Lathwahandi Diyohan De Silva
Dr. Karen Mazidi
CS 4395.001

Chatbot Project Report

1. Introduction

The purpose of this document is to provide an overview of the project Chatbot and establish a basis of its functionalities. This project consist of a chatbot constructed via deep learning that interacts with the user in a limited manner in regards to a specific domain, in this scenario the domain is movies. Initially the paper will discuss the development methodologies used in the Chatbot project, proceeded by its functionalities, evaluation, and challenges.

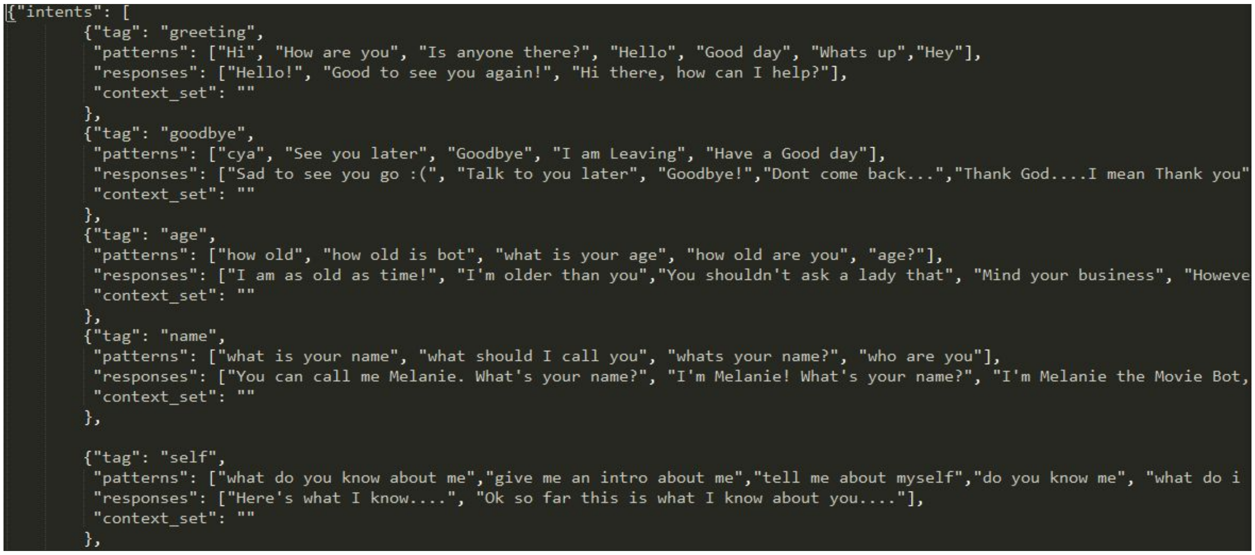
2. Development Methodology

2.1 Preprocessing

During the initial development process two input files were fed into the project to construct a knowledge base and input data for the neural network. The two files used in the model are named “movies_metadata.csv” and “chatdata.json”. In order to establish a clear notion of each input data we will look at the preprocess conducted on each module individually.

For the “movies_metadata.csv” preprocessing I implemented pandas to perform couple data cleaning processes. The movies_metadata consist of the following columns, **Adult, belongs_to_collection, budget, genres, homepage, id, imdb_id, original_language, original_title, overview, popularity, poster_path, production_companies, production_countries, Release_date, rev enue, runtime, spoken_languages. Status, tagline, title, video, vote_average, vote_count.** For the sake of the scope of the project I focused on two specific columns which is original_title and overview. Initially I took the column original_title and set it as the index form, which resulted in movie title index based dataframes, which will provide faster access to the knowledge base in conjunction to input provided by the user by performing less actions when looking up the movie title.

