TIGERBOT (Chatbot using DNN, Keras and Tkinter)

Harshavardhan Perasani
University of Memphis
Data Science
hprasani@memphis.edu

Maria Vijayanjali Yeruva
University of Memphis
Data Science
myeruva@memphis.edu

Sugnan Kumar Makkena
University of Memphis
Data Science
smkkena1@memphis.edu

Abstract

A chatbot is computer software that replicates human-like text-based conversations with users using deep learning. Chatbots are being used more frequently to improve human performance, enable communication, and deliver better faster services. It can be applied to many different tasks. In general, it must understand what the user will do and act appropriately. In this project, we used deep neural networks to build a chatbot that helps to provide information about our university. The multi-layered neural network is used to train and process the data. A Python module or program called NLTK (Natural Language Toolkit) can do symbolic and statistical Natural Language Processing for English written in programming. It is used to evaluate input and produce replies that resemble humans. To make the conversation more user-friendly we have created basic Graphical user interface using Tkinter.

1 Introduction

If developed and implemented properly, chatbots could assist us by quickly communicating up-to-date information. These applications were created to imitate human interactions. Chatbots search for keywords within the input phrase and use the query string to find relevant answers.[6] They rely on keyword similarity, and the text which is returned is taken from external or internal data sources, such as the internet or any organization's database. live chat is simple, quick, and convenient, many students use it but there are some concerns, to which the administration must spend a considerable amount of time responding. To solve this problem, this project proposes an approach for creating a question-and-answer system using chatbot technology. An efficient way to provide university information to the users is using a Chatbot created by using Deep neural networks. Deep Learning is a field of machine learning in artificial intelligence that focuses on algorithms that can learn unsupervised from unlabeled, unstructured data. It makes use of neural networks, a feature of the brain. Due to the Chatbot's open-domain nature, it could be used to create an AI assistant that can have actual conversations with its user about any topic and situation. The model is developed to perform English-to-English translation. [3]

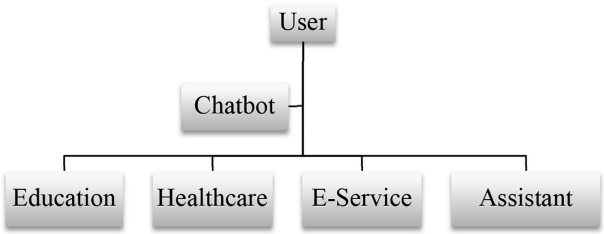

Figure 1: Applications of Chatbot

2 Related work

An overview of the technologies behind chatbots, such as Information Extraction and Deep Learning, is given in the paper "CHATBOT TECHNOLOGIES AND CHALLENGES." They discussed about how "conversational chatbots" are created using free-form chat logs, and "transactional chatbots" are created manually to carry out certain tasks, such as making an online travel reservation. They also provided an overview of available commercial tools and platforms for creating and utilizing chatbots [5].

A paper "A Medical Chatbot that Acts as a virtual Doctor" describes how an interactive chatbot for medical purposes works as a virtual doctor. This chatbot was created in Python using NLP and pattern-matching methods. [11] In a survey evaluating its performance, the chatbot correctly responded to 80 percent of the questions, while 20 percent were unclear or inaccurate. These results

indicate the possibility of using the chatbot to teach medical students as well as to serve as a virtual doctor for care and awareness.

In "Deep Reinforcement Learning for Dialogue Generation" by Dan Jurafsky, Deep Reinforcement Learning (DRL) is used to create chatbots that can have lengthy conversations. Although Seq2Seqmodel can produce logical dialogues, it can also give generic responses repeatedly regardless of input and become trapped in a loop during extended discussions. Seq2Seqmodels frequently produce responses like "I don't know" that is repeated many times. This is because generic responses appear frequently in the training set and because they work better with a wider variety of input material. Despite being successful, Dufarsky claims that the RL model is not optimized to anticipate the next utterance in this paper. This model reduced generic reactions to improve long-term rewards for longer conversations and maintain discussion. However, because there is a trade-off between relevance and less repetition, their experimentation results in less relevant responses [10]. badness ¡100¿ The Google Neural Machine Translation (GNMT) model is a module for neural machine translation into and out of English and other languages. Additionally, GNMT has been tried out for dialogue generation. It is based on the well-known Seq2Seq approach for dialogue creation. Additionally, GNMT includes a number of strategies that are essential for the development of intelligent chatbots. Sequence to Sequence modeling with encoder-decoder architecture created using unidirectional or bidirectional LSTM cells is a component of the GNMT model. Additionally, they have the choice of using Google's sub-word module to generate vocabulary, the Neural Attention Mechanism, and beam search. Additionally, users can change the hyper-parameters for better model training [14].

