to not lose any words. There will be PAD symbols in the encoded version of the sentence. This will overshadow the actual information in the sentence. Bucketing kind of solves this problem, by putting sentences into buckets of different sizes. The query will be padded to a certain length and the response will be padded to a length . While running the model (training or predicting), we use a different model for each bucket, compatible with the lengths of query and response. All these models share the same parameters and hence function exactly the same way. Word Embedding is a technique for learning dense form of words in a low dimensional vector space. Each word can be seen as a point in this space, represented by a fixed length vector. Semantic relations between words are captured by this technique. The word vectors have some interesting properties. Word Embedding is typically done in the first layer of the network: Embedding layer, that maps a word (index to word in vocabulary) from vocabulary to a dense vector of given size. In the seq2seq model, the weights of the embedding layer are jointly trained with the other parameters of the model.

4.2 Seq2Seq Model with Attention mechanism:

4.2.1 Proposed Algorithm:

The algorithm to develop the model and evaluate are:

1. Start

2. Load the dataset from the official repository and importing all the necessary packages.

3. Preprocessing of dataset with removing punctuations and stop words and appending necessary tokens and filtering the dataset based on threshold length needed and frequency of words.

4. Building vocabulary dictionary, segregating dataset to question and answer and building dictionary for that and forming index to vocabulary dictionary and vocabulary to index dictionary.

5. Transforming the dataset to embedding vectors/features using word2vec and the embedding lookup is done to map the vocabulary dictionary to fed into encoder and decoder part of the model.

6. Building the encoder model with Bi-directional LSTM with their hidden states collectively fed into the attention model which is a feedforward neural network with a fixed window size with some weights.

7. Building the decoder model with Uni-LSTM with context vectors from the attention mechanism is fed into the decoder model and corresponding outputs are generated with the use of decoder initial state and context vectors.

8. Building the Bahdanau Attention model with their inputs from the encoder hidden states and generating the context vectors from a fixed window size.

9. Configuring the model with Adam optimizer and Sequence loss function and configuring the parameters such as rnn size, embedding size, batch size, epochs, learning rate, etc.

10. Splitting the dataset into train and test dataset with question dictionary fed into encoder model and answer dictionary fed into the decoder model and preparing the validation dataset based on the batch size mentioned.

11. Training the seq2seq model with feeding the data with the help of embedding vectors to predict the correct output.

12. Compute the accuracy and loss by comparing the output generated with the true words.

13. Update the parameters of the model until minimal loss is achieved.

14. Testing the model with validation data with feeding the validation data sentences and predicting the output with the true words and measuring the bleu score.

15. High bleu score measures the accuracy of the model.

16. End

4.2.2 Implementation Environment:

In this project, we used TensorFlow V1.14.0 to develop seq2seq model.Seq2Seq is industry standard choice for dialogue generation and many NLP tasks. It was coming from the TensorFlow Seq2Seq contribution API. But also reasonably it is removed and no more accessible with recent TensorFlow V2.X.But still the Seq2Seq API is accessible via TensorFlow-addon package. Also to improve the model performance using state of art technique proposed by many research paper. In Seq2Seq models we can have options of attention mechanism, Beam search, bidirectionalRNN module. We have implemented attention mechanism with the Seq2Seq model for getting more accurate results than a basic seq2seq model.

The model is trained in Google Colab with gpu runtime and for training the model to evaluate the output we have used sublime text editor with the cpu hardware configurations of 8th Gen Intel Core i5-8250U CPU@1.60GHz processor and 8GB ram. So these are the environments we used for implementing the model.

4.2.3 Proposed Model Implementation:

To implement the model, we load the Cornell movie dataset from the official repository and we import all the necessary packages. We then preprocess the dataset with splitting based on the identifiers and removing punctuations and replacing improper words and stop words using regular expression in python and appending necessary tokens and filtering the dataset based on threshold length needed and frequency of words.

We build the vocabulary dictionary and segregating dataset to question and answer and building dictionary for that and forming index to vocabulary dictionary and vocabulary to index dictionary. We then append the <EOS> tag to end of answer sentences and then we compare the sentences with the vocab dictionary if it is not present we then append <UNK> token with the sentence.

We then transform the dataset to embedding vectors/features using word2vec and the embedding lookup is done to map the vocabulary dictionary to fed into encoder and decoder part of the model. It's also important because this is the part that at every end of the sentence there will be a token to tell the network that the input is finished. For every word in the sentence, it will get the index from the appropriate word in the dictionary and add a token at the end of the sentence.

We build the encoder model with Bi-directional LSTM with their hidden states collectively fed into the attention model which is a feedforward neural network with a fixed window size with some weights. The Encoder will encode the sentence word by words into an indexed of vocabulary or known words with index, and the decoder will predict the output of the coded input by decoding the input in sequence and will try to use the last input as the next input if its possible. With this method, it is also possible to predict the next input to create a sentence. Each sentence will be assigned a token to mark the end of the sequence. At the end of prediction, there will also be a token to mark the end of the output. So, from the encoder, it will pass a state to the decoder to predict the output.

The Encoder will encode our input sentence word by word in sequence and in the end there will be a token to mark the end of a sentence. The encoder consists of an Embedding layer and a LSTM layers. The Embedding layer is a lookup table that stores the embedding of our input into a fixed sized dictionary of words. It will be passed to a LSTM layer. LSTM layer is a Long Short-Term Memory that consists of multiple layer type of RNN that will calculate the sequenced input. This layer will calculate the hidden state from the previous one and update the reset, update, and new gates.

We then build the decoder model with Uni-LSTM with context vectors from the attention mechanism is fed into the decoder model and corresponding outputs are generated with the use of decoder initial state and context vectors and greedy search is also used for improving the model by taking the argmax of the output (treated as logits) and passes the result through an embedding layer to get the next input. The Decoder will decode the input from the encoder output. It will try to predict the next output and try to use it as the next input if it's possible. The Decoder consists of an Embedding layer, LSTM layer, and a Linear layer. The embedding layer will make a lookup table for the output and passed into a GRU layer to calculate the predicted output state. After that, a Linear layer will help to calculate the activation function to determine the true value of the predicted output.

We build Bahdanau Attention model with their inputs from the encoder hidden states and generating the context vectors from a fixed window size from the encoder hidden states using a feedforward neural layer and applying a SoftMax activation function to generate the context vectors.

We then perform padding and masking the data and we do gradient clipping to prevent gradient explosion problem. We configure the model with Adam optimizer and Sequence loss function and configuring the parameters such as rnn size, embedding size, batch size, epochs, learning rate, etc.

We split the dataset into train and test dataset with question dictionary fed into encoder model and answer dictionary fed into the decoder model and preparing the validation dataset based on the batch size mentioned.

We train the seq2seq model using Interactive Session with feeding the data with the help of embedding vectors to predict the correct output. As we created the filtered question and answer list, we create training data (length of 22992) and validation data (We took the length of batch size).

As we trained the model for three different conversation as we discussed before using Adam optimizer as it can computes individual adaptive learning rates for different parameters from estimates of first and second moments of the gradient. We used Bidirectional LSTM is used in encoder side and attention mechanism is used in decoder side to improve model performance. We then compute the average accuracy and average loss by comparing the output generated with the true words.

At each step then we update the parameters of the model until minimal loss is achieved. We then tested the model with validation data with feeding the validation data sentences and predicting the output with the true words and measuring the bleu score and taking the average of bleu scores.

High bleu score measures the accuracy of the model. We then plot the graphs for comparison from different configuration of models.

4.3 Frontend Implementation:

GUI (graphical user interface) was developed using flask app. Flask app that provides a chat interface to the user, to interact with our seq2seq model. The reply method in the code is called via an AJAX request from index.js.It sends the text from user to our seq2seq model, which returns a reply. The reply is passed to index.js and rendered as text in the chat box.

4.4 Performance analysis metrics:

Bleu score:

The closer a machine translation is to a professional human translation, the better it is BLEU, a method for automatic evaluation of machine translation. We look at the adequacy, fluency, and fidelity of the translations to know it’s effectiveness. Another challenge with translations for a sentence is in the usage of different word choices and changing the word order. We will measure the closeness of translation by finding legitimate differences in word choice and word order between the reference human translation and translation generated by the machine.

A few terms in context with BLEU

(a)Reference translation is Human translation

(b)Candidate Translation is Machine translation.

To measure the machine translation effectiveness, we will evaluate the closeness of the machine translation to human reference translation using a metric known as BLEU-Bilingual Evaluation Understudy. By comparing n-gram matches between each candidate translation to the reference translations.

An n-gram is a sequence of words occurring within a given window where n represents the window size. BLEU compares the n-gram of the candidate translation with n-gram of the reference translation to count the number of matches. These matches are independent of the positions where they occur. The more the number of matches between candidate and reference translation, the better is the machine translation.In terms of Machine Translation, we define Precision as ‘the count of the number of candidate translation words which occur in any reference translation’ divided by the ‘total number of words in the candidate translation.’

$$\text{Precision} = \frac{\text{No. of candidate translation words occuring in any reference translation}}{\text{Total no. of words in the candidate translation}} \quad \textbf{(4.1)}$$

High precision are not good translations.To solve the issue, we will use modified n-gram precision. It is computed in multiple steps for each n-gram.

We first calculate Count clip for any n-gram using the following steps:

Step1: Count the maximum number of times a candidate n-gram occurs in any single reference translation; this is referred to as Count.

Step 2: For each reference sentence, count the number of times a candidate n-gram occurs. As we have three reference translations, we calculate, Ref 1 count, Ref2 count, and Ref 3 count.

Step 3: Take the maximum number of n-grams occurrences in any reference count. Also known as Max Ref Count.

Step 4: Take the minimum of the Count and Max Ref Count. Also known as Count clip as it clips the total count of each candidate word by its maximum reference count

$$\boldsymbol{Count_{Clip}} = \textbf{min(Count,Max\_Ref\_Count)} \tag{4.2}$$

Step 5: Add all these clipped counts.

Step 6: Finally, divide the clipped counts by the total (unclipped) number of candidate n-grams to get the modified precision score.

$$p_n = \frac{\sum_{C \in \{Candidates\}} \sum_{n\text{-}gram \in C} Count_{clip}(n\text{-}gram)}{\sum_{C' \in \{Candidates\}} \sum_{n\text{-}gram' \in C'} Count(n\text{-}gram')} . \tag{4.3}$$

Modified precision P_n: Sum of the clipped n-gram counts for all the candidate sentences in the corpus divide by the number of candidate n-grams. Modified n-gram precision score captures two aspects of translation: adequacy and fluency. A translation using the same words as in the references tends to satisfy adequacy. The longer n-gram matches between candidate and reference translation account for fluency We add brevity penalty to handle too short translations. Brevity Penalty(BP) will be 1.0 when the candidate translation length is the same as any reference translation length. The closest reference sentence length is the “best match length.” With the brevity penalty, we see that a high-scoring candidate translation will match the reference translations in length, in word choice, and word order. BP is an exponential decay and is calculated as shown below

$$\text{Brevity Penalty} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \leq r \end{cases} \tag{4.4}$$

r- count of words in a reference translation

c- count of words in a candidate translation

Note: Neither the brevity penalty nor the modified n-gram precision length directly considers the source length; instead, they only consider the range of reference translation lengths of the target language

Finally, we calculate BLEU

$$\text{BLEU} = \text{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right) \tag{4.5}$$

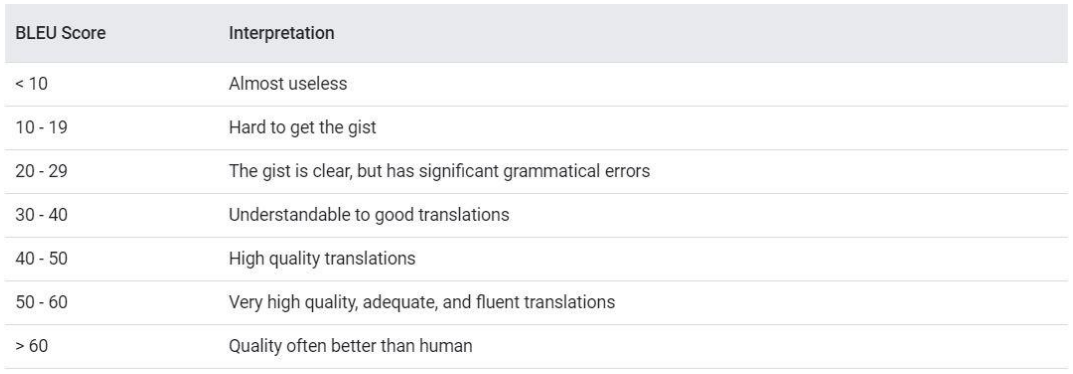
BP- brevity penalty

N: No. of n-grams, we usually use unigram, bigram, 3-gram, 4-gram

w_n: Weight for each modified precision, by default N is 4, w_n is 1/4=0.25

P_n: Modified precision

The BLEU metric ranges from 0 to 1. When the machine translation is identical to one of the reference translations, it will attain a score of 1.



BLEU Score	Interpretation
< 10	Almost useless
10 - 19	Hard to get the gist
20 - 29	The gist is clear, but has significant grammatical errors
30 - 40	Understandable to good translations
40 - 50	High quality translations
50 - 60	Very high quality, adequate, and fluent translations
> 60	Quality often better than human

Fig 4.1 Index of BLEU score

Adam optimizer:

The Adam optimization algorithm is an extension to stochastic gradient descent that has recently seen broader adoption for deep learning applications in computer vision and natural language processing. Adam is different to classical stochastic gradient descent. Stochastic gradient descent maintains a single learning rate (termed alpha) for all weight updates and the learning rate does not change during training. A learning rate is maintained for each network weight (parameter) and separately adapted as learning unfolds.

The method computes individual adaptive learning rates for different parameters from estimates of first and second moments of the gradients. Adaptive Gradient Algorithm (AdaGrad) that maintains a per-parameter learning rate that improves performance on problems with sparse gradients (e.g. natural language and computer vision problems). Root Mean Square Propagation (RMSProp) that also maintains per-parameter learning rates that are adapted based on the average of recent magnitudes of the gradients for the weight (e.g. how quickly it is changing). This means the algorithm does well on online and non-stationary problems (e.g. noisy). Adam realizes the benefits of both AdaGrad and RMSProp.

Instead of adapting the parameter learning rates based on the average first moment (the mean) as in RMSProp, Adam also makes use of the average of the second moments of the gradients (the uncentered variance). Adam is a replacement optimization algorithm for stochastic gradient descent for training deep learning models. Adam combines the best properties of the AdaGrad and RMSProp algorithms to provide an optimization algorithm that can handle sparse gradients on noisy problems. Adam is relatively easy to configure where the default configuration parameters do well on most problems. alpha (in our case lr_rate): Also referred to as the learning rate or step size. The amount that the weights are updated during training is referred to as the step size or the “learning rate.”The proportion that weights are updated (e.g. 0.001). Larger values (e.g. 0.3) results in faster initial learning before the rate is updated. Smaller values (e.g. 1.0E-5) slow learning right down during training.

Accuracy:

Accuracy is defined as the percentage of correct predictions for the test data. It can be calculated easily by dividing the number of correct predictions by the number of total prediction. In a classification problem accuracy is a measure of correct predictions made by the model in order to all the predictions. It is the proportion of correctly classified instances.

accuracy=(correct predictions)/all predictions

Based on our project we have computed accuracy based on comparing the predicted answer and real answer

np.pad will take the input array and add the padding based on the shape np.equalcompare the target and prediction elementwise

np.meanAverage of the inputs given

Eg:

target = [1, 2, 3, 4, 5] print(np.pad(target, (2, 3), 'constant', constant_values=(4, 6)))

output: [4 4 1 2 3 4 5 6 6 6]

target='Iam good'=[1 5]length=2, logits='Iam doing good'=[1 7 5]length=3

if max_seq=3 then target=[1 5 0] (After np.pad-Pad an array.) logits=[1 7 5]

take mean to get average accuracy of all batch:

compare sentence using np.equal([1, 5, 0], [1, 7 ,5]) output:[True False False]

mean of the result using np.mean(np.equal([1, 5, 0], [1, 7 ,5])))

output:0.3333333333

Loss:

Sequence loss is a loss function which is just a weighted SoftMax cross entropy loss function, but it is particularly designed to be applied in time series model (RNN). Weights should be explicitly provided as an argument, and it can be created by TF sequence mask.

In this project, tf.sequence_mask creates [batch_size, max_target_sequence_length] size of variable, then masks only the first target_sequence_length number of elements to 1. It means parts will have less weight than others. It can be implemented using tf.contrib.seq2seq.sequence_loss.It is a weighted cross-entropy loss for a sequence of logits. Logits is a tensor of shape [batch_size, sequence_length, num_decoder_symbols] and dtype float. The logits correspond to the prediction across all classes at each timestep. The targets are a tensor of shape [batch_size, sequence_length] and dtype int. The target represents the true class at each timestep. The weights are a tensor of shape [batch_size, sequence_length] and dtype float. weights constitutes the weighting of each prediction in the sequence. When using weights as masking, set all valid timesteps to 1 and all padded timesteps to 0, e.g. a mask returned by tf.sequence_mask.

4.5 Challenges:

Training is a long process which demands higher processing power and configured computing machine. Another problem is finding right hyper parameters to optimize our model. Developing a generic chatbot is really challenging. The model used in this experiment is for machine translation, the dialogue generation is treated as translation problem, where histories of earlier conversations are not taken into account. Hence, the model can be limited in performance regarding long conversation. Many of the output were repetitive and generic. Also, due to lack of real-life quality data the chatbot performed somehow below optimum for imitating human interaction. Also, many utterances were discarded due to longer length or discrepancy.

4.6 Summary:

The summary of Implementation are as follows:

Table 3: Summary of Implementation

Algorithm	Deep Neural Network (DNN), Recurrent Neural Network (RNN)
Main Technique	Sequence to Sequence (Seq2seq) modelling with encoder
Decoder Enhancement Techniques	Long Short-Term Memory (LSTM) based RNN cell, Bidirectional LSTM, Neural attention Mechanism

Web Application:

Frontend and Backend Technology used	HTML, CSS, JavaScript, Python FLASK app

CHAPTER 5

EVALUATION AND RESULT

5.1 Seq2Seq model with Attention Mechanism:

The Seq2Seq model is evaluated using metrics such as accuracy, loss and BLEU score. The main metric of comparison of models is BLEU score as it gives the correct precision of matching of generated sentences to the true sentences.