The “chatdata.json” file consist of input data for the neural network, and the file consists of the following form,



```
{"intents": [
        {"tag": "greeting",
         "patterns": ["Hi", "How are you", "Is anyone there?", "Hello", "Good day", "Whats up","Hey"],
         "responses": ["Hello!", "Good to see you again!", "Hi there, how can I help?"],
         "context_set": ""
        },
        {"tag": "goodbye",
         "patterns": ["cya", "See you later", "Goodbye", "I am Leaving", "Have a Good day"],
         "responses": ["Sad to see you go :(", "Talk to you later", "Goodbye!","Dont come back...","Thank God....I mean Thank you"
         "context_set": ""
        },
        {"tag": "age",
         "patterns": ["how old", "how old is bot", "what is your age", "how old are you", "age?"],
         "responses": ["I am as old as time!", "I'm older than you","You shouldn't ask a lady that", "Mind your business", "Howeve
         "context_set": ""
        },
        {"tag": "name",
         "patterns": ["what is your name", "what should I call you", "whats your name?", "who are you"],
         "responses": ["You can call me Melanie. What's your name?", "I'm Melanie! What's your name?", "I'm Melanie the Movie Bot,
         "context_set": ""
        },

        {"tag": "self",
         "patterns": ["what do you know about me","give me an intro about me","tell me about myself","do you know me", "what do i
         "responses": ["Here's what I know....", "Ok so far this is what I know about you...."],
         "context_set": ""
        },
```

Figure 1: Chat data

In order to provide necessary input data to the neural network, I initially looped through every pattern and tokenized/ stemmed the words founded. Then I proceed to append the results to a list name **word. U**pon appending the tokenized words in order to remove the duplicate values that might exist within the list, I used the set () function**.** As for each tag that represent the context of the input, I appended to a list named **label.** Furthermore, during the tokenized phase, I appended the tokens to another list named **feature.** The primary reason for creating this list is to provide a reference index for the output set, that will be used in the neural network module. Proceeding this task, I moved onto the creation of a **bag of words**, which will be used to represent the input data in the neural network model. I appended the bag to training list after converting it to a num py array. Furthermore, the output list that consists of the tags were converted into a num py array.

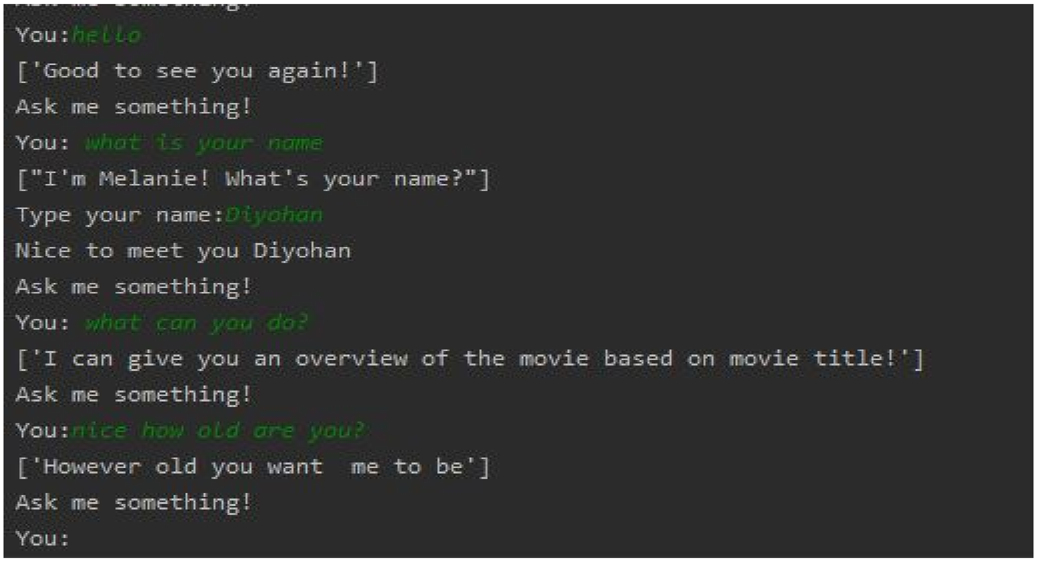
2.2 Neural Network

For the creation of the neural network model, I used Keras library, and within Keras module, I implemented a sequential model. The model consists of four layers, the initial input consist of an input shape that represents the number of tokenized words in the training data and after experimenting, I implemented the activation function **relu.** Within the input layer, and for the two hidden layers, I assigned 16 neurons in total, Due to the nature of this solution, which is a multi classification based solution, Idecided to use the activation function, **soft max**, and the output layer consists of neurons that's equivalent to the number of tags. For the lost function of the model, I used categorical cross entropy, and optimized it with adam. Furthermore, for the training runs, I had

2,000 iterations, with a batch size of eight, and a 10% validation split. Due to the nature of the data set i.e smaller dataset, I got extremely high accuracy for the model.