3 Data

The dataset is publicly not available, so we collected the data from the world wide web of the University of Memphis domain (https://www.memphis.edu/) and created the dataset. A JSON file called "intents.json" has been created and this file will contain all the text needed to develop our chatbot. The dataset contains an object called intents. There are many distinct purpose instances in the dataset, each with its own tag, context, patterns, and replies. If you carefully examine the dataset, you'll see that responses are the reactions to those interactive statements, whereas patterns are close to the interactive statements that we anticipate from our consumers. Additionally, a tag is a general description of the user's request. The main challenge is to train our bot to recognize patterns and tags in such a way that it can understand the correct tag when a statement is entered and return one of the responses. We have created an intents file with more than 100 tags, patterns, and responses with 5 to 10 patterns and responses.

4 Methods

The experiment was conducted using these methods:

4.1 Neural networks

A typical neural network (NN) is made up of multiple small, interconnected processors called neurons, each of which produces a series of real-valued activations. Sensors detecting the environment activate input neurons, whereas weighted connections from previously active neurons also activate other neurons.

The values from the previous layer are multiplied by corresponding weights, and then all of these values are added (basically simple matrix/vector multiplication). Each connection is given a weight. The result of that neuron will be fed into the following layer, which is obtained by passing that value through a non-linear function. This is continued until you get the output layer.

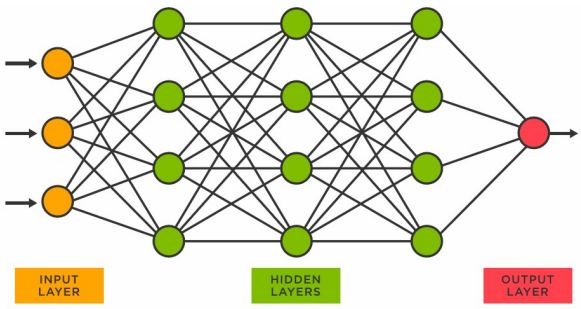

Figure 2: Building Neural network

The weights are initially completely random, but when you train the neural network using a dataset, the weights are altered and the neural network begins to perform the task you have trained it to do. A loss function is used during training to evaluate how accurate or inaccurate the neural network's

output is, and backpropagation is then used to adjust the weights based on the difference between both the actual and predicted outputs for the given input [13].

4.2 Lemmatization

A word's lemma or vocabulary form is created by combining its inflected parts together such that they may be recognized as a single unit. The WordNetLemmatizer class which is offered by NLTK is a thin wrap for the WordNetCorpus. To find a root word or lemma, this class uses a function named Morphy() from the WordNetCorpusReader class [8].

Figure 3: Lemmatization

4.3 Bag of words model

BoW is one of the techniques used for feature selection and categorization. Bag of Words can be used for text classification, image recognition and classification etc...[12].

In this project, we have used text classification which involves classifying or categorizing texts and determining the weights—that is, the number of times each word appears—for each group of words.

4.4 Sequential model

We have created 3 layered Keras sequential model. Python-based Keras is a high-level neural network API that can be used with TensorFlow and other lower-level frameworks. A deep learning library called Keras makes it feasible to build and train models quickly. The sequential model allows the sequential layer-by-layer creation of models. This is a simple and easy-to-use sequential API provided by Keras.[2]

```
# we created a model which is a deep neural network with three layers. F
model = Sequential()
model.add(Dense(128, input_shape=(len(train_x[0]),), activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(len(train_y[0]), activation='softmax'))
```

Figure 4: Example of a sequential model code

5 Experiments

5.1 Intent file creation

We should create an intents file to store the question and answer pair for casual discussion When a user joins in interaction with our chatbot, the input is matched to pattern sets in the intents file, and the appropriate response is returned[9]. The intention of a user's interaction with a chatbot or the objective of each message a certain user sends to the chatbot is what is meant by "intent" [7].

5.2 Model training

If stop words are present in the user's queries, the chatbot system first removes them from the user input. Tokenization and lemmatization are done after the stop words are eliminated from the user's query[6],' Tokenization is the process of breaking a group of words or sentences into their individual components. A variation of stemming, lemmatization is the process of assembling a word's numerous inflected forms so they may be processed as a single entity. Text data should be converted to numbers (0 and 1). Make a bag of words by adding 1 if a word is present in "word patterns" and 0 otherwise [4].

Use Kera's sequential model to define a 3 layered neural network. The first layer of this sequential model is the input layer, which specifies the size of the input that is sent to the neural network. The input to this network will be the array trainX. These would then traverse through the model of 3 different layers with the first having 128 neurons, the second having 64 neurons, and the last layer is output. After then, further layers (hidden layers) could be added and modified until they reach the final output layer. So we are using Neural network to know the patterns between the input features and corresponding output in the dataset.