The following results show the results for each of configuration defined

5.1.1 Configurations and Results:

Table 4: Configuration table

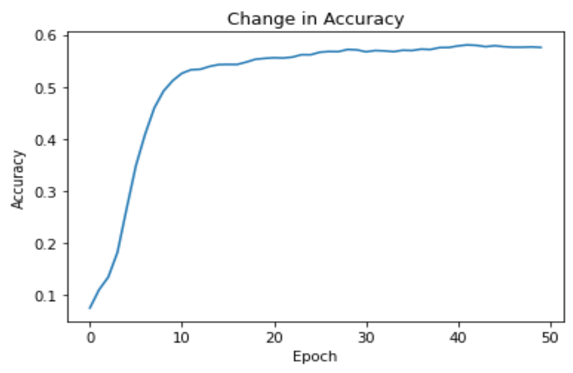
Parameters	Config1	Config2	Config3	Config4	Config5	Config6	Config7	Config8	Config9	Config10
Batch Size	128	128	512	512	128	512	512	1024	1024	1024
Embedding Size	2048	2048	1024	2048	128	512	512	512	1024	2048
RNN Size	1024	2048	1024	1024	128	512	512	512	512	1024
Learning Rate	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001
Epochs	50	50	80	80	50	80	80	50	50	50
Keep Probability	0.75	0.8	0.85	0.90	0.8	0.75	0.85	0.85	0.90	0.90
Minimum Learning Rate	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001	0.001	0.0001	0.0001
Learning Rate Decay	0.99	0.99	0.99	0.99	0.9	0.99	0.99	0.99	0.99	0.99

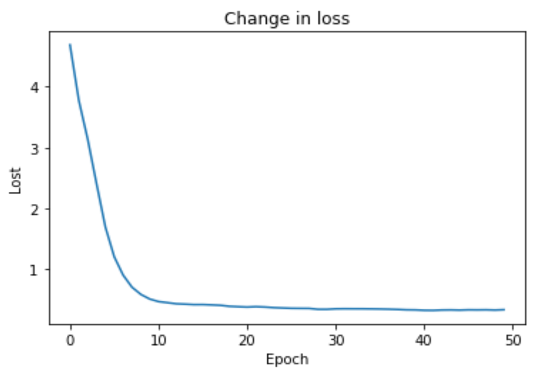
Table 5: Result table

Results	Config1	Config2	Config3	Config4	Config5	Config6	Config7	Config8	Config9	Config10
Average Accuracy	0.575	0.533	0.641	0.639	0.344	0.637	0.639	0.638	0.629	0.634
Average Loss	0.334	0.515	0.129	0.129	1.289	0.157	0.148	0.189	0.165	0.130
BLEU Score	30.99	26.39	36.13	36.52	11.98	34.77	35.74	38.67	38.77	39.06

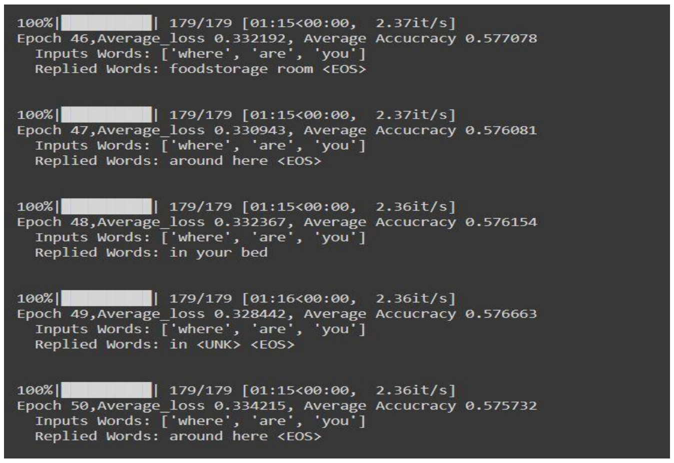
Configuration 1:

Accuracy and Loss:

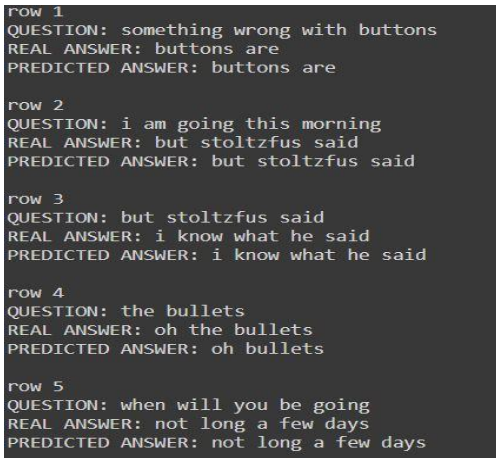




Epochs and Predictions:



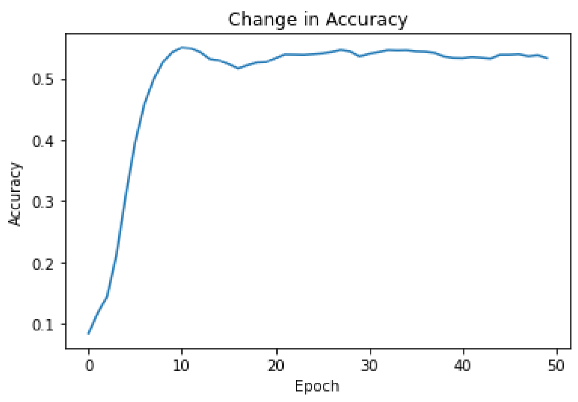
```
100%|██████████| 179/179 [01:15<00:00,  2.37it/s]
Epoch 46,Average_loss 0.332192, Average Accucracy 0.577078
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>

100%|██████████| 179/179 [01:15<00:00,  2.37it/s]
Epoch 47,Average_loss 0.330943, Average Accucracy 0.576081
  Inputs Words: ['where', 'are', 'you']
  Replied Words: around here <EOS>

100%|██████████| 179/179 [01:15<00:00,  2.36it/s]
Epoch 48,Average_loss 0.332367, Average Accucracy 0.576154
  Inputs Words: ['where', 'are', 'you']
  Replied Words: in your bed

100%|██████████| 179/179 [01:16<00:00,  2.36it/s]
Epoch 49,Average_loss 0.328442, Average Accucracy 0.576663
  Inputs Words: ['where', 'are', 'you']
  Replied Words: in <UNK> <EOS>

100%|██████████| 179/179 [01:15<00:00,  2.36it/s]
Epoch 50,Average_loss 0.334215, Average Accucracy 0.575732
  Inputs Words: ['where', 'are', 'you']
  Replied Words: around here <EOS>
```

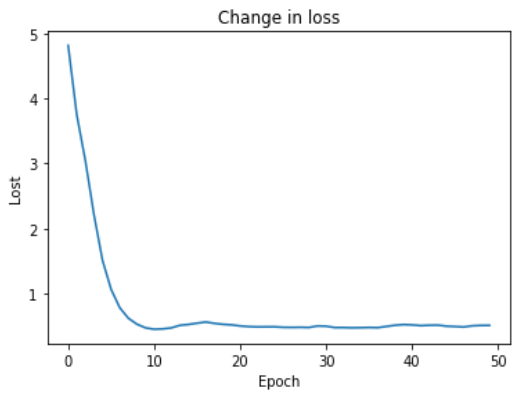


```
row 1
QUESTION: something wrong with buttons
REAL ANSWER: buttons are
PREDICTED ANSWER: buttons are

row 2
QUESTION: i am going this morning
REAL ANSWER: but stoltzfus said
PREDICTED ANSWER: but stoltzfus said

row 3
QUESTION: but stoltzfus said
REAL ANSWER: i know what he said
PREDICTED ANSWER: i know what he said

row 4
QUESTION: the bullets
REAL ANSWER: oh the bullets
PREDICTED ANSWER: oh bullets

row 5
QUESTION: when will you be going
REAL ANSWER: not long a few days
PREDICTED ANSWER: not long a few days
```

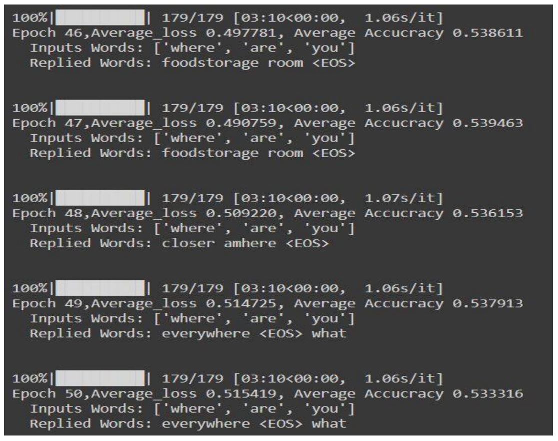
Configuration 2:

Accuracy and Loss:

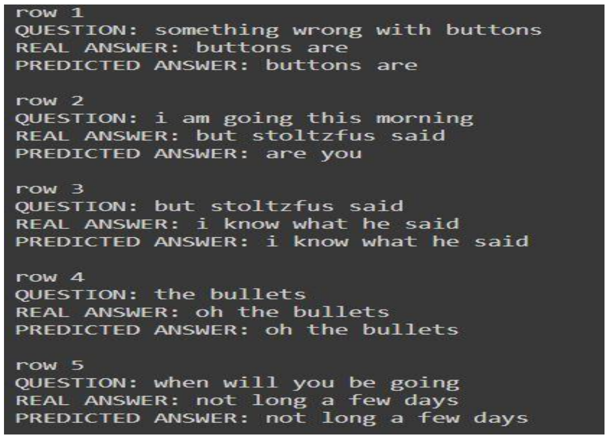




Epochs and Predictions:



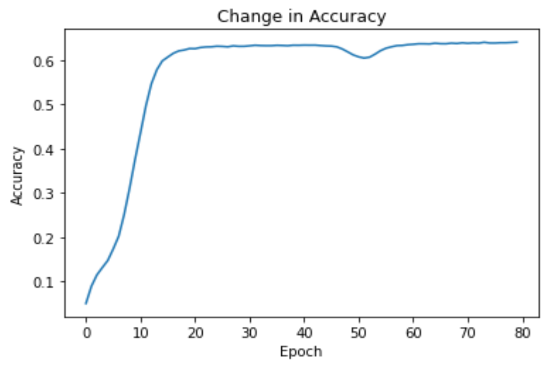
```
100%|██████████| 179/179 [03:10<00:00,  1.06s/it]
Epoch 46,Average_loss 0.497781, Average Accucracy 0.538611
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>


100%|██████████| 179/179 [03:10<00:00,  1.06s/it]
Epoch 47,Average_loss 0.490759, Average Accucracy 0.539463
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>


100%|██████████| 179/179 [03:10<00:00,  1.07s/it]
Epoch 48,Average_loss 0.509220, Average Accucracy 0.536153
  Inputs Words: ['where', 'are', 'you']
  Replied Words: closer amhere <EOS>


100%|██████████| 179/179 [03:10<00:00,  1.06s/it]
Epoch 49,Average_loss 0.514725, Average Accucracy 0.537913
  Inputs Words: ['where', 'are', 'you']
  Replied Words: everywhere <EOS> what


100%|██████████| 179/179 [03:10<00:00,  1.06s/it]
Epoch 50,Average_loss 0.515419, Average Accucracy 0.533316
  Inputs Words: ['where', 'are', 'you']
  Replied Words: everywhere <EOS> what
```

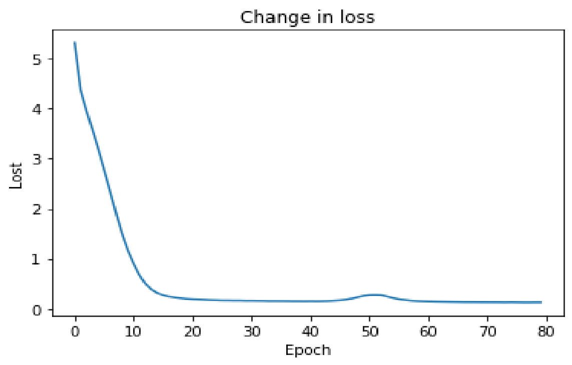


```
row 1
QUESTION: something wrong with buttons
REAL ANSWER: buttons are
PREDICTED ANSWER: buttons are

row 2
QUESTION: i am going this morning
REAL ANSWER: but stoltzfus said
PREDICTED ANSWER: are you

row 3
QUESTION: but stoltzfus said
REAL ANSWER: i know what he said
PREDICTED ANSWER: i know what he said

row 4
QUESTION: the bullets
REAL ANSWER: oh the bullets
PREDICTED ANSWER: oh the bullets

row 5
QUESTION: when will you be going
REAL ANSWER: not long a few days
PREDICTED ANSWER: not long a few days
```

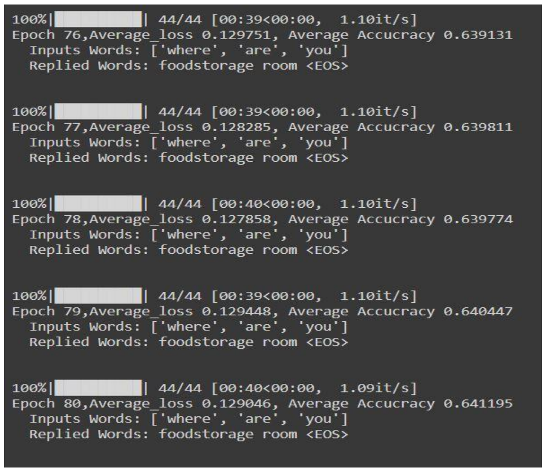
Configuration 3:

Accuracy and Loss:

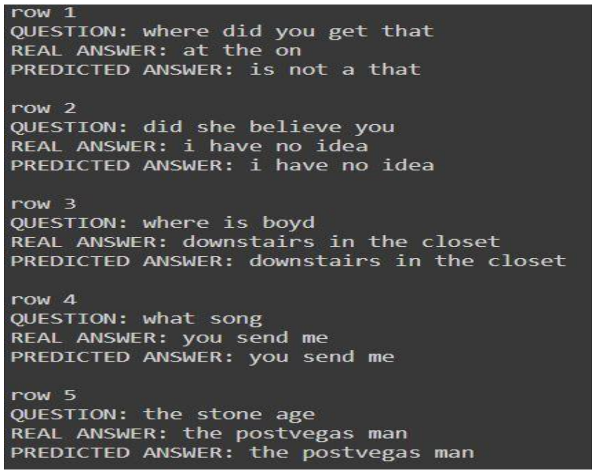




Epochs and Predictions:



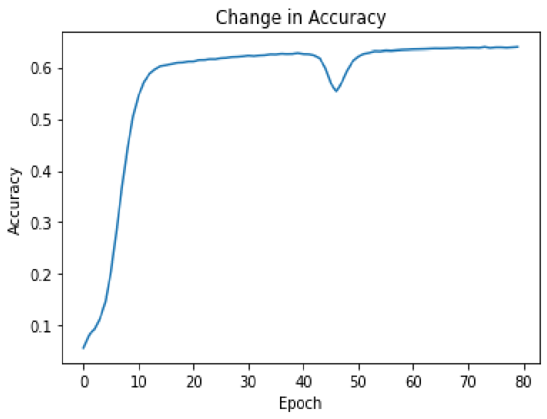
```
100%|██████████| 44/44 [00:39<00:00,  1.10it/s]
Epoch 76,Average_loss 0.129751, Average Accucracy 0.639131
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>


100%|██████████| 44/44 [00:39<00:00,  1.10it/s]
Epoch 77,Average_loss 0.128285, Average Accucracy 0.639811
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>


100%|██████████| 44/44 [00:40<00:00,  1.10it/s]
Epoch 78,Average_loss 0.127858, Average Accucracy 0.639774
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>


100%|██████████| 44/44 [00:39<00:00,  1.10it/s]
Epoch 79,Average_loss 0.129448, Average Accucracy 0.640447
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>


100%|██████████| 44/44 [00:40<00:00,  1.09it/s]
Epoch 80,Average_loss 0.129046, Average Accucracy 0.641195
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>
```

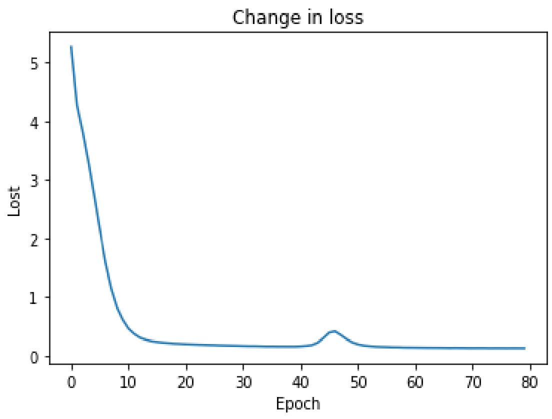


```
row 1
QUESTION: where did you get that
REAL ANSWER: at the on
PREDICTED ANSWER: is not a that

row 2
QUESTION: did she believe you
REAL ANSWER: i have no idea
PREDICTED ANSWER: i have no idea

row 3
QUESTION: where is boyd
REAL ANSWER: downstairs in the closet
PREDICTED ANSWER: downstairs in the closet

row 4
QUESTION: what song
REAL ANSWER: you send me
PREDICTED ANSWER: you send me

row 5
QUESTION: the stone age
REAL ANSWER: the postvegas man
PREDICTED ANSWER: the postvegas man
```

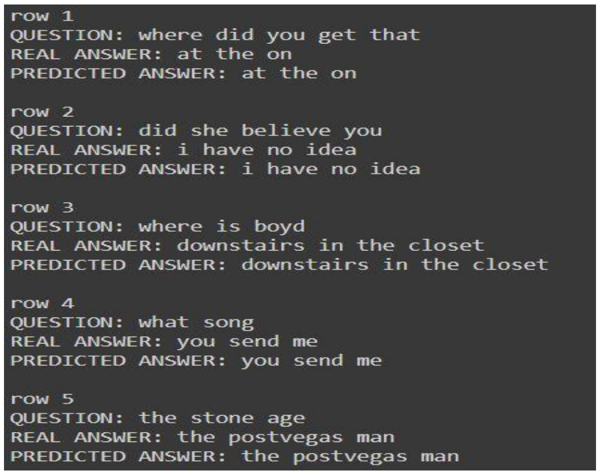
Configuration 4:

Accuracy and Loss:

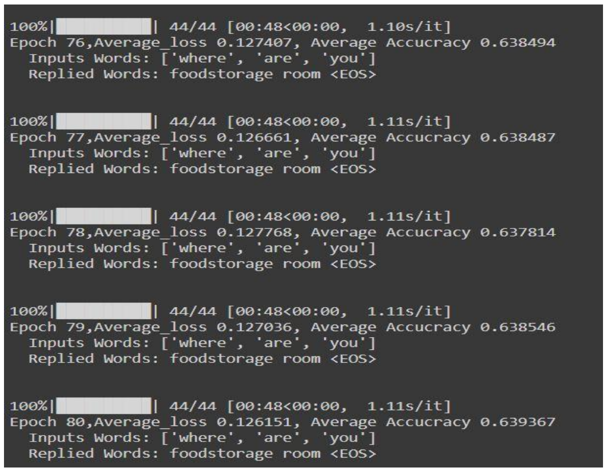




Epochs and Predictions:



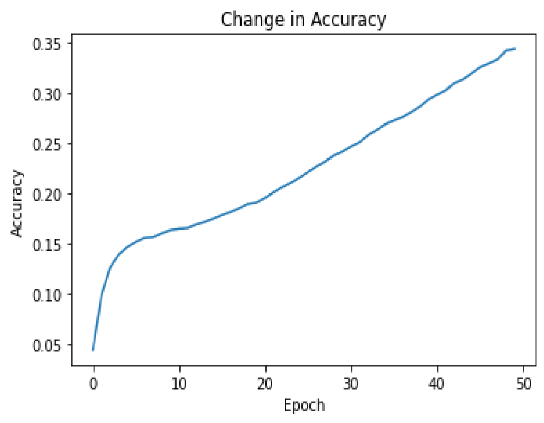
```
row 1
QUESTION: where did you get that
REAL ANSWER: at the on
PREDICTED ANSWER: at the on

row 2
QUESTION: did she believe you
REAL ANSWER: i have no idea
PREDICTED ANSWER: i have no idea

row 3
QUESTION: where is boyd
REAL ANSWER: downstairs in the closet
PREDICTED ANSWER: downstairs in the closet

row 4
QUESTION: what song
REAL ANSWER: you send me
PREDICTED ANSWER: you send me

row 5
QUESTION: the stone age
REAL ANSWER: the postvegas man
PREDICTED ANSWER: the postvegas man
```