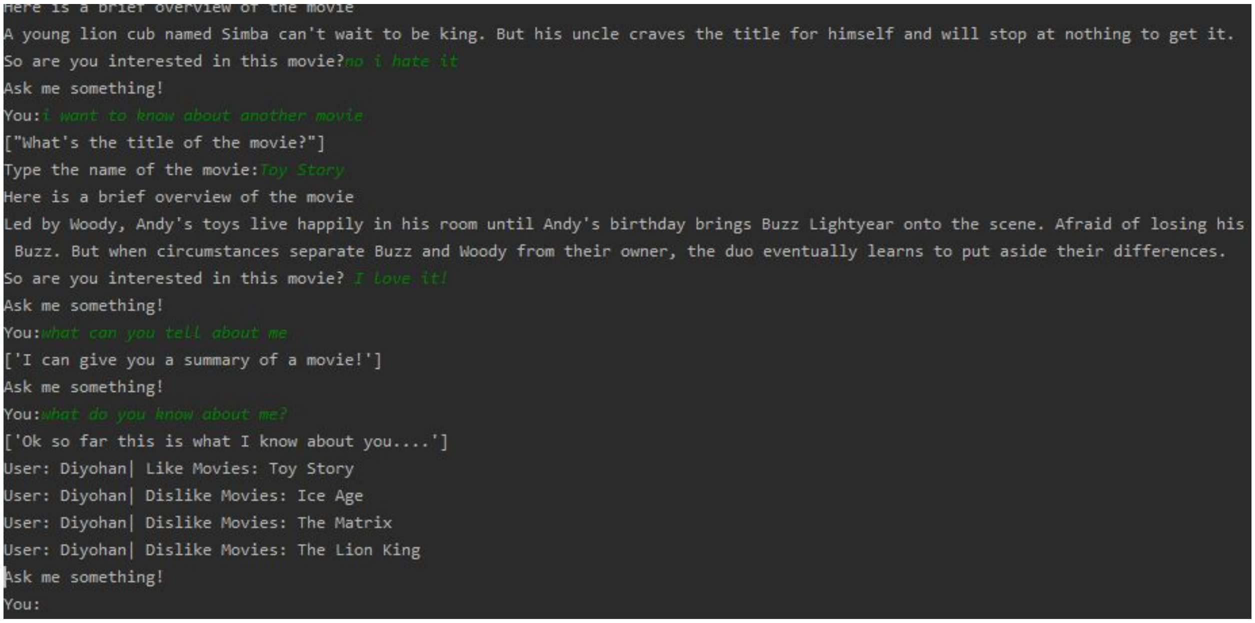
2.3 ChatBot Framework

Majority of the chatbot dialogue consists of textual and auditory output. For the auditory output, I used text to speech library known as pyttsx3. How the model worked was initially the bot will prompt the user to ask anything to obtain some form input dialog. Upon obtaining the input dialog, the input was converted into a vectorized representation i.e **bag of words**, via the function **bag words.** This bag words is sent to the neural network model to predict multi classification labeling. The model returns an array of probability that indicates the likelihood of which tag the input might belong to. Then I selected the array value with the highest probability and collect the index associated with that tag, then selects a response featured in that respective tag. Another point to be noted is, before selecting a response, I check the error threshold of the results given by the model if the bot has an 80% or higher confidence rate, only then the bot will select a response. In order to perform specific functionalities, upon obtaining the tag, the bot will check the tag name to determine if the bot needs to initiate a special instance in the chat. For the dialog, this chatbot has three special instances that will be triggered by three specific tags. They are name, search, self. The name tag represents a scenario where the bot is prompted for its name. The search tag consists of a scenario where the user asks the bot to provide information regarding a movie. While the self tag represents a scenario where the bot is prompted to talk about the user. Below consists some samples interactions that may occur between the user and the bot.



```
You:hello
['Good to see you again!']
Ask me something!
You: what is your name
["I'm Melanie! What's your name?"]
Type your name:Diyohan
Nice to meet you Diyohan
Ask me something!
You: what can you do?
['I can give you an overview of the movie based on movie title!']
Ask me something!
You:nice how old are you?
['However old you want  me to be']
Ask me something!
You:
```