We have used two activation functions: Rectified Linear Unit(ReLu) and Softmax functions. Relu

is a type of activation function which can be used in both the hidden and output layers. It takes an input and then generates an output between 0 to 1. So you can get outputs such as 0.1, 0.3, and 0.9 as output values in Relu. The softmax activation function is used mostly on the output layer and it is used when you are classifying more than 2 outputs.

Define the optimization algorithm which will be used to train the sequential model; for this, we use the ”SGD” (Stochastic Gradient Descent) optimizer and select the loss function. The neural network is then compiled, which converts the simple sequence of layers into the more complicated set of matrix operations that defines the network’s behavior.

Train or fit the neural network and save it in “uofmchatbot-model” file. Now that the model or network has been trained, predictions may be made using an input.

5.3 Predicting the response for user’s query

Accept input query/message from user with developed frontend using Tkinter library.

Then we have created some methods which will take the user inputs and form an output based on the prediction of the deep learning model created.

Predict Class for Input: Delete any classes or tags which meet the specified threshold from user input, then compile a list of all filtered tags. Reverse-sort the list to get the class or tag with the highest probability. Return the list containing the classes and corresponding probabilities at the end [7].

Bot Response: Save the class or tag that the previous function returned with the highest probability. Compare the tags defined in the ”intents.json” file with that class or tag. Give that class/matching tag’s reply back

5.4 Graphical user interface

Finally, We have created a Graphical user interface using the Tkinter library. Tkinter, also known as ”Tk interface,” is a Python module that allows access to the Tk GUI toolkit. This toolkit was created in TCL (Tool Command Language), and it is multiplatform, supporting Linux, MAC OS, and Microsoft Windows [1].

The three basic concepts of Tk, which also apply to Tkinter, are widgets, geometry management, and event handling. Widgets, often known as controls or window elements, are all components that may be seen on a graphical user interface. Frames, labels, buttons, text fields, checkboxes, tree views, scrollbars, and text areas are a few examples.

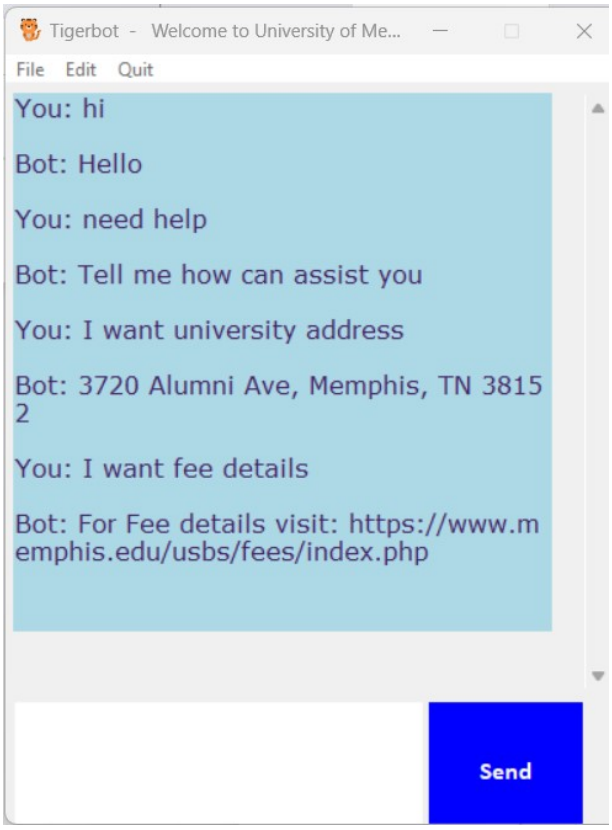

Figure 5: GUI of Chatbot

The positioning of widgets in the onscreen window is a crucial component of interface design. The best way to accomplish that using Tk or Tkinter is through a geometry manager like ”grid.” In actuality, ”grid()” is a technique available to all supporting widgets that specifies exactly where to be placed within an invisible matrix of columns and rows.

6 Results

Our chatbot’s results varies because it is a natural language chatbot that can provide the same answer in a number of different ways. This chatbot is working 87 percent accurately.

```
Epoch 198/200
86/86 [==============================] - 0s 2ms/step - loss: 0.3017 - accuracy: 0.8944
Epoch 199/200
86/86 [==============================] - 0s 2ms/step - loss: 0.3017 - accuracy: 0.8873
Epoch 200/200
86/86 [==============================] - 0s 2ms/step - loss: 0.3470 - accuracy: 0.8779
INFO:tensorflow:Assets written to: uofmchatbot_model\assets
model is created and saved as uofmchatbot model
```

Figure 6: Accuracy of chatbot

7 Conclusion

In this project, we created a chatbot system primarily for University of Memphis that can be customized to the education sector. By integrating this chatbot system into the University website, the webpage will become more interactive with users because it provides extremely accurate answers to their questions. We can add new patterns and make improvements to existing patterns in the intent file that we used to train our chatbot. This will help the chatbot be more accurate and helpful when interacting with users. The administrator must feed the chatbot with more data about University and expand its knowledge base in order to train it to provide meaningful and correct responses. However, getting input from future users might be helpful in creating Chatbot system, which will eventually respond to user inquiries.