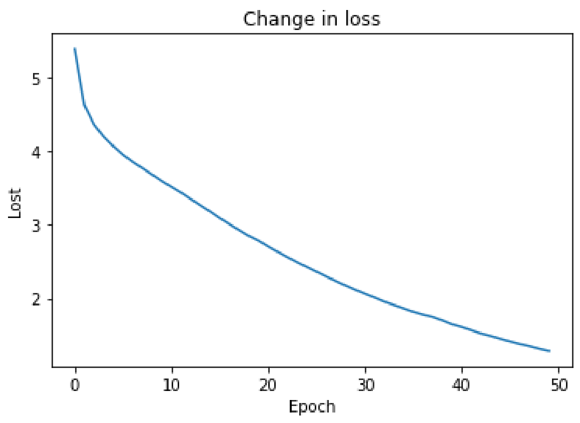


```
100%|██████████| 44/44 [00:48<00:00,  1.10s/it]
Epoch 76,Average_loss 0.127407, Average Accucracy 0.638494
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>


100%|██████████| 44/44 [00:48<00:00,  1.11s/it]
Epoch 77,Average_loss 0.126661, Average Accucracy 0.638487
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>


100%|██████████| 44/44 [00:48<00:00,  1.11s/it]
Epoch 78,Average_loss 0.127768, Average Accucracy 0.637814
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>


100%|██████████| 44/44 [00:48<00:00,  1.11s/it]
Epoch 79,Average_loss 0.127036, Average Accucracy 0.638546
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>


100%|██████████| 44/44 [00:48<00:00,  1.11s/it]
Epoch 80,Average_loss 0.126151, Average Accucracy 0.639367
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>
```

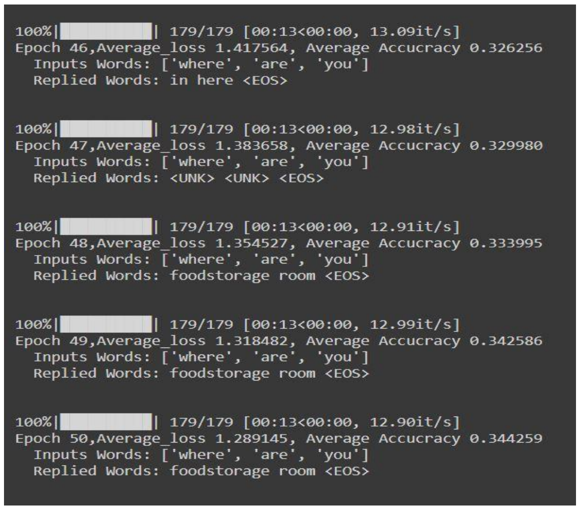
Configuration 5:

Accuracy and Loss:

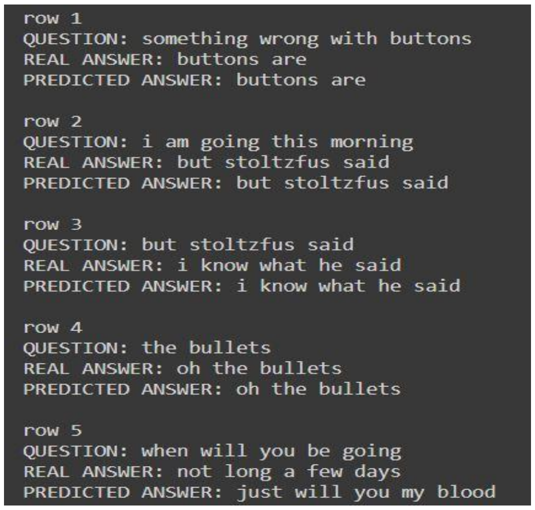




Epochs and Predictions:



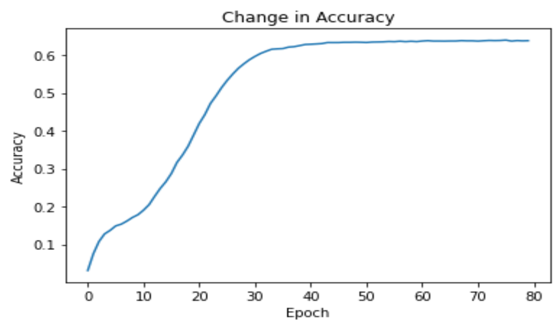
```
100%|██████████| 179/179 [00:13<00:00, 13.09it/s]
Epoch 46,Average_loss 1.417564, Average Accucracy 0.326256
  Inputs Words: ['where', 'are', 'you']
  Replied Words: in here <EOS>


100%|██████████| 179/179 [00:13<00:00, 12.98it/s]
Epoch 47,Average_loss 1.383658, Average Accucracy 0.329980
  Inputs Words: ['where', 'are', 'you']
  Replied Words: <UNK> <UNK> <EOS>


100%|██████████| 179/179 [00:13<00:00, 12.91it/s]
Epoch 48,Average_loss 1.354527, Average Accucracy 0.333995
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>


100%|██████████| 179/179 [00:13<00:00, 12.99it/s]
Epoch 49,Average_loss 1.318482, Average Accucracy 0.342586
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>


100%|██████████| 179/179 [00:13<00:00, 12.90it/s]
Epoch 50,Average_loss 1.289145, Average Accucracy 0.344259
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>
```

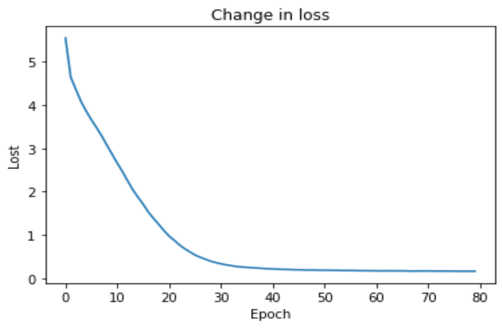


```
row 1
QUESTION: something wrong with buttons
REAL ANSWER: buttons are
PREDICTED ANSWER: buttons are

row 2
QUESTION: i am going this morning
REAL ANSWER: but stoltzfus said
PREDICTED ANSWER: but stoltzfus said

row 3
QUESTION: but stoltzfus said
REAL ANSWER: i know what he said
PREDICTED ANSWER: i know what he said

row 4
QUESTION: the bullets
REAL ANSWER: oh the bullets
PREDICTED ANSWER: oh the bullets

row 5
QUESTION: when will you be going
REAL ANSWER: not long a few days
PREDICTED ANSWER: just will you my blood
```

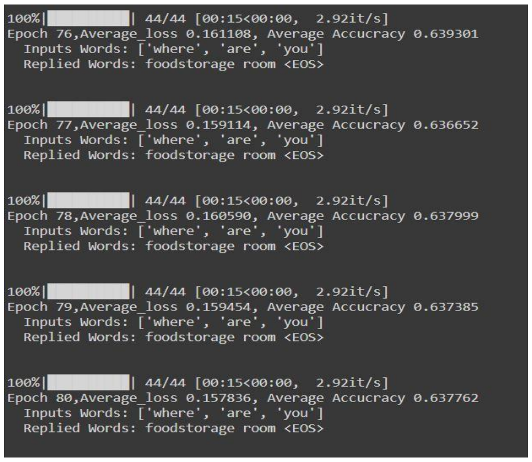
Configuration 6:

Accuracy and Loss:

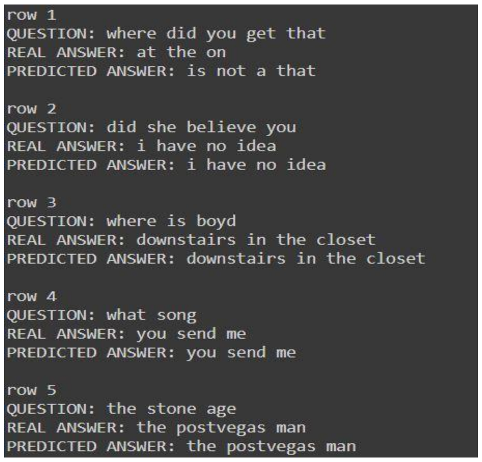




Epochs and Predictions:



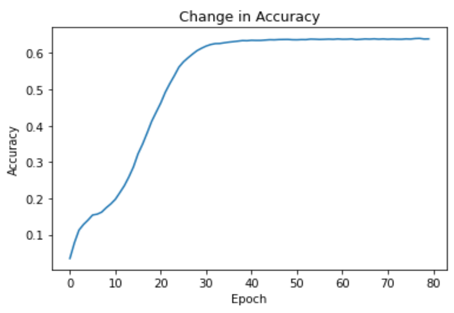
```
100%|██████████| 44/44 [00:15<00:00,  2.92it/s]
Epoch 76,Average_loss 0.161108, Average Accucracy 0.639301
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>


100%|██████████| 44/44 [00:15<00:00,  2.92it/s]
Epoch 77,Average_loss 0.159114, Average Accucracy 0.636652
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>


100%|██████████| 44/44 [00:15<00:00,  2.92it/s]
Epoch 78,Average_loss 0.160590, Average Accucracy 0.637999
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>


100%|██████████| 44/44 [00:15<00:00,  2.92it/s]
Epoch 79,Average_loss 0.159454, Average Accucracy 0.637385
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>


100%|██████████| 44/44 [00:15<00:00,  2.92it/s]
Epoch 80,Average_loss 0.157836, Average Accucracy 0.637762
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>
```

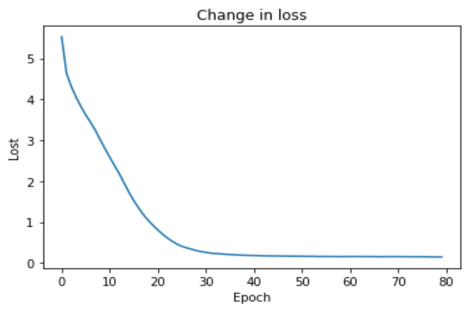


```
row 1
QUESTION: where did you get that
REAL ANSWER: at the on
PREDICTED ANSWER: is not a that

row 2
QUESTION: did she believe you
REAL ANSWER: i have no idea
PREDICTED ANSWER: i have no idea

row 3
QUESTION: where is boyd
REAL ANSWER: downstairs in the closet
PREDICTED ANSWER: downstairs in the closet

row 4
QUESTION: what song
REAL ANSWER: you send me
PREDICTED ANSWER: you send me

row 5
QUESTION: the stone age
REAL ANSWER: the postvegas man
PREDICTED ANSWER: the postvegas man
```

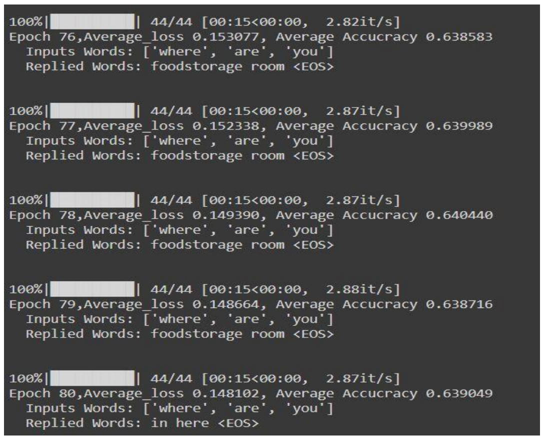
Configuration 7:

Accuracy and Loss:

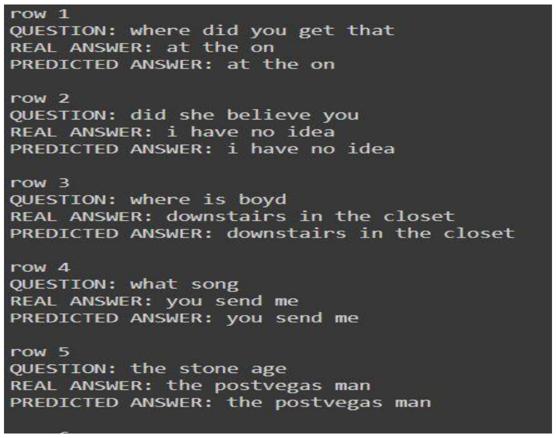




Epochs and Predictions:



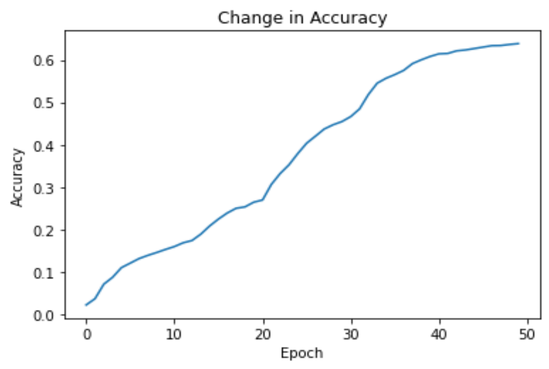
```
100%|██████████| 44/44 [00:15<00:00,  2.82it/s]
Epoch 76,Average_loss 0.153077, Average Accucracy 0.638583
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>


100%|██████████| 44/44 [00:15<00:00,  2.87it/s]
Epoch 77,Average_loss 0.152338, Average Accucracy 0.639989
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>


100%|██████████| 44/44 [00:15<00:00,  2.87it/s]
Epoch 78,Average_loss 0.149390, Average Accucracy 0.640440
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>


100%|██████████| 44/44 [00:15<00:00,  2.88it/s]
Epoch 79,Average_loss 0.148664, Average Accucracy 0.638716
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>


100%|██████████| 44/44 [00:15<00:00,  2.87it/s]
Epoch 80,Average_loss 0.148102, Average Accucracy 0.639049
  Inputs Words: ['where', 'are', 'you']
  Replied Words: in here <EOS>
```

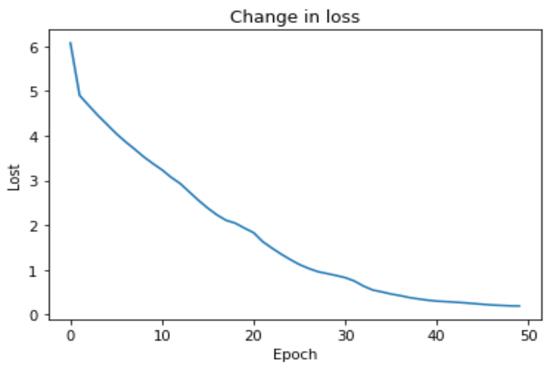


```
row 1
QUESTION: where did you get that
REAL ANSWER: at the on
PREDICTED ANSWER: at the on

row 2
QUESTION: did she believe you
REAL ANSWER: i have no idea
PREDICTED ANSWER: i have no idea

row 3
QUESTION: where is boyd
REAL ANSWER: downstairs in the closet
PREDICTED ANSWER: downstairs in the closet

row 4
QUESTION: what song
REAL ANSWER: you send me
PREDICTED ANSWER: you send me

row 5
QUESTION: the stone age
REAL ANSWER: the postvegas man
PREDICTED ANSWER: the postvegas man
```

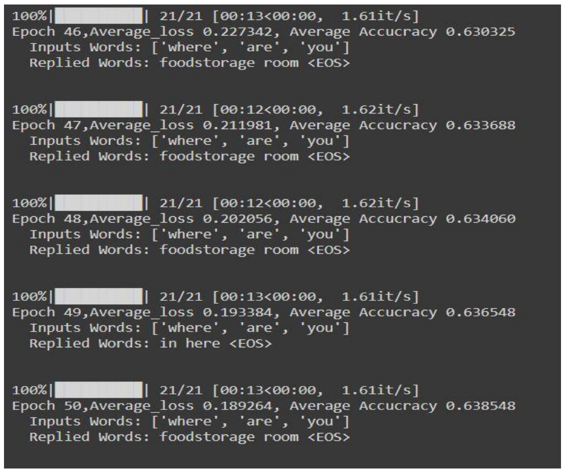
Configuration 8:

Accuracy and Loss:

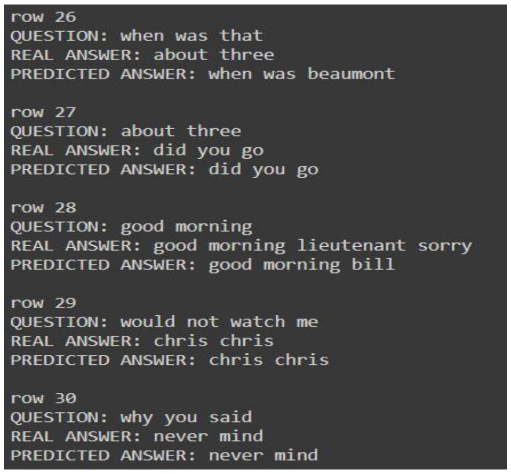




Epochs and Predictions:



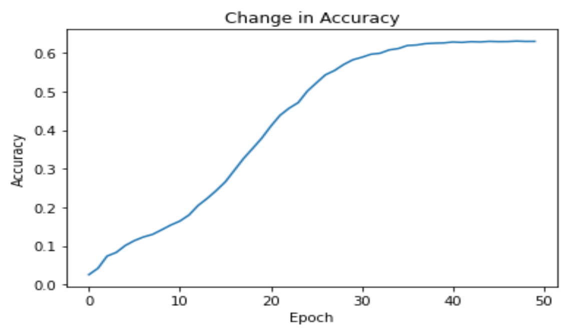
```
100%|██████████| 21/21 [00:13<00:00,  1.61it/s]
Epoch 46,Average_loss 0.227342, Average Accucracy 0.630325
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>


100%|██████████| 21/21 [00:12<00:00,  1.62it/s]
Epoch 47,Average_loss 0.211981, Average Accucracy 0.633688
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>


100%|██████████| 21/21 [00:12<00:00,  1.62it/s]
Epoch 48,Average_loss 0.202056, Average Accucracy 0.634060
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>


100%|██████████| 21/21 [00:13<00:00,  1.61it/s]
Epoch 49,Average_loss 0.193384, Average Accucracy 0.636548
  Inputs Words: ['where', 'are', 'you']
  Replied Words: in here <EOS>


100%|██████████| 21/21 [00:13<00:00,  1.61it/s]
Epoch 50,Average_loss 0.189264, Average Accucracy 0.638548
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>
```

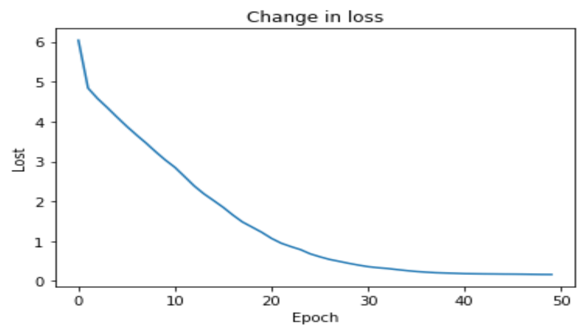


```
row 26
QUESTION: when was that
REAL ANSWER: about three
PREDICTED ANSWER: when was beaumont

row 27
QUESTION: about three
REAL ANSWER: did you go
PREDICTED ANSWER: did you go

row 28
QUESTION: good morning
REAL ANSWER: good morning lieutenant sorry
PREDICTED ANSWER: good morning bill

row 29
QUESTION: would not watch me
REAL ANSWER: chris chris
PREDICTED ANSWER: chris chris

row 30
QUESTION: why you said
REAL ANSWER: never mind
PREDICTED ANSWER: never mind
```

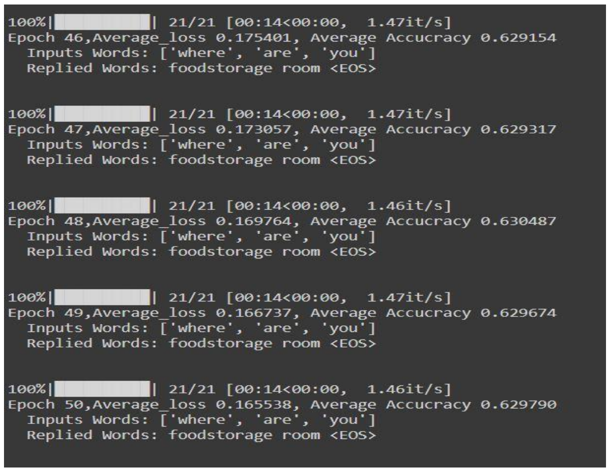
Configuration 9:

Accuracy and Loss:

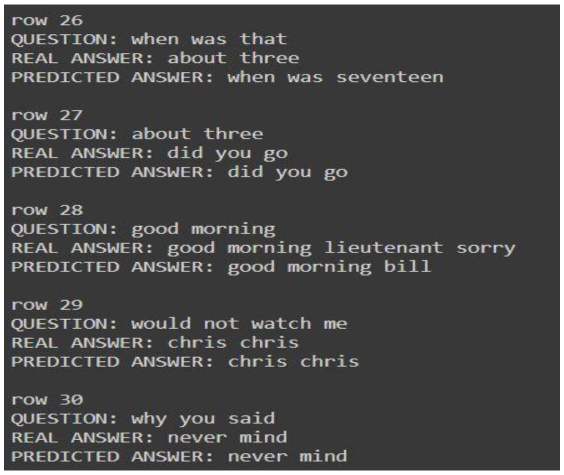




Epochs and Predictions:



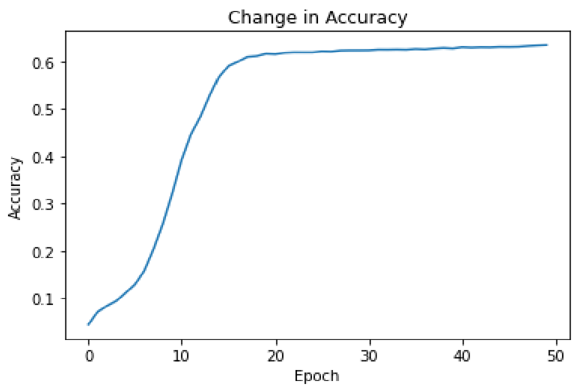
```
100%|██████████| 21/21 [00:14<00:00,  1.47it/s]
Epoch 46,Average_loss 0.175401, Average Accucracy 0.629154
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>

100%|██████████| 21/21 [00:14<00:00,  1.47it/s]
Epoch 47,Average_loss 0.173057, Average Accucracy 0.629317
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>

100%|██████████| 21/21 [00:14<00:00,  1.46it/s]
Epoch 48,Average_loss 0.169764, Average Accucracy 0.630487
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>

100%|██████████| 21/21 [00:14<00:00,  1.47it/s]
Epoch 49,Average_loss 0.166737, Average Accucracy 0.629674
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>

100%|██████████| 21/21 [00:14<00:00,  1.46it/s]
Epoch 50,Average_loss 0.165538, Average Accucracy 0.629790
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>
```

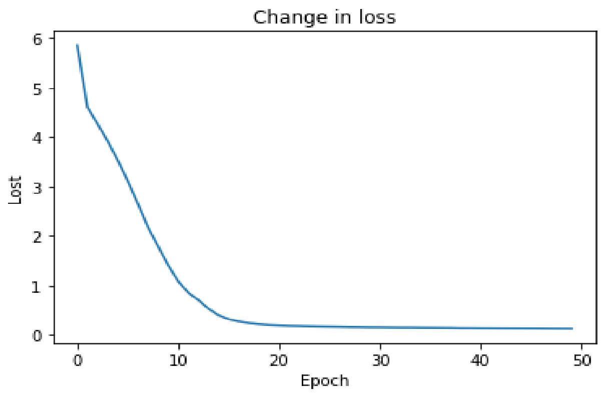


```
row 26
QUESTION: when was that
REAL ANSWER: about three
PREDICTED ANSWER: when was seventeen

row 27
QUESTION: about three
REAL ANSWER: did you go
PREDICTED ANSWER: did you go

row 28
QUESTION: good morning
REAL ANSWER: good morning lieutenant sorry
PREDICTED ANSWER: good morning bill

row 29
QUESTION: would not watch me
REAL ANSWER: chris chris
PREDICTED ANSWER: chris chris

row 30
QUESTION: why you said
REAL ANSWER: never mind
PREDICTED ANSWER: never mind
```

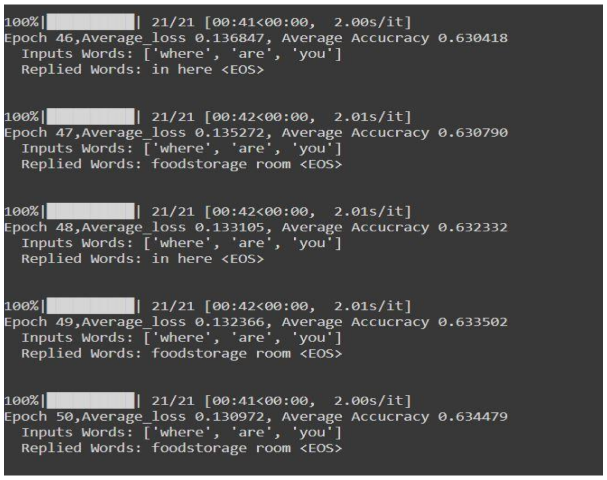
Configuration 10:

Accuracy and Loss:

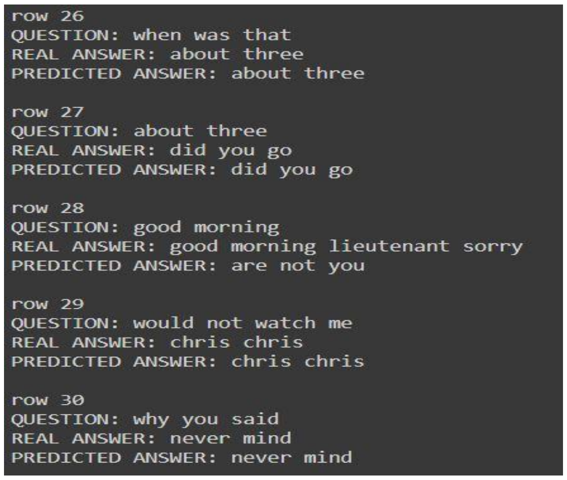




Epochs and Predictions:

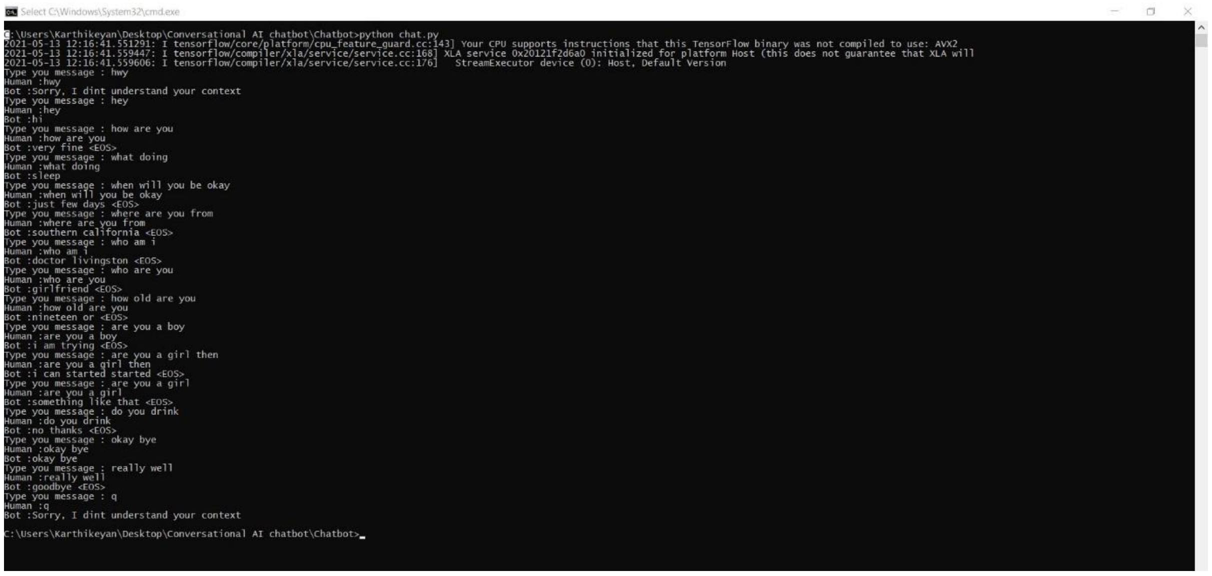


```
100%|██████████| 21/21 [00:41<00:00,  2.00s/it]
Epoch 46,Average_loss 0.136847, Average Accucracy 0.630418
  Inputs Words: ['where', 'are', 'you']
  Replied Words: in here <EOS>

100%|██████████| 21/21 [00:42<00:00,  2.01s/it]
Epoch 47,Average_loss 0.135272, Average Accucracy 0.630790
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>

100%|██████████| 21/21 [00:42<00:00,  2.01s/it]
Epoch 48,Average_loss 0.133105, Average Accucracy 0.632332
  Inputs Words: ['where', 'are', 'you']
  Replied Words: in here <EOS>

100%|██████████| 21/21 [00:42<00:00,  2.01s/it]
Epoch 49,Average_loss 0.132366, Average Accucracy 0.633502
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>

100%|██████████| 21/21 [00:41<00:00,  2.00s/it]
Epoch 50,Average_loss 0.130972, Average Accucracy 0.634479
  Inputs Words: ['where', 'are', 'you']
  Replied Words: foodstorage room <EOS>
```



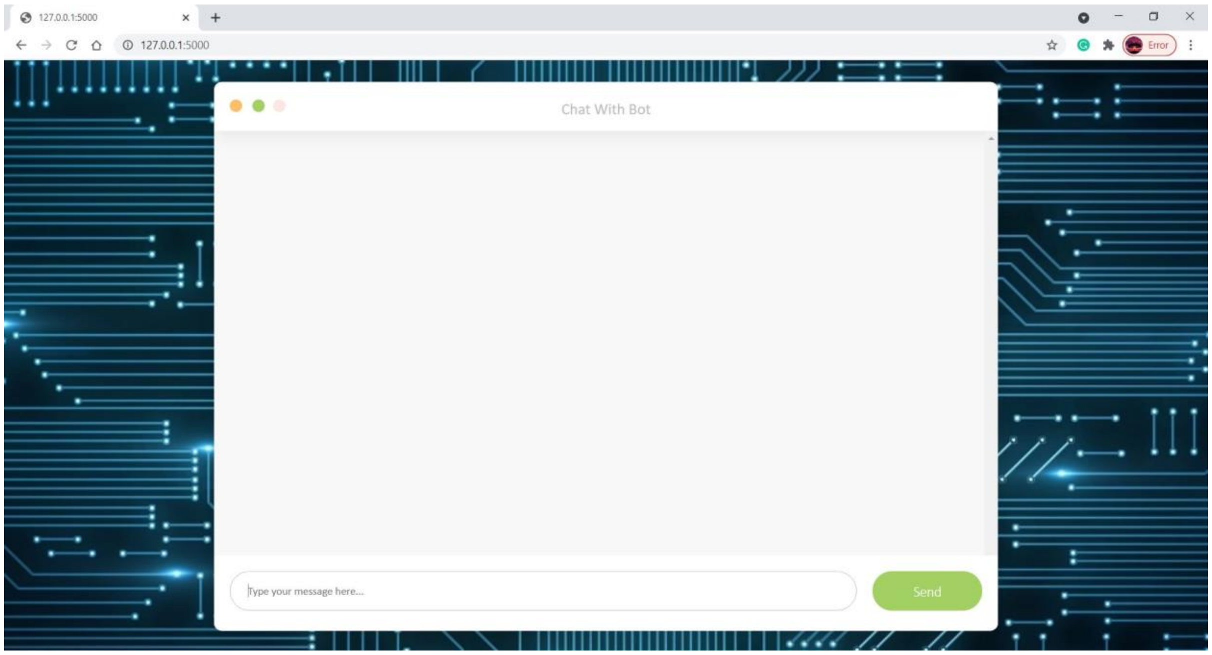
```
row 26
QUESTION: when was that
REAL ANSWER: about three
PREDICTED ANSWER: about three

row 27
QUESTION: about three
REAL ANSWER: did you go
PREDICTED ANSWER: did you go

row 28
QUESTION: good morning
REAL ANSWER: good morning lieutenant sorry
PREDICTED ANSWER: are not you

row 29
QUESTION: would not watch me
REAL ANSWER: chris chris
PREDICTED ANSWER: chris chris

row 30
QUESTION: why you said
REAL ANSWER: never mind
PREDICTED ANSWER: never mind
```

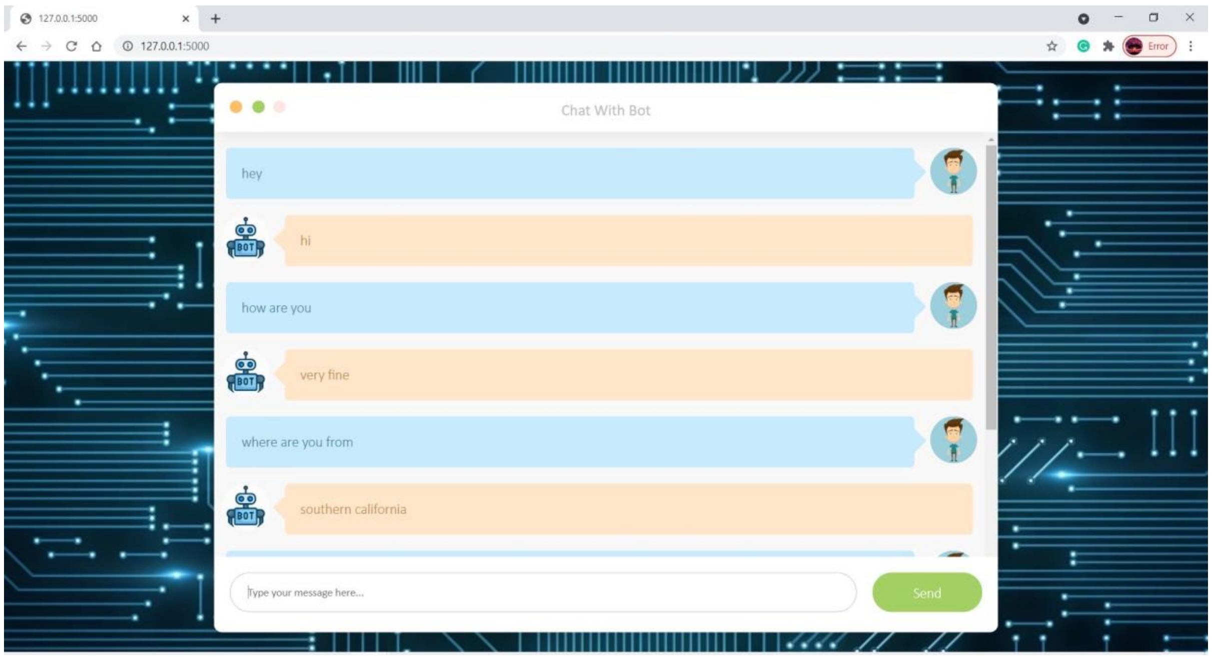
Command Prompt Output:



```
C:\Users\Karthikeyan\Desktop\Conversational AI chatbot\Chatbot>python chat.py
2021-05-13 12:16:41.551291: I tensorflow/core/platform/cpu_feature_guard.cc:143] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX2
2021-05-13 12:16:41.559447: I tensorflow/compiler/xla/service/service.cc:168] XLA service 0x20121f2d6a0 initialized for platform Host (this does not guarantee that XLA will
2021-05-13 12:16:41.559606: I tensorflow/compiler/xla/service/service.cc:176]   StreamExecutor device (0): Host, Default Version
Type you message : hwy
Human :hwy
Bot :Sorry, I dint understand your context
Type you message : hey
Human :hey
Bot :hi
Type you message : how are you
Human :how are you
Bot :very fine <EOS>
Type you message : what doing
Human :what doing
Bot :sleep
Type you message : when will you be okay
Human :when will you be okay
Bot :just few days <EOS>
Type you message : where are you from
Human :where are you from
Bot :southern california <EOS>
Type you message : who am i
Human :who am i
Bot :doctor livingston <EOS>
Type you message : who are you
Human :who are you
Bot :girlfriend <EOS>
Type you message : how old are you
Human :how old are you
Bot :nineteen or <EOS>
Type you message : are you a boy
Human :are you a boy
Bot :i am trying <EOS>
Type you message : are you a girl then
Human :are you a girl then
Bot :i can started started <EOS>
Type you message : are you a girl
Human :are you a girl
Bot :something like that <EOS>
Type you message : do you drink
Human :do you drink
Bot :no thanks <EOS>
Type you message : okay bye
Human :okay bye
Bot :okay bye
Type you message : really well
Human :really well
Bot :goodbye <EOS>
Type you message : q
Human :q
Bot :Sorry, I dint understand your context

C:\Users\Karthikeyan\Desktop\Conversational AI chatbot\Chatbot>
```

Front end UI based Output:





hey

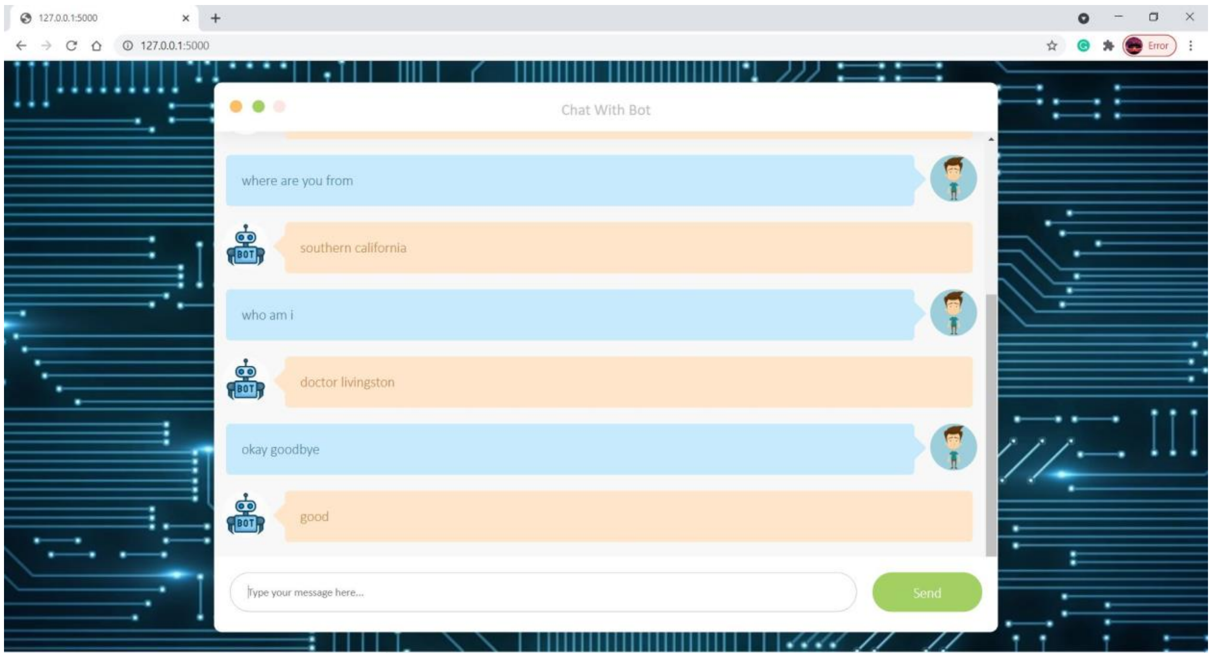
hi

how are you

very fine

where are you from

southern california



where are you from

southern california

who am i

doctor livingston

okay goodbye

good

Fig 5. Screenshots of epochs, configurations, predictions, plots of accuracy and loss, command prompt output and web application outputs

5.2 Performance Analysis:

While performing analysis of the various models we tend to find the most precise model with better results and accuracy. We see that model 10 has a bleu score of 39.06 which is high when compared to other models and accuracy is high and loss is also low when compared to other models, so it has high degree of correctness of predictions that match with the true sentences.

The other models that we have used are having bleu scores in range of 26-40 which is of median level that can still be used. The lowest accuracy model that we have resulted is that model 5 with a BLEU score of 11.98 which is not of high use since it is predicting incorrectly for many sentences. The model 10 is of high BLEU score and according to the BLEU score indicator it is in the range of 30-40 mentioning understandable to good translations. Yet we can still achieve high BLEU score but since computation power is essential for achieving that. Compared to other standard models by referring the research papers most of the models are having bleu score within the range of 30-35,but model 10 is more than that and hence can predict more accurately.

So model 10 is definitely more better than other models and can predict the sentence better than other models.