Figure 2. Chatbot interaction 1



```
Here is a brief overview of the movie
A young lion cub named Simba can't wait to be king. But his uncle craves the title for himself and will stop at nothing to get it.
So are you interested in this movie?no i hate it
Ask me something!
You:i want to know about another movie
["What's the title of the movie?"]
Type the name of the movie:Toy Story
Here is a brief overview of the movie
Led by Woody, Andy's toys live happily in his room until Andy's birthday brings Buzz Lightyear onto the scene. Afraid of losing his
 Buzz. But when circumstances separate Buzz and Woody from their owner, the duo eventually learns to put aside their differences.
So are you interested in this movie? I love it!
Ask me something!
You:what can you tell about me
['I can give you a summary of a movie!']
Ask me something!
You:what do you know about me?
['Ok so far this is what I know about you....']
User: Diyohan| Like Movies: Toy Story
User: Diyohan| Dislike Movies: Ice Age
User: Diyohan| Dislike Movies: The Matrix
User: Diyohan| Dislike Movies: The Lion King
Ask me something!
You:
```

Figure 2. Movie interaction and user model overview

As you can see for the special instances the chatbot was able to provide fairly accurate responses. The chatbot employ the use of couple functions to achieve the results shown above.

Firstly we shall discuss about the movie overview, in order to gain information regarding the movie overview I employ the use of function **checkmov()**. This function takes a string value specifically the name of the movie, and searches the metadata obtained from input file “movies_metadata.csv” for the overview of the respective movie title.

Second function I’ll be discussing likemovie(), this function takes three parameters, specifically the user’s review of the movie, the name of the movie, and the name of the user. This function performs sentinel analysis on the user review string via the library sentiwordnet. Upon evaluation, a positive and negative value for the intent of the string is obtained. Whichever is the likely is appended to the respective dictionary that keeps track of user preferences.

The third function is the self() function, this takes in three parameters, which is the name of the user, dictionary containing liked movies and disliked movies. The objective of this function is to simply print user preferences obtained via interactions with theu user.

3. Evaluation

3.1 Strength and Weaknesses

Initially, there were some misclassified tags which resulted in poor responses. However, upon the optimization of the model and cleaning up the input data, the chatbot managed to respond to events classified in the input data file fairly accurately. As long as the input context is somewhat relatively similar to the original input context given by the input data. Chatbot will be able to determine the responses fairly accurately. However, due to the size of the input data sets, the versatile nature of the chatbot is somewhat limited. The way the input data/model is constructed this chatbot is easily scalable, and in order to create a more dynamic version of this chatbot, all I have to do is provide more event-driven tags, and expand on the context in the input data.

Another issue one may encounter is the limited range of movie selections. To be elaborate, the current movie database consists of 45,467 movie titles. However, if the movie is not located within the database, or if spelt incorrectly, the bot will not be able to provide and overview. Beyond that, as long as the user provides the title of the movie accurately, the user will be able to obtain a fairly accurate overview of the movie.

Lastly, the final topic I would like to address is the sentinel analysis. The metrics used to way the weight of the review can sometimes produce incorrect results. For example, when I used sentences such as, “no, I don’t like it”, due to positive weighted words such as,“like”, the sentinel analysis was not able to provide accurate feedback. Overall, the sentinel analysis perform accurately.