8 Contribution for each team member

Maria Vijayanjali Yeruva (U00822234)

• Requirement gathering, data extraction, preprocessing and cleaning the data: process the data in many ways before we can create a machine learning or deep learning model from text data. Preprocess the data using different methods depending on the requirements - tokenization.

• Created training and testing data, I have converted the words to numbers because the machine cannot read words.

• Identifying Feature and Target for the NLP Model: I took words and their corresponding tags out of patterns. This has been accomplished by tokenizing each pattern using nltk.word tokenize while iterating over each pattern with nested for loops.

• Created 3 layers of networks using the Keras sequential API and built a neural network model, Evaluation metrics, developed code for chatbot features, and created final report.

• Created front end – Building GUI (graphical user interface) for chatbot using Tkinter library, Developing integrations between frontend- backend communication.

Sugnan kumar Makkena (U00828231)

• Anaconda Installation and environment setup. Project code review and Analysis.

• Helped in applying the bag of words model to make the data machine friendly: It is one of the most often used models for converting text into numeric forms that may then be processed by machine learning algorithms.

• Helped in creating intents file: A structure for defining conversational intents is necessary for a chatbot framework. Using a JSON file is a simple method to execute this. requirements gathering for the final report.

Harshavardhan Perasani (U00827959)

• Documentation and helped in data gathering, and preprocessing the user input: Lemmatization

• Backpropagation of Error to train Neural Network: The primary goal of the Neural Network method is to identify the correct set of weights for each layer that will lead to the best output, making this stage the most significant one because it focuses on achieving that goal. Created poster presentation ppt and worked on creating latex file for the final report.

9 Source code

This is my Github link: https://github.com/mariayeruva/University-of-Memphis-Chatbot-TigerBot-

References

[1] Douglas Bezerra Beniz and Alexey Espíndola. Using tkinter of python to create graphical user interface (gui) for scripts in lnls. In *USING TKINTER OF PYTHON TO CREATE GRAPHICAL USER INTERFACE (GUI) FOR SCRIPTS IN LNLS*, 10 2016.

[2] Bahzad Taha Chicho and Amira Bibo Sallow. A comprehensive survey of deep learning models based on keras framework. *Journal of Soft Computing and Data Mining*, 2(2):49–62, Oct. 2021.

[3] Manyu Dhyani and Rajiv Kumar. An intelligent chatbot using deep learning with bidirectional rnn and attention model. *Materials Today: Proceedings*, 34:817–824, 2021. 3rd International Conference on Science and Engineering in Materials.

[4] Himanshu Gadge. A chatbot for medical purpose using deep learning. In *A Chatbot for Medical Purpose using Deep Learning*, 2021.

[5] Vagelis Hristidis. Chatbot technologies and challenges. In *2018 First International Conference on Artificial Intelligence for Industries (AI4I)*, pages 126–126, 2018.

[6] Hrushikesh Koundinya K., Ajay Krishna Palakurthi, Vaishnavi Putnala, and Ashok Kumar K. Smart college chatbot using ml and python. *2020 International Conference*, pages 1–5, 2020.

[7] Prathamesh Kandpal, Kapil Jasnani, Ritesh Raut, and Siddharth Bhorge. Contextual chatbot for healthcare purposes (using deep learning). In *2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*, pages 625–634, 2020.

[8] Divya Khyani and Siddhartha B S. An interpretation of lemmatization and stemming in natural language processing. *Shanghai Ligong Daxue Xuebao/Journal of University of Shanghai for Science and Technology*, 22:350–357, 01 2021.

[9] Tarun Lalwani, Shashank Bhalotia, Ashish Pal, Shreya Bisen, and Vasundhara Rathod. Implementation of a chat bot system using ai and nlp. *International Journal of Innovative Research in Computer Science and Technology*, 6:26–30, 05 2018.

[10] Jiwei Li, Will Monroe, Alan Ritter, Michel Galley, Jianfeng Gao, and Dan Jurafsky. Deep reinforcement learning for dialogue generation. *CoRR*, abs/1606.01541, 2016.

[11] Saurav Mishra, Dhirendra Bharti, and Nidhi Mishra. Dr. vdoc: A medical chatbot that acts as a virtual doctor. *Medical Science and Technology*, 6:16–20, 2018.

[12] Wisam A. Qader, Musa M. Ameen, and Bilal I. Ahmed. An overview of bag of words;importance, implementation, applications, and challenges. In *2019 International Engineering Conference (IEC)*, pages 200–204, 2019.

[13] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *CoRR*, abs/1404.7828, 2014.

[14] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144, 2016.