Table 6: Comparison of BLEU scores with existent model approaches

Research Paper	BLEU score
K. Cho, B. van Merrienboer, et.al - “Learning Phrase Representation RNN Encoder-Decoder for Statistical Machine Translation”	33.3
D. Bahdanau, K. Cho, Y. Bengio - “Neural Machine Translation By Jointly Learning To Align And Translate ”	35.63
Sutskever, O. Vinyals, Q. Le – “Sequence to Sequence Learning with Neural Networks”	36.5
A. Sordoni, M. Galley, et.al – “A Neural Network Approach to Context-Sensitive Generation of Conversational Responses”	33.0
Proposed Approach – “Conversational AI Chatbot”	39.06

5.3 Summary:

The summary of the results for the different configurated models are as follows

Table 7 Summary of the Results

Results	Config1	Config2	Config3	Config4	Config5	Config6	Config7	Config8	Config9	Config10
Average Accuracy	0.575	0.533	0.641	0.639	0.344	0.637	0.639	0.638	0.629	0.634
Average Loss	0.334	0.515	0.129	0.129	1.289	0.157	0.148	0.189	0.165	0.130
BLEU Score	30.99	26.39	36.13	36.52	11.98	34.77	35.74	38.67	38.77	39.06

Thus summary of the models and their configurations, accuracies, losses, bleu score, outputs, web application outputs, and comparison with existent approaches are discussed.

CHAPTER 6
FUTURE WORK AND CONCLUSION

6.1 FUTURE WORK:

There is always room for improvements in any app. Right now, we are just dealing with text communication. There are several chat apps which serve similar purpose as this project, but these apps were rather difficult to use and provide confusing interfaces. The created private chat system promises to contribute towards obtaining organizational competitive advantage. It is believed that the successful deployment of this system will aid faster and secure communication among users. A positive first impression is essential in human relationship as well as in human computer interaction. Use different attention mechanism like Luong attention mechanism and can try different parameters to improve the performance of the chatbot and to deploy the model using cloud services so that the organizations could benefit from our application.

6.2 CONCLUSION:

Various techniques and algorithm were discussed. The performance of the training was analyzed with the help of automatic evaluation metrices and by comparing output responses for a set of source utterances. We can also try different combination of hyperparameters. The training of Cornell movie subtitles corpus produced result which needs further improvement and attention on training parameters. We can also judge the quality of dataset by using another similar dataset with the Hyperparameters we tried for cornel data.

APPENDIX I

SIMULATION ENVIRONMENT

GOOGLE COLABORATORY:

Google Colaboratory sometimes called Colaboratory for short, is a Google cloud-based service that replicates Jupyter Notebook in the cloud. You don't have to install anything on your system to use it. In most respects, you use Colaboratory as you would a desktop installation of Jupyter Notebook. Google Colaboratory is primarily for those readers who use something other than a standard desktop setup to work through the examples.

You can use Colaboratory to perform many tasks, such as to write and run code, create its associated documentation, and display graphics, just as you do with Jupyter Notebook. The techniques you use are similar, in fact, to using Jupyter Notebook, but there are small differences between the two.

Colaboratory supports a number of online storage options, so you can regard Colaboratory as your online partner in creating Python code.

The other reason about Colaboratory is that you can use it with your alternative device. During the writing process, some of the example code was tested on an Android-based tablet (an ASUS ZenPad 3S 10). The target tablet has Chrome installed and executes the code well enough to follow the examples. All this said, you likely won't want to try to write code using a tablet of that size — the text was incredibly small, for one thing, and the lack of a keyboard could be a problem, too. The point is that you don't absolutely have to have a Windows, Linux, or OS X system to try the code, but the alternatives might not provide quite the performance you expect.

FLASK:

Flask is a web application framework written in Python. Armin Ronacher, who leads an international group of Python enthusiasts named Pocco, develops it. Flask is based on Werkzeug WSGI toolkit and Jinja2 template engine. Both are Pocco projects. Importing flask module in the project is mandatory.

Werkzeug

It is a WSGI toolkit, which implements requests, response objects, and other utility functions. This enables building a web framework on top of it. The Flask framework uses Werkzeug as one of its bases.

Jinja 2

Jinja2 is a popular templating engine for Python. A web templating system combines a template with a certain data source to render dynamic web pages.

Flask is often referred to as a micro framework. It aims to keep the core of an application simple yet extensible. Flask does not have built-in abstraction layer for database handling, nor does it have form a validation support. Instead, Flask supports the extensions to add such functionality to the application.

Tensorflow:

TensorFlow is one of the most used open-source frameworks for developing Machine Learning and AI-equipped models. It provides multiple libraries, packages, and tools that help developers build robust applications powered by Machine Learning and Artificial Intelligence.

TensorFlow helps us train and execute neural network image recognition, natural language processing, digit classification, and many more. Also, using the same models used for development, TensorFlow facilitates the estimation of the output at various scales.

The main objective of using TensorFlow is not just the development of a deep neural network. But, it is focused to reduce the complexity of implementing computations on large numerical datasets. Since Deep Learning models require a lot of computation for attaining accuracy, companies started using TensorFlow. Thus, Google made TensorFlow available to everyone.

One of the best things about TensorFlow is it provides a feature that helps us create structures for our Machine Learning models. These structures are made of dataflow graphs. The dataflow graphs denote the functionalities that we want to implement. It consists of a set of nodes in a well-defined order where we can specify the methods for computation.

Also, the dataflow graphs show us how the data moves through the graph, including its functioning. The above diagram gives more clarity about the mechanism of TensorFlow.

However, the data that we need to feed into the model should be a multidimensional array while using TensorFlow for our applications. These multidimensional arrays are known as tensors, and they are very helpful while dealing with massive amounts of data.

In a graph, every node will represent a mathematical operation, while each connection or the edge between nodes will be a multidimensional data array (tensor).

Languages and text editor:

In this project we solely used python as our programming language with html, css, js as our front-end scripting languages for building the application. We have used sublime text editor in our desktop environment to execute our flask application and to build our web application.

So these are the technology stacks and simulation environments that we have used to build our project.

APPENDIX II

SCREENSHOTS

Fig A1 Encoder-Decoder Model:

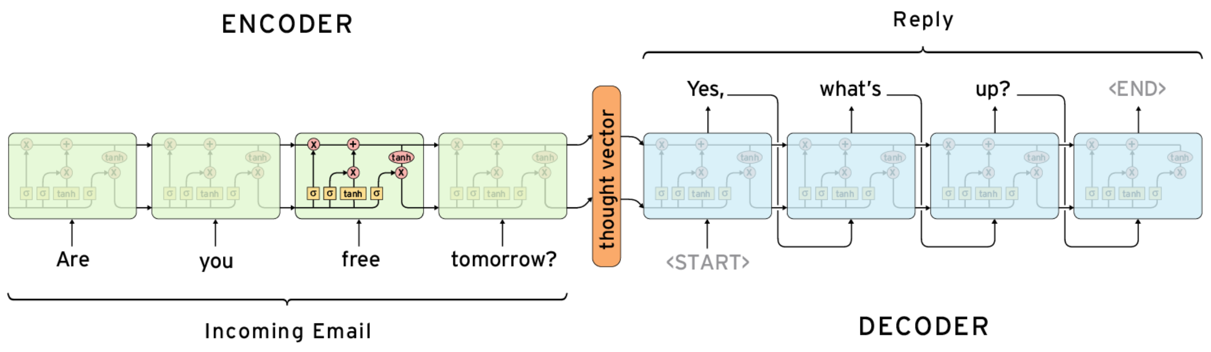


Fig A2 Encoder-Decoder Model (Greedy):

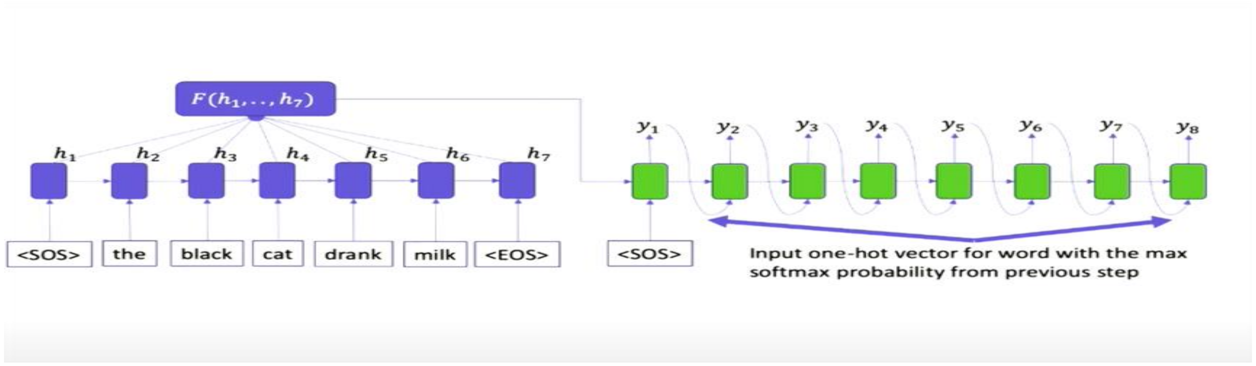
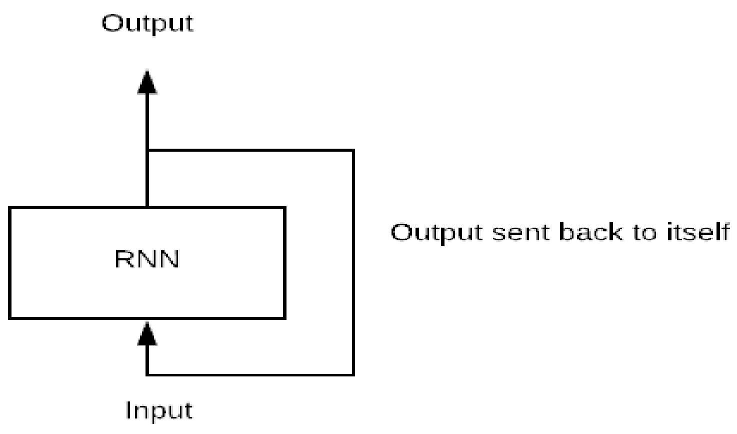

$F(h_1, ..., h_7)$
h_1 h_2 h_3 h_4 h_5 h_6 h_7
y_1 y_2 y_3 y_4 y_5 y_6 y_7 y_8
<SOS>
the
black
cat
drank
milk
<EOS>
<SOS>
Input one-hot vector for word with the max softmax probability from previous step

Fig A3 Recurrent Neural Network (RNN):



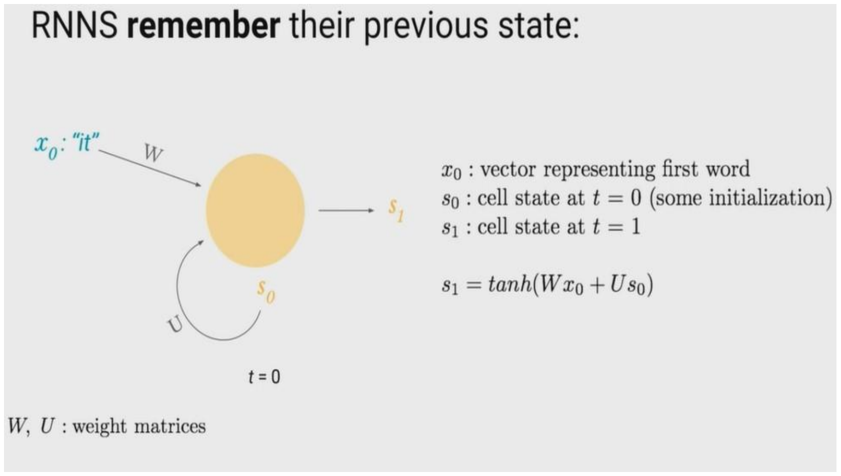

RNNS **remember** their previous state:
x_0: "it"
W
s_1
s_0
U
$t = 0$
x_0 : vector representing first word
s_0 : cell state at $t = 0$ (some initialization)
s_1 : cell state at $t = 1$
$s_1 = tanh(Wx_0 + Us_0)$
$W,\ U$: weight matrices

Fig A4 Types of RNN:

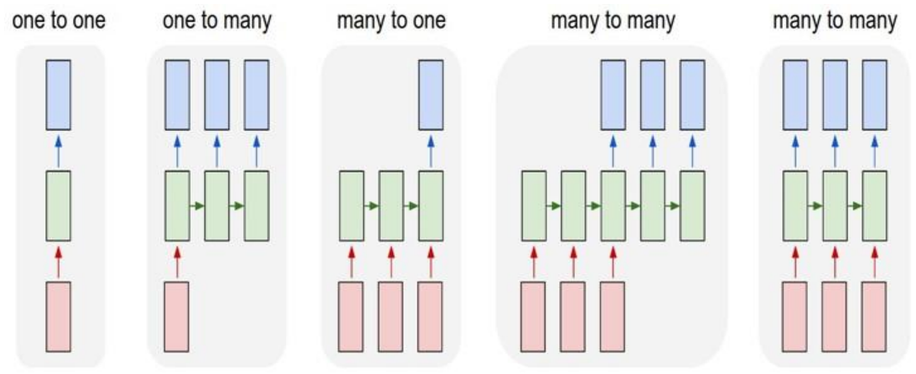


Fig A5 Bidirectional RNN:

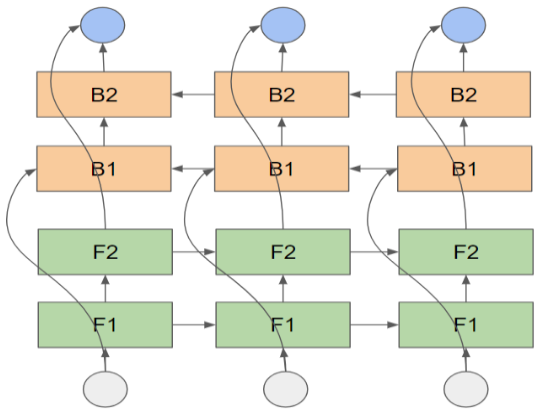
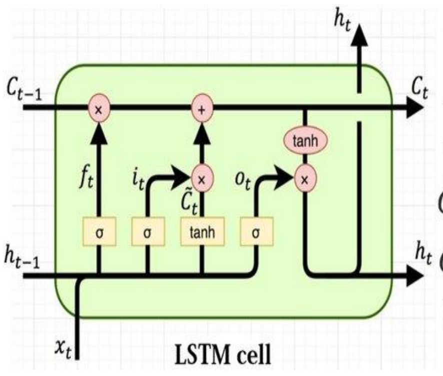


Fig A6 Long Short-Term Memory (LSTM):



$$i_t = \sigma\left(x_t U^i + h_{t-1} W^i\right)$$
$$f_t = \sigma\left(x_t U^f + h_{t-1} W^f\right)$$
$$o_t = \sigma\left(x_t U^o + h_{t-1} W^o\right)$$
$$\tilde{C}_t = \tanh\left(x_t U^g + h_{t-1} W^g\right)$$
$$C_t = \sigma\left(f_t * C_{t-1} + i_t * \tilde{C}_t\right)$$
$$h_t = \tanh(C_t) * o_t$$

Fig A7 LSTM Architecture:

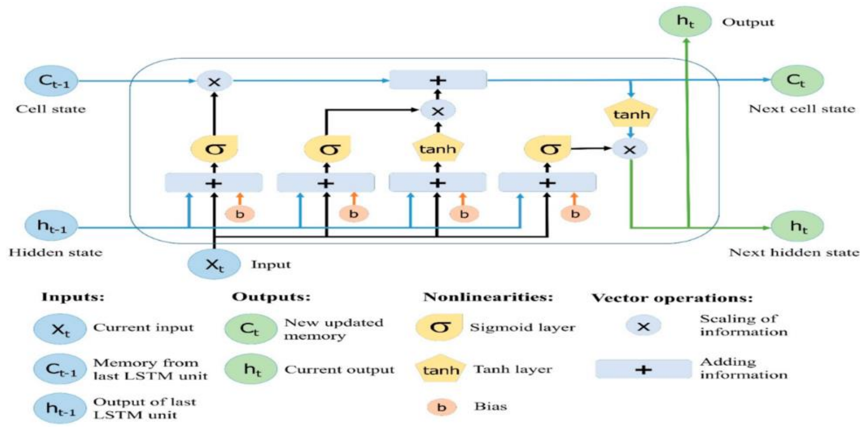


Fig A8 Bidirectional LSTM:

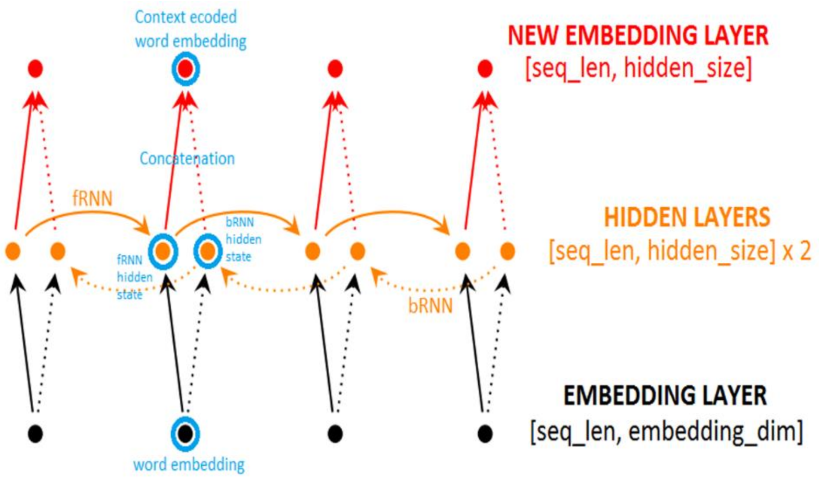


Fig A9 Embedding Lookup:

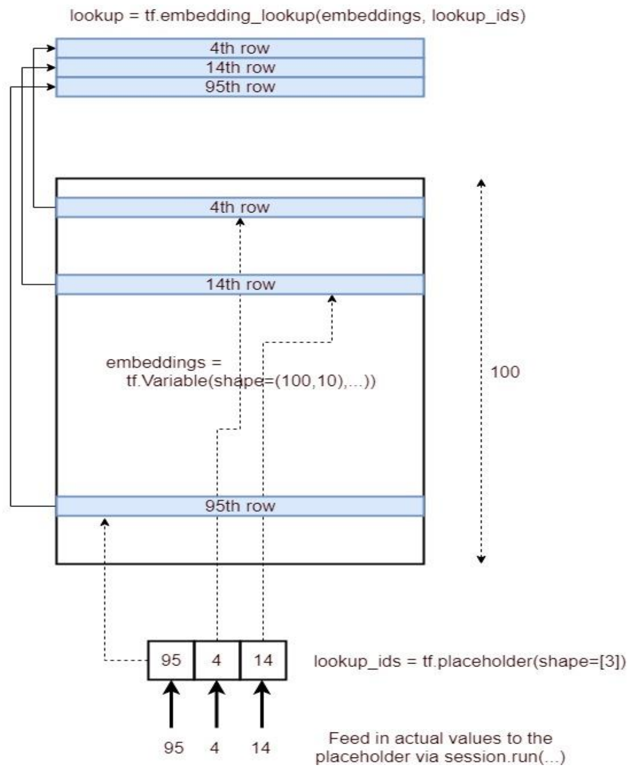
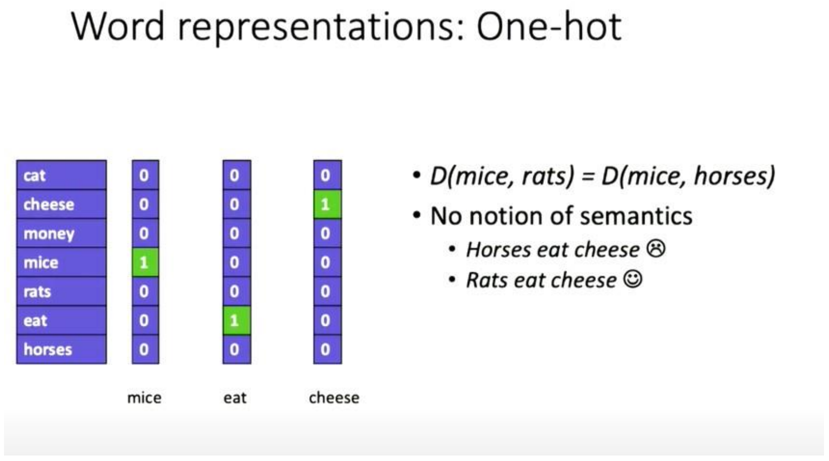

lookup = tf.embedding_lookup(embeddings, lookup_ids)
4th row
14th row
95th row
4th row
14th row
embeddings =
tf.Variable(shape=(100,10),...))
100
95th row
95
4
14
lookup_ids = tf.placeholder(shape=[3])
95
4
14
Feed in actual values to the
placeholder via session.run(...)