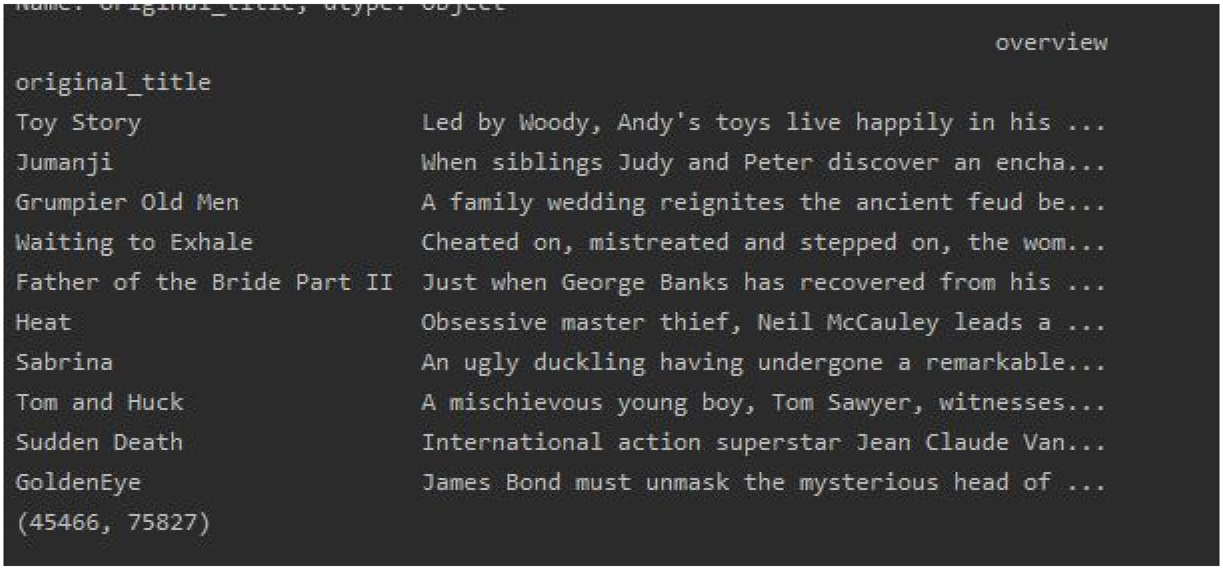
3.2 Survey Results

Based on the user’s experience, she initially didn’t know what was the basis of the chatbot. She didn’t know what to ask because it just simply displayed, “user”, rather than an instructed introduction. After giving the chatbot a prompt for the user, it was easier for her to come up with a follow up question to ask the chatbot. After asking a few more questions, she came upon another problem where the chatbot misclassified a question she had asked about herself and filtered it as a question about a movie. After fixing the mistakes, the overall experience for the user was simple and satisfying.

4. Appendix, References

4.1 Appendix

Knowledge Base Structure:



```
Name: original_title, dtype: object
                                                                          overview
original_title
Toy Story                     Led by Woody, Andy's toys live happily in his ...
Jumanji                       When siblings Judy and Peter discover an encha...
Grumpier Old Men              A family wedding reignites the ancient feud be...
Waiting to Exhale             Cheated on, mistreated and stepped on, the wom...
Father of the Bride Part II   Just when George Banks has recovered from his ...
Heat                          Obsessive master thief, Neil McCauley leads a ...
Sabrina                       An ugly duckling having undergone a remarkable...
Tom and Huck                  A mischievous young boy, Tom Sawyer, witnesses...
Sudden Death                  International action superstar Jean Claude Van...
GoldenEye                     James Bond must unmask the mysterious head of ...
(45466, 75827)
```

- **User model example can be found in Figure 2.**

4.2 References

[1] https://keras.io/getting-started/sequential-model-guide/

[2]https://towardsdatascience.com/deep-learning-for-nlp-creating-a-chatbot-with-keras-da5ca051e051

[3]https://towardsdatascience.com/chatbots-are-cool-a-framework-using-python-part-1-overview-7c69af7a7439

[4]https://www.kaggle.com/rounakbanik/movie-recommender-systems/data?source=post_page-----7c69af7a7439----------------------