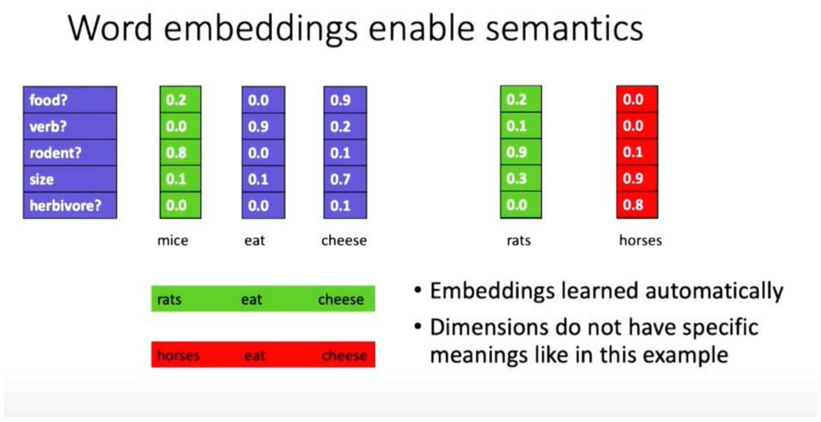
Fig A10 Word Representations: One-Hot:


Word representations: One-hot

	mice	eat	cheese
cat	0	0	0
cheese	0	0	1
money	0	0	0
mice	1	0	0
rats	0	0	0
eat	0	1	0
horses	0	0	0

- *D(mice, rats) = D(mice, horses)*
- No notion of semantics
  - *Horses eat cheese* ☹
  - *Rats eat cheese* ☺

Fig A11 Word Embeddings:



Word embeddings enable semantics

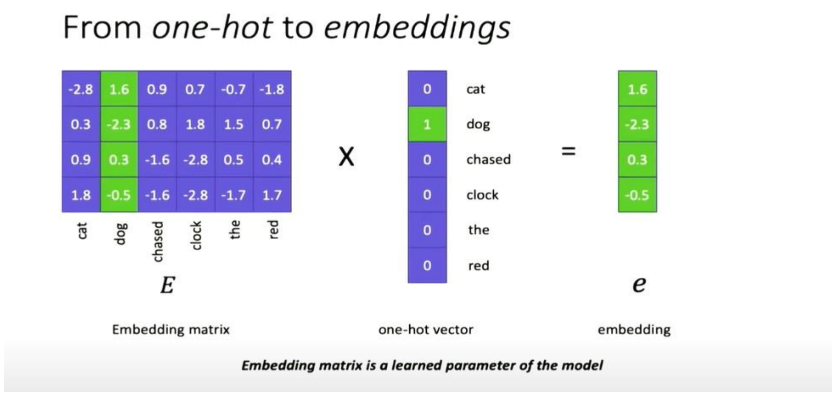
	mice	eat	cheese	rats	horses
food?	0.2	0.0	0.9	0.2	0.0
verb?	0.0	0.9	0.2	0.1	0.0
rodent?	0.8	0.0	0.1	0.9	0.1
size	0.1	0.1	0.7	0.3	0.9
herbivore?	0.0	0.0	0.1	0.0	0.8

rats eat cheese

horses eat cheese

- Embeddings learned automatically
- Dimensions do not have specific meanings like in this example

Fig A12 One-Hot to Word Embeddings:



From *one-hot* to *embeddings*

$$\underbrace{\begin{array}{cccccc} -2.8 & 1.6 & 0.9 & 0.7 & -0.7 & -1.8 \\ 0.3 & -2.3 & 0.8 & 1.8 & 1.5 & 0.7 \\ 0.9 & 0.3 & -1.6 & -2.8 & 0.5 & 0.4 \\ 1.8 & -0.5 & -1.6 & -2.8 & -1.7 & 1.7 \\ \text{cat} & \text{dog} & \text{chased} & \text{clock} & \text{the} & \text{red} \end{array}}_{E} \times \underbrace{\begin{array}{cc} 0 & \text{cat} \\ 1 & \text{dog} \\ 0 & \text{chased} \\ 0 & \text{clock} \\ 0 & \text{the} \\ 0 & \text{red} \end{array}}_{} = \underbrace{\begin{array}{c} 1.6 \\ -2.3 \\ 0.3 \\ -0.5 \end{array}}_{e}$$

Embedding matrix

one-hot vector

embedding

Embedding matrix is a learned parameter of the model

Fig A13 Bahanau Attention:

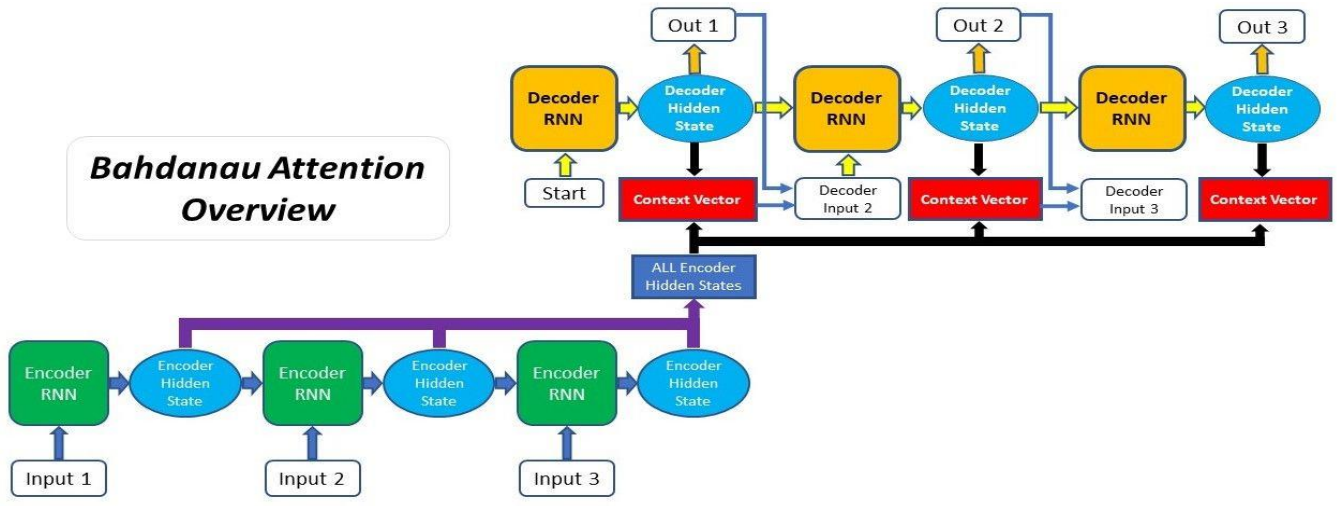
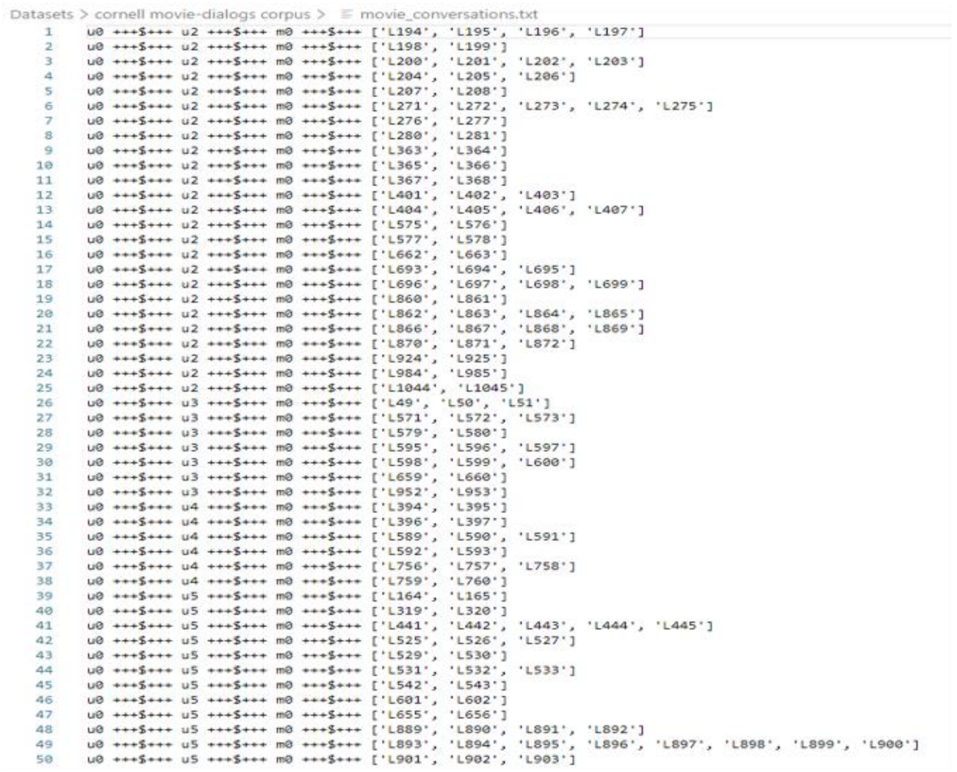


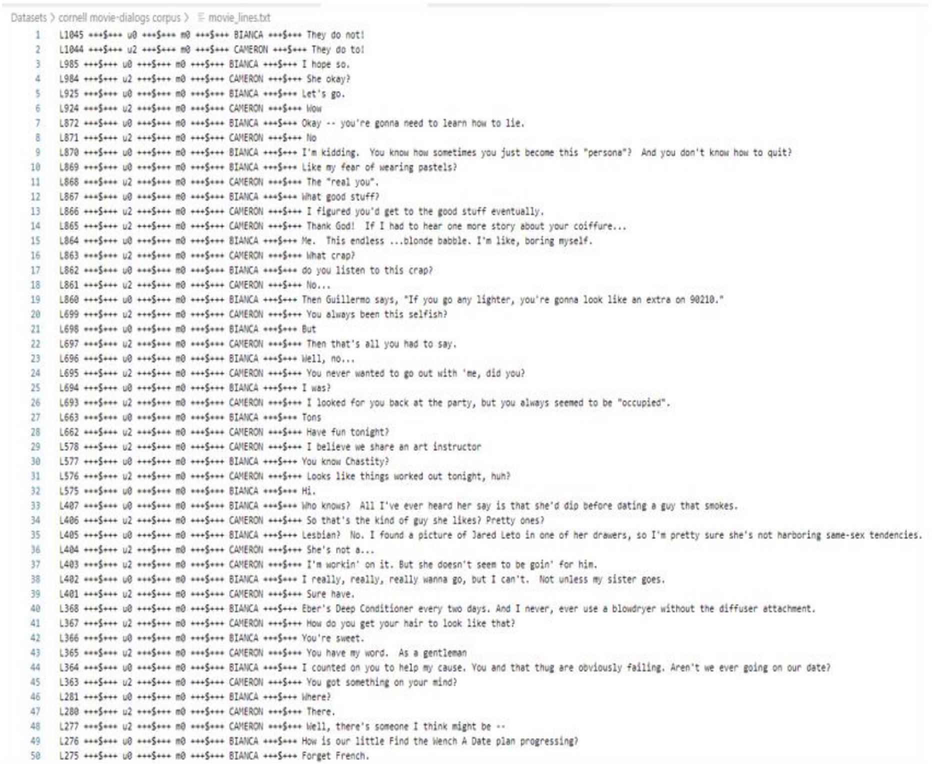
Fig A13 Dataset 1:



Datasets > cornell movie-dialogs corpus > movie_conversations.txt

```
u0 +++$+++ u2 +++$+++ m0 +++$+++ ['L194', 'L195', 'L196', 'L197']
u0 +++$+++ u2 +++$+++ m0 +++$+++ ['L198', 'L199']
u0 +++$+++ u2 +++$+++ m0 +++$+++ ['L200', 'L201', 'L202', 'L203']
u0 +++$+++ 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 +++$+++ ['L1044', 'L1045']
u0 +++$+++ u3 +++$+++ m0 +++$+++ ['L49', 'L50', 'L51']
u0 +++$+++ u3 +++$+++ m0 +++$+++ ['L571', 'L572', 'L573']
u0 +++$+++ u3 +++$+++ m0 +++$+++ ['L579', 'L580']
u0 +++$+++ u3 +++$+++ m0 +++$+++ ['L595', 'L596', 'L597']
u0 +++$+++ u3 +++$+++ m0 +++$+++ ['L598', 'L599', 'L600']
u0 +++$+++ u3 +++$+++ m0 +++$+++ ['L659', 'L660']
u0 +++$+++ u3 +++$+++ m0 +++$+++ ['L952', 'L953']
u0 +++$+++ u4 +++$+++ m0 +++$+++ ['L394', 'L395']
u0 +++$+++ u4 +++$+++ m0 +++$+++ ['L396', 'L397']
u0 +++$+++ u4 +++$+++ m0 +++$+++ ['L589', 'L590', 'L591']
u0 +++$+++ u4 +++$+++ m0 +++$+++ ['L592', 'L593']
u0 +++$+++ u4 +++$+++ m0 +++$+++ ['L756', 'L757', 'L758']
u0 +++$+++ u4 +++$+++ m0 +++$+++ ['L759', 'L760']
u0 +++$+++ u5 +++$+++ m0 +++$+++ ['L164', 'L165']
u0 +++$+++ u5 +++$+++ m0 +++$+++ ['L319', 'L320']
u0 +++$+++ u5 +++$+++ m0 +++$+++ ['L441', 'L442', 'L443', 'L444', 'L445']
u0 +++$+++ u5 +++$+++ m0 +++$+++ ['L525', 'L526', 'L527']
u0 +++$+++ u5 +++$+++ m0 +++$+++ ['L529', 'L530']
u0 +++$+++ u5 +++$+++ m0 +++$+++ ['L531', 'L532', 'L533']
u0 +++$+++ u5 +++$+++ m0 +++$+++ ['L542', 'L543']
u0 +++$+++ u5 +++$+++ m0 +++$+++ ['L601', 'L602']
u0 +++$+++ u5 +++$+++ m0 +++$+++ ['L655', 'L656']
u0 +++$+++ u5 +++$+++ m0 +++$+++ ['L889', 'L890', 'L891', 'L892']
u0 +++$+++ u5 +++$+++ m0 +++$+++ ['L893', 'L894', 'L895', 'L896', 'L897', 'L898', 'L899', 'L900']
u0 +++$+++ u5 +++$+++ m0 +++$+++ ['L901', 'L902', 'L903']
```

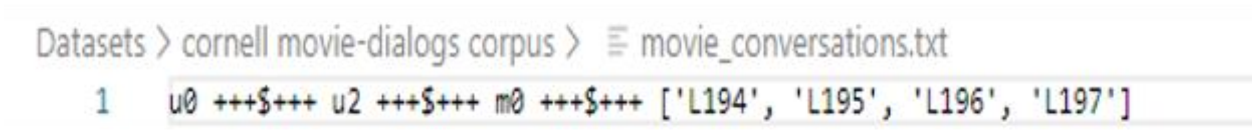
Fig A14 Dataset 2:



Datasets > cornell movie-dialogs corpus > movie_lines.txt

```
L1045 +++$+++ u0 +++$+++ m0 +++$+++ BIANCA +++$+++ They do not!
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 you don't know how to quit?
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 extra on 90210."
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?
L693 +++$+++ u2 +++$+++ m0 +++$+++ CAMERON +++$+++ I looked for you back at the party, but you always seemed to be "occupied".
L663 +++$+++ u0 +++$+++ m0 +++$+++ BIANCA +++$+++ Tons
L662 +++$+++ u2 +++$+++ m0 +++$+++ CAMERON +++$+++ Have fun tonight?
L578 +++$+++ u2 +++$+++ m0 +++$+++ CAMERON +++$+++ I believe we share an art instructor
L577 +++$+++ u0 +++$+++ m0 +++$+++ BIANCA +++$+++ You know Chastity?
L576 +++$+++ u2 +++$+++ m0 +++$+++ CAMERON +++$+++ Looks like things worked out tonight, huh?
L575 +++$+++ u0 +++$+++ m0 +++$+++ BIANCA +++$+++ Hi.
L407 +++$+++ u0 +++$+++ m0 +++$+++ BIANCA +++$+++ Who knows?  All I've ever heard her say is that she'd dip before dating a guy that smokes.
L406 +++$+++ u2 +++$+++ m0 +++$+++ CAMERON +++$+++ So that's the kind of guy she likes? Pretty ones?
L405 +++$+++ u0 +++$+++ m0 +++$+++ BIANCA +++$+++ Lesbian?  No. I found a picture of Jared Leto in one of her drawers, so I'm pretty sure she's not harboring same-sex tendencies.
L404 +++$+++ u2 +++$+++ m0 +++$+++ CAMERON +++$+++ She's not a...
L403 +++$+++ u2 +++$+++ m0 +++$+++ CAMERON +++$+++ I'm workin' on it. But she doesn't seem to be goin' for him.
L402 +++$+++ u0 +++$+++ m0 +++$+++ BIANCA +++$+++ I really, really, really wanna go, but I can't.  Not unless my sister goes.
L401 +++$+++ u2 +++$+++ m0 +++$+++ CAMERON +++$+++ Sure have.
L368 +++$+++ u0 +++$+++ m0 +++$+++ BIANCA +++$+++ Eber's Deep Conditioner every two days. And I never, ever use a blowdryer without the diffuser attachment.
L367 +++$+++ u2 +++$+++ m0 +++$+++ CAMERON +++$+++ How do you get your hair to look like that?
L366 +++$+++ u0 +++$+++ m0 +++$+++ BIANCA +++$+++ You're sweet.
L365 +++$+++ u2 +++$+++ m0 +++$+++ CAMERON +++$+++ You have my word.  As a gentleman
L364 +++$+++ u0 +++$+++ m0 +++$+++ BIANCA +++$+++ I counted on you to help my cause. You and that thug are obviously failing. Aren't we ever going on our date?
L363 +++$+++ u2 +++$+++ m0 +++$+++ CAMERON +++$+++ You got something on your mind?
L281 +++$+++ u0 +++$+++ m0 +++$+++ BIANCA +++$+++ Where?
L280 +++$+++ u2 +++$+++ m0 +++$+++ CAMERON +++$+++ There.
L277 +++$+++ u2 +++$+++ m0 +++$+++ CAMERON +++$+++ Well, there's someone I think might be --
L276 +++$+++ u0 +++$+++ m0 +++$+++ BIANCA +++$+++ How is our little Find the Wench A Date plan progressing?
L275 +++$+++ u0 +++$+++ m0 +++$+++ BIANCA +++$+++ Forget French.
```

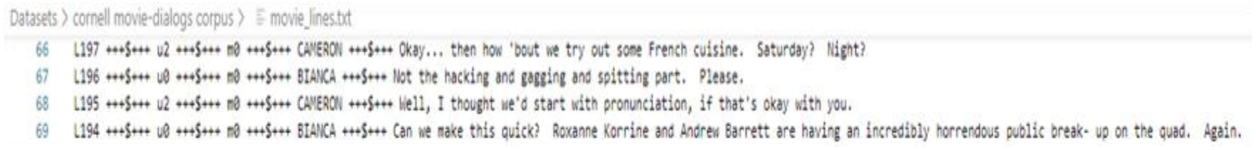
Fig A15 Dataset 3:



Datasets > cornell movie-dialogs corpus > movie_conversations.txt

```
u0 +++$+++ u2 +++$+++ m0 +++$+++ ['L194', 'L195', 'L196', 'L197']
```

Fig A16 Dataset 4:



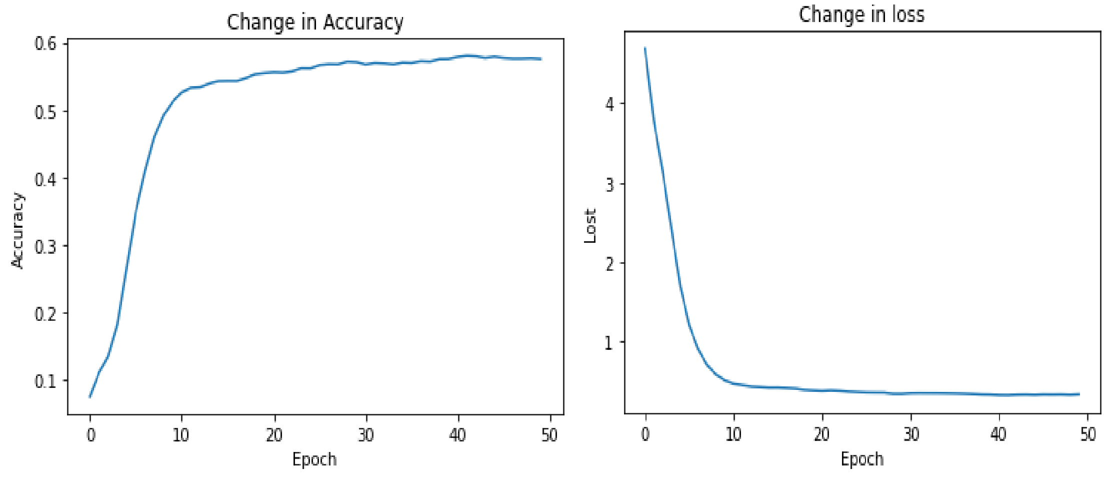
Datasets > cornell movie-dialogs corpus > movie_lines.txt

```
L197 +++$+++ u2 +++$+++ m0 +++$+++ CAMERON +++$+++ Okay... then how 'bout we try out some French cuisine.  Saturday?  Night?
L196 +++$+++ u0 +++$+++ m0 +++$+++ BIANCA +++$+++ Not the hacking and gagging and spitting part.  Please.
L195 +++$+++ u2 +++$+++ m0 +++$+++ CAMERON +++$+++ Well, I thought we'd start with pronunciation, if that's okay with you.
L194 +++$+++ u0 +++$+++ m0 +++$+++ BIANCA +++$+++ Can we make this quick?  Roxanne Korrine and Andrew Barrett are having an incredibly horrendous public break- up on the quad.  Again.
```

Accuracy, Losses, Epochs and Predictions:

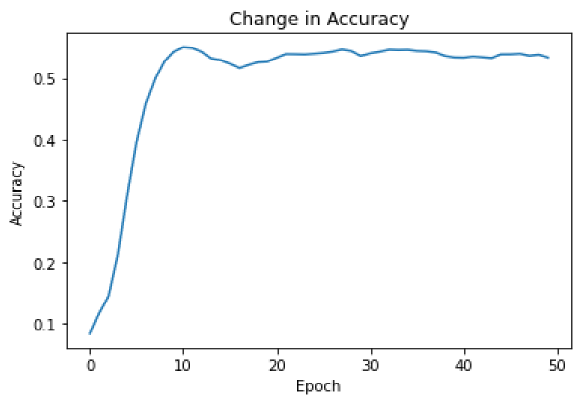
Configuration 1:

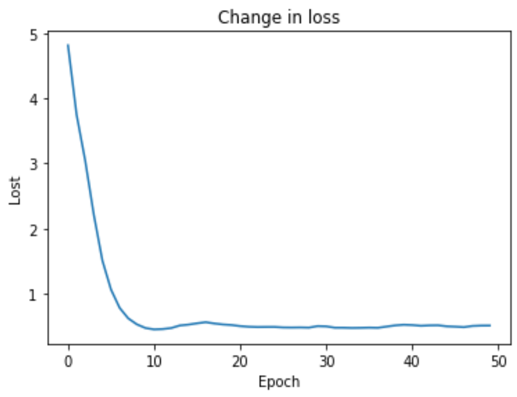
Accuracy and Loss:



Configuration 2:

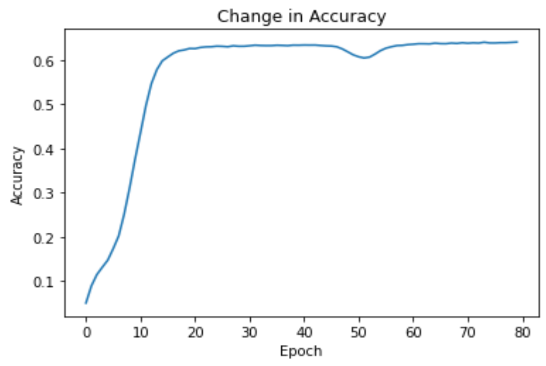
Accuracy and Loss:

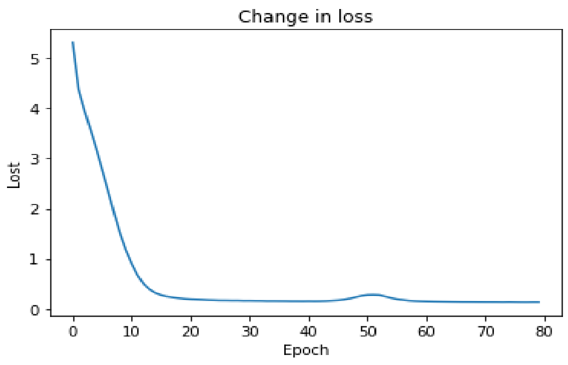




Configuration 3:

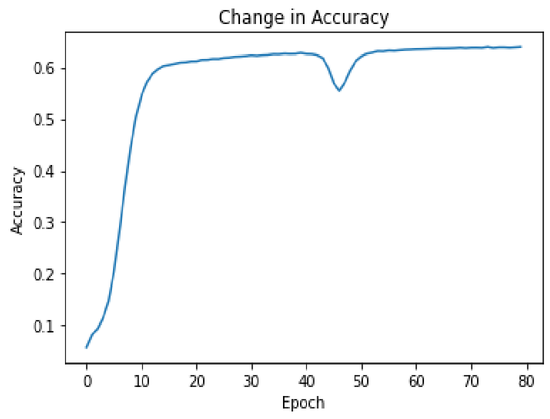
Accuracy and Loss:

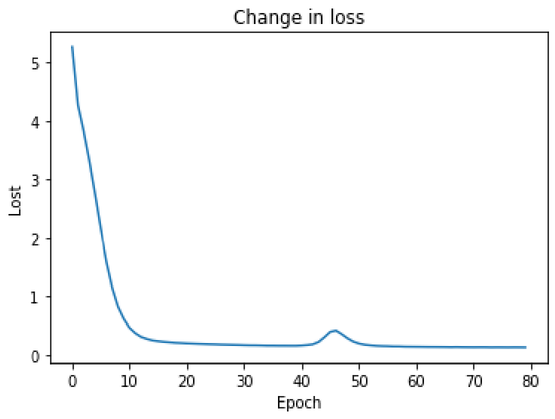




Configuration 4:

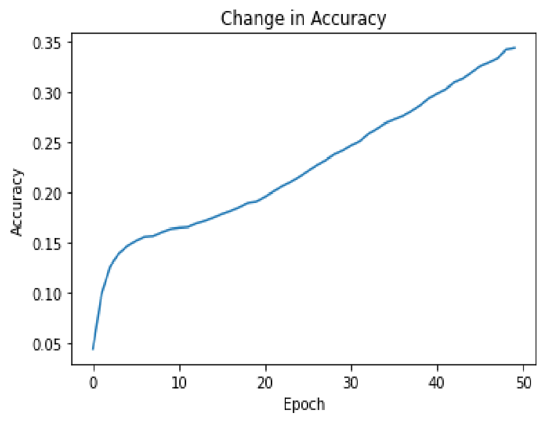
Accuracy and Loss:

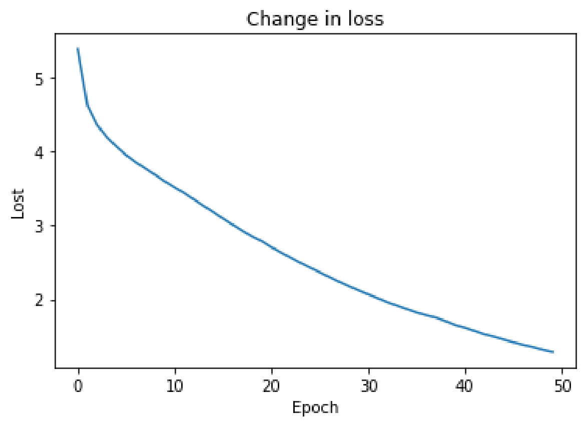




Configuration 5:

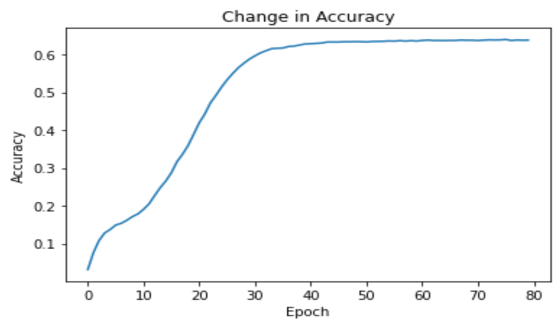
Accuracy and Loss:

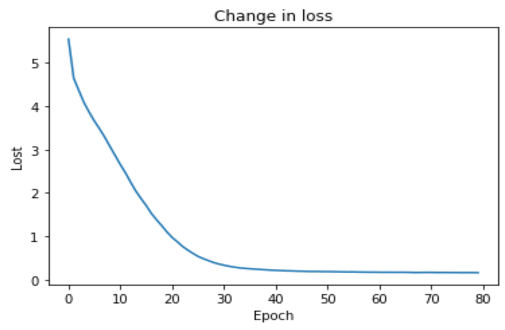




Configuration 6:

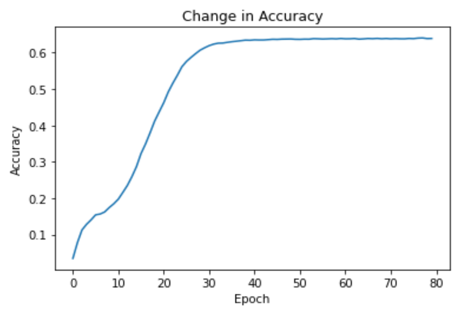
Accuracy and Loss:

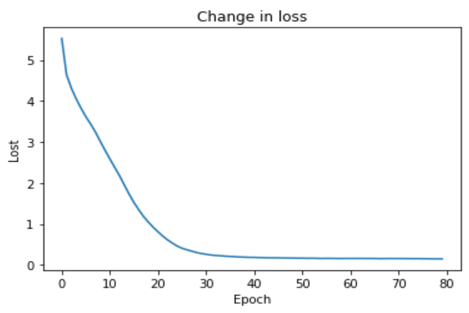




Configuration 7:

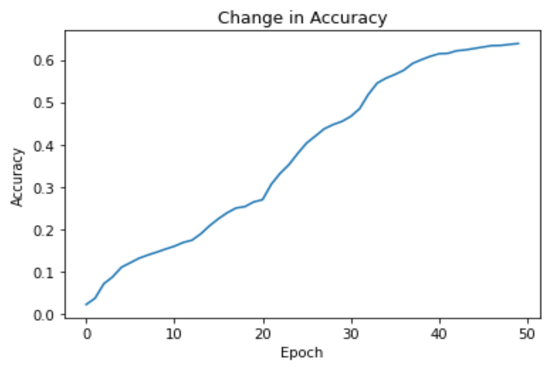
Accuracy and Loss:

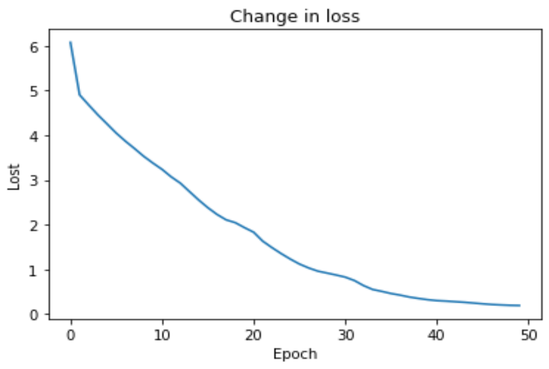




Configuration 8:

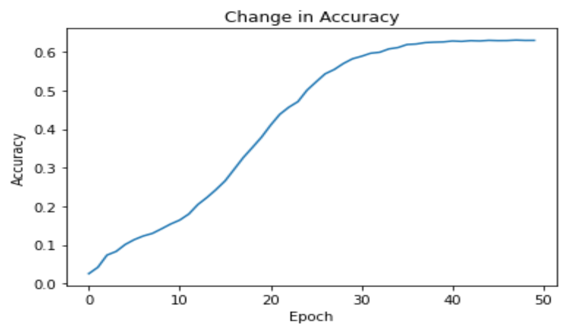
Accuracy and Loss:

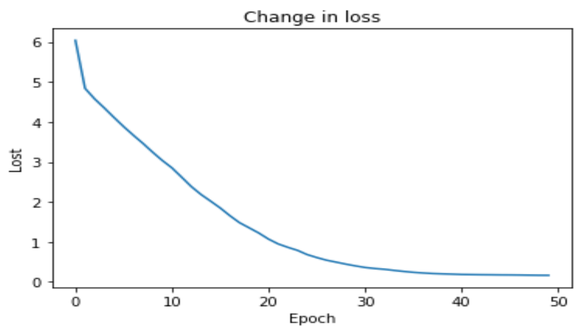




Configuration 9:

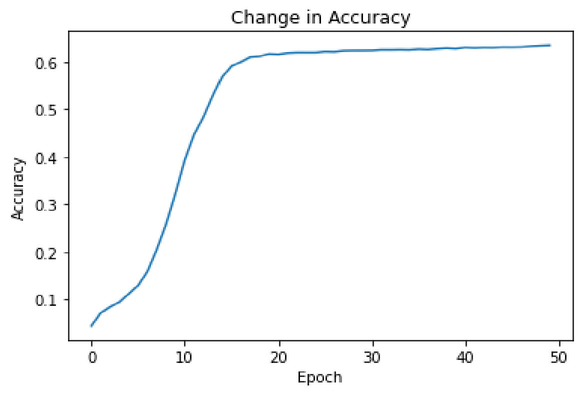
Accuracy and Loss:





Configuration 10:

Accuracy and Loss:



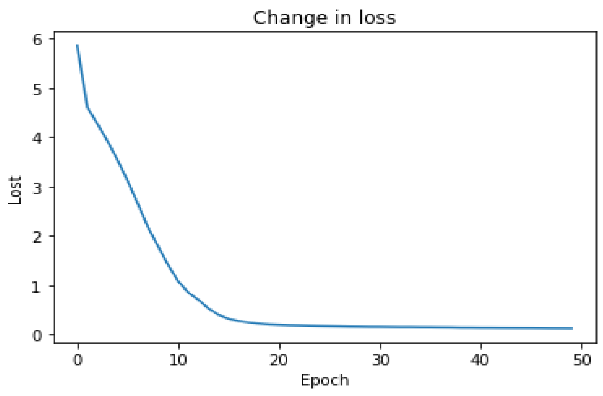
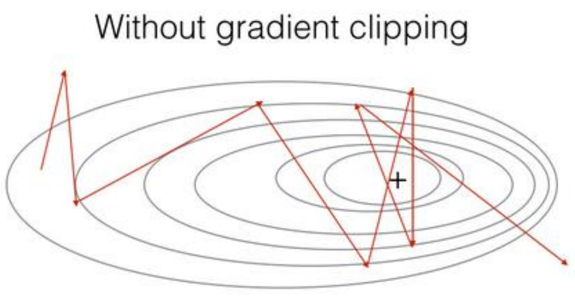


Fig A17-27 Screenshots of accuracies and loss plots of configurations

Fig A28 Gradient Clipping:



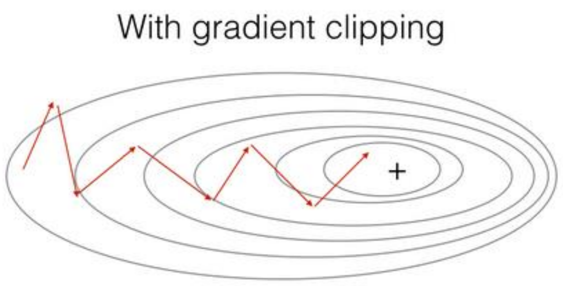
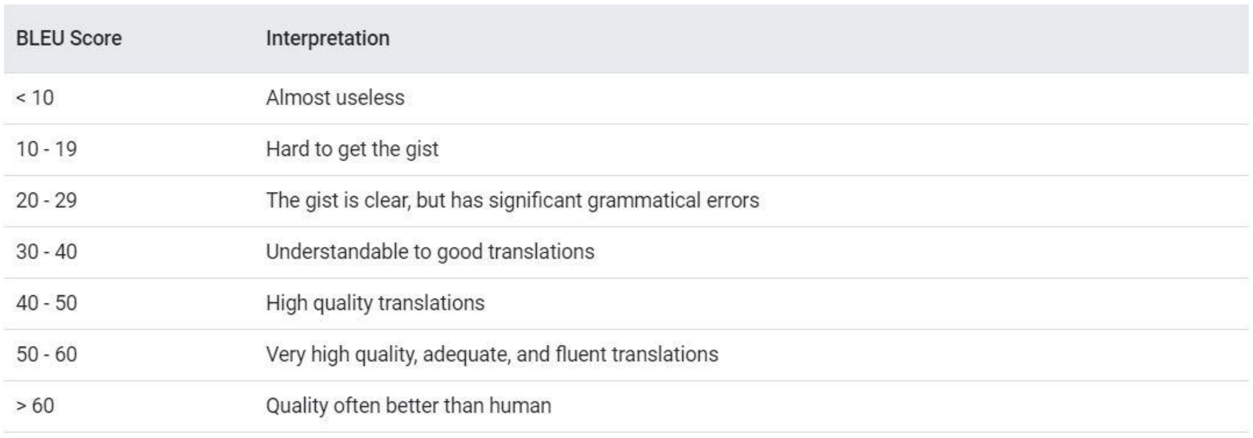


Fig A29 BLEU SCORES CATEGORY:



BLEU Score	Interpretation
< 10	Almost useless
10 - 19	Hard to get the gist
20 - 29	The gist is clear, but has significant grammatical errors
30 - 40	Understandable to good translations
40 - 50	High quality translations
50 - 60	Very high quality, adequate, and fluent translations
> 60	Quality often better than human

Fig A30 Static File Structure:

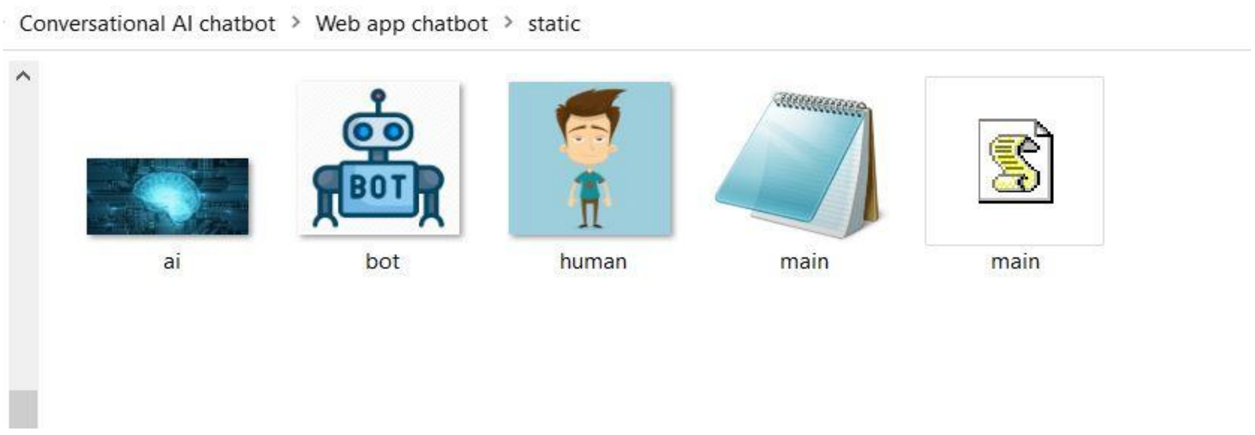


Fig A31 Images File Structure:

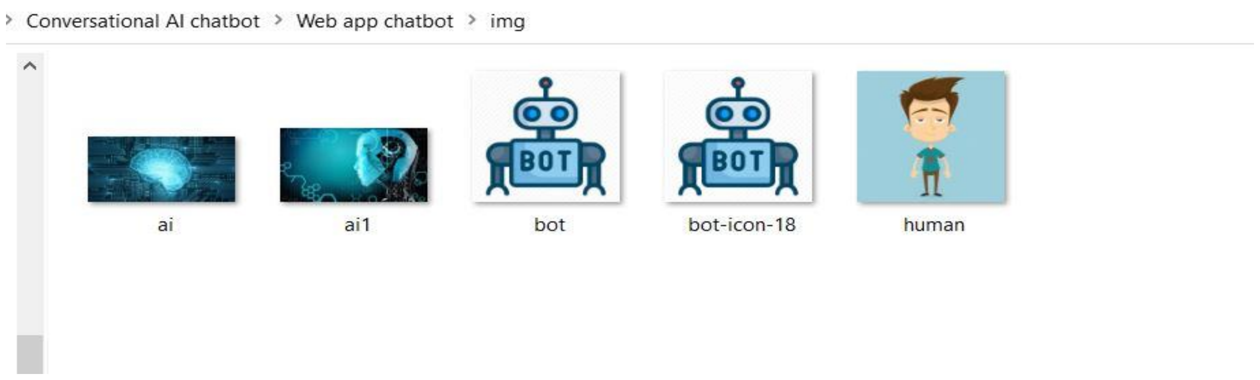
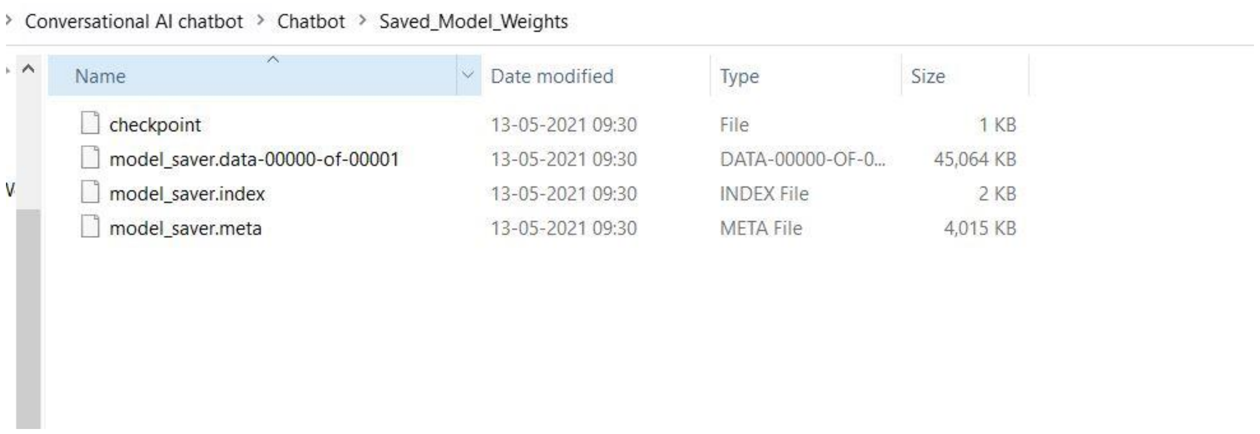
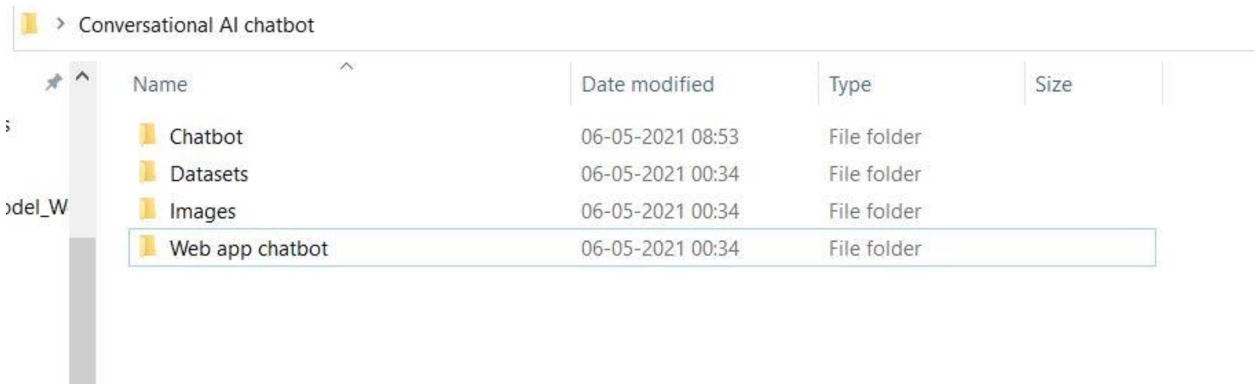


Fig A32 Model File Structure:



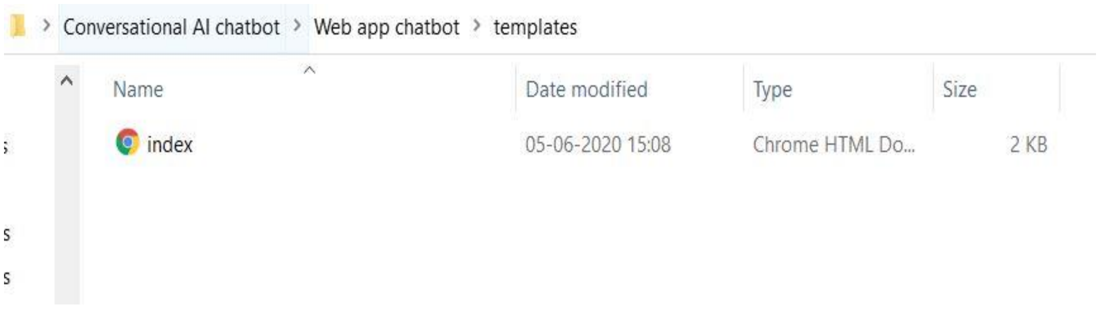
Name	Date modified	Type	Size
checkpoint	13-05-2021 09:30	File	1 KB
model_saver.data-00000-of-00001	13-05-2021 09:30	DATA-00000-OF-0...	45,064 KB
model_saver.index	13-05-2021 09:30	INDEX File	2 KB
model_saver.meta	13-05-2021 09:30	META File	4,015 KB

Fig A33 Project File Structure:



Name	Date modified	Type	Size
Chatbot	06-05-2021 08:53	File folder	
Datasets	06-05-2021 00:34	File folder	
Images	06-05-2021 00:34	File folder	
Web app chatbot	06-05-2021 00:34	File folder	

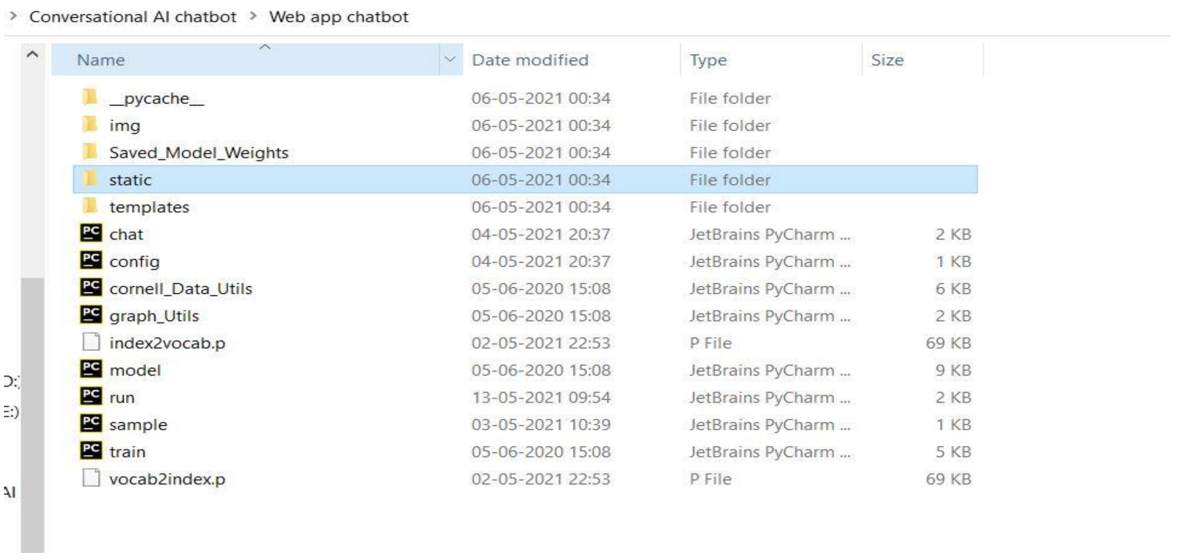
Fig A34 Template File Structure:



> Conversational AI chatbot > Web app chatbot > templates

Name	Date modified	Type	Size
index	05-06-2020 15:08	Chrome HTML Do...	2 KB

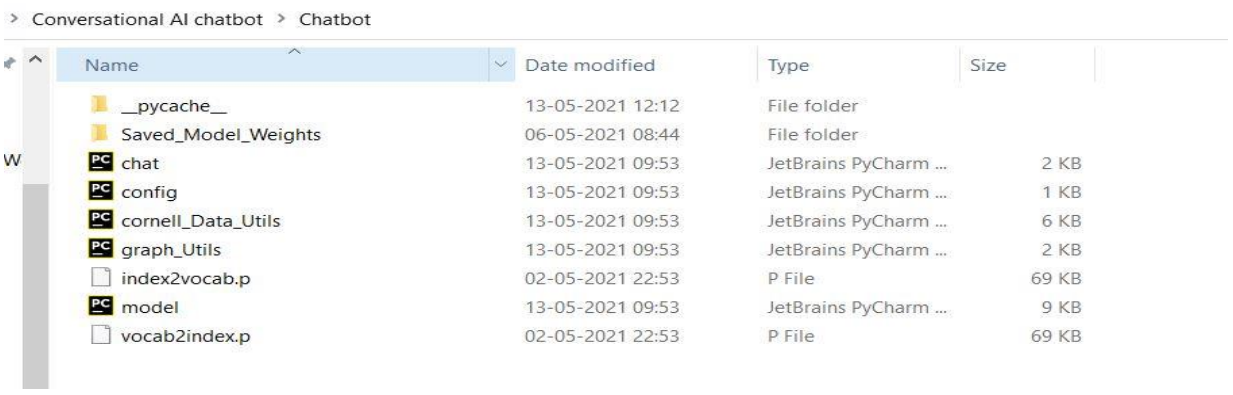
Fig A35 FLASK APP FILE STRUCTURE:



> Conversational AI chatbot > Web app chatbot

Name	Date modified	Type	Size
__pycache__	06-05-2021 00:34	File folder	
img	06-05-2021 00:34	File folder	
Saved_Model_Weights	06-05-2021 00:34	File folder	
static	06-05-2021 00:34	File folder	
templates	06-05-2021 00:34	File folder	
chat	04-05-2021 20:37	JetBrains PyCharm ...	2 KB
config	04-05-2021 20:37	JetBrains PyCharm ...	1 KB
cornell_Data_Utils	05-06-2020 15:08	JetBrains PyCharm ...	6 KB
graph_Utils	05-06-2020 15:08	JetBrains PyCharm ...	2 KB
index2vocab.p	02-05-2021 22:53	P File	69 KB
model	05-06-2020 15:08	JetBrains PyCharm ...	9 KB
run	13-05-2021 09:54	JetBrains PyCharm ...	2 KB
sample	03-05-2021 10:39	JetBrains PyCharm ...	1 KB
train	05-06-2020 15:08	JetBrains PyCharm ...	5 KB
vocab2index.p	02-05-2021 22:53	P File	69 KB

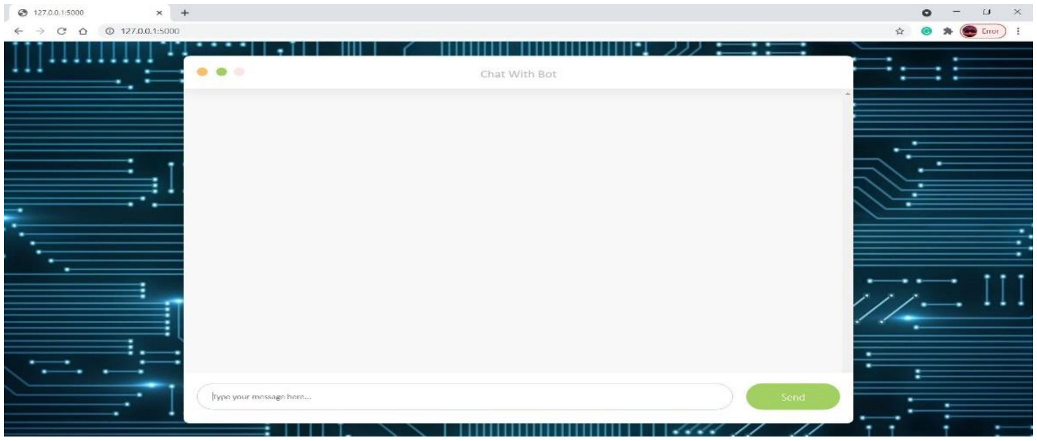
Fig A36 Chat Bot File Structure:



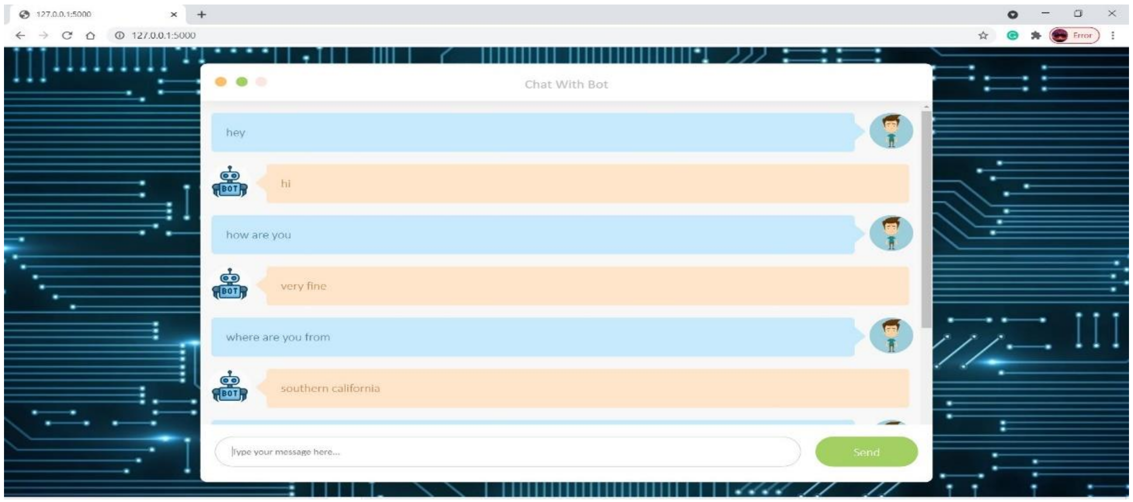
> Conversational AI chatbot > Chatbot

Name	Date modified	Type	Size
__pycache__	13-05-2021 12:12	File folder	
Saved_Model_Weights	06-05-2021 08:44	File folder	
chat	13-05-2021 09:53	JetBrains PyCharm ...	2 KB
config	13-05-2021 09:53	JetBrains PyCharm ...	1 KB
cornell_Data_Utils	13-05-2021 09:53	JetBrains PyCharm ...	6 KB
graph_Utils	13-05-2021 09:53	JetBrains PyCharm ...	2 KB
index2vocab.p	02-05-2021 22:53	P File	69 KB
model	13-05-2021 09:53	JetBrains PyCharm ...	9 KB
vocab2index.p	02-05-2021 22:53	P File	69 KB

Initial Chat Bot:



Chat Window with Chats:



hey

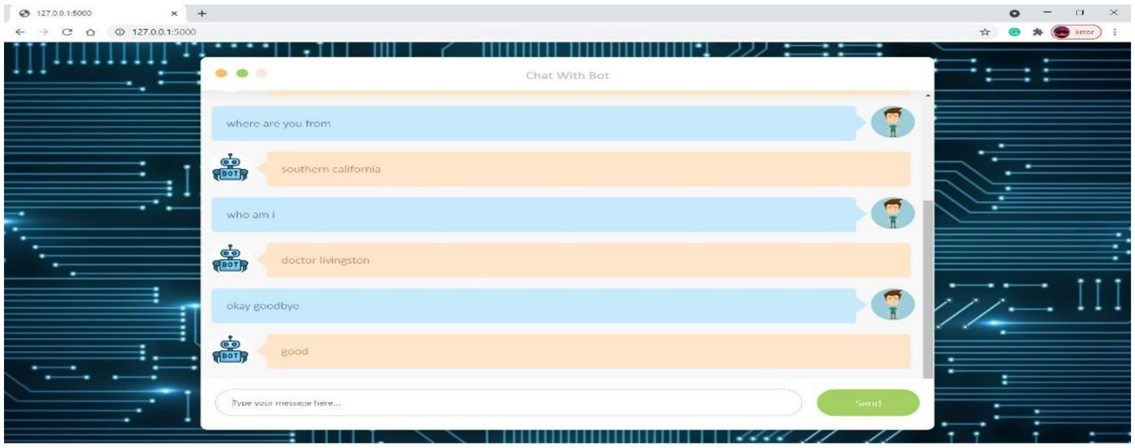
hi

how are you

very fine

where are you from

southern california



where are you from

southern california

who am i

doctor livingston

okay goodbye

good

Fig A37-A40 Screenshots of Chatbot application

REFERENCES

[1] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk,and Y. Bengio ,(2014), “Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation “,vol 1406.1078, pp 1724–1734.

[2] D. Bahdanau, K. Cho, and Y. Bengio, (2014), “Neural Machine Translation by Jointly Learning to Align and Translate”, vol 1409.0473.

[3] I. Sutskever, O. Vinyals, and Q. Le. CoRR, (2014), “Sequence to Sequence Learning with Neural Networks”, vol 1409.3215.

[4] O. Vinyals, and Q. Le, (2015), “A Neural Conversational Model”, vol 1506.05869.

[5] A. Sordoni, M. Galley, M. Auli, C. Brockett, Y. Ji, M. Mitchell, J.-Y. Nie, J. Gao, B. Dolan, (2015), “A Neural Network Approach to Context-Sensitive Generation of Conversational Responses”, vol 1506.06714, pp 196–205.

[6] K. Yao, G. Zweig, and B. Peng, (2015), “Attention with Intention for a Neural Network Conversation Model”, vol 1510.08565.