

DEEP NEURAL NETWORK BASED CHATBOT

Garima Srivastava¹, Surbhi Agarwal², Mr. Deepak Vishwakarma³

¹Student, Department of IT, IMS Engineering College, Ghaziabad, India

²Student, Department of IT, IMS Engineering College, Ghaziabad, India

³Assistant Professor, Department of IT, IMS Engineering College, Ghaziabad, India

Abstract - A Chatbot is an automated computer software program that are capable to carry out intelligent live conversations with humans. It is a technology that provides a new way to interact with computer systems using dense neural network. Chatbot responds to user queries in the same language. Chatbots are very popular in business right now as it handles multiple users at the same time and reduces customer costs. But to complete other tasks there is a need to make chatbots as efficient as possible. In this project, the chatbot seeks twitter data and answers to the relative questions using natural language processing and Dense neural network. A Chatbot can give different responses from the same input given by the user according to the current conversation issue. We aim to develop such a Chatbot to solve user queries.

Key Words: Chatbot, Dense neural network, Natural language processing.

1. INTRODUCTION

A chatbot is a software program that are capable to conversate with humans. A chatbot responds to user's queries. It provides a new way to interact with computer systems. For example, a user saying "Thankyou" will result in chatbot saying "You're Welcome". The benefit of using a chatbot is, it responds in the same language as user has input a query, easy to use, respond in a timely fashion and be all round user friendly.

Now-a-days, Chatbots are being used on every platform. In 2016, Chatbots became too popular on messenger. Businesses will be looking to take advantage of Chatbots and started to develop their own chatbots. For example; In business world, a chatbot provides a first level of communication between a customer and company.

Even some companies are developing chatbots and providing it to the other companies to reduce customer costs.

In this project, the structure of the chatbot is kept simple. Figure 1.1 shows the working of a chatbot. When a user asks a query or a statement to the bot, the bot analyse the query and searches reply relevant to the query in its database. When a relevant reply is found the bot compose the reply.



Fig 1.1 How Chatbot works

Python and Dense neural network, which is a deep learning model are used for creating a chatbot. The modules used in python are tensorflow, nltk, numpy, keras, json, tkinter.

Figure 1.2 shows the sample instance of using a chatbot. This shows that, when a user simply enters 'get started' it composes a reply to get started and more and also shows some other options below, may be this can be user's next query.

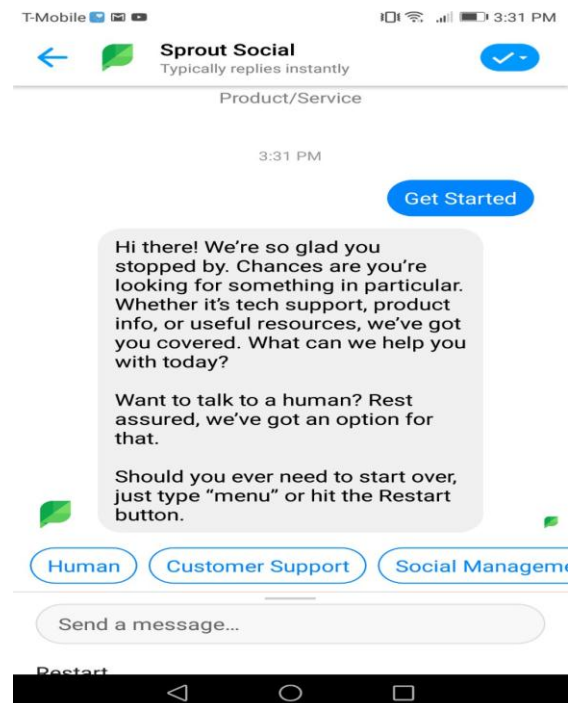


Fig 1.2 Sample Instance of using a Chatbot

2. PROBLEM DEFINITION

Some chatbots gives different replies to the same input given by the user in a current conversation scenario. So, we aim to develop such a Chatbot to solve these user's queries.

3. MODULES DESCRIPTION

The structure of the project is kept simple. The description of the libraries/modules used in this project are given below –

Tensorflow :

It is a framework of machine learning that ease the process of acquiring data, training model, solving predictions and refining future results. Tensorflow bundles together a study of machine learning and deep learning models and algorithm and them useful by way of common metaphor.

Natural Language Processing :

Natural language processing analyze, understands and generates the languages that humans uses naturally and provides interaction with computers in both written and spoken contexts using natural human languages.

Natural language Toolkit (nlTK) :

Natural language toolkit is a collection of libraries and programs for symbolic and statistical NLP for writing English in Python. NLTK involves graphical demonstrations and sample data. NLTK supports researches and teachings in NLP or similar areas, which includes artificial intelligence, machine learning. It has been used as a teaching tool, as an individual study tool, and is a successful platform for prototyping models and building research systems.

NLTK includes classification, tokenization, stemming, tagging, parsing, and semantic reasoning functionalities.

CLASSIFICATION

Classification is a process of choosing the correct class label for an input. In classification process, each input is considered independent from all other inputs, and the sets of labels are defined in advance. For example: Deciding whether an email is spam or not.

TOKENIZATION

In tokenization, the text is splitted into tokens. These tokens can be paragraphs, sentences, or individual words. NLTK provides a large number of tokenizers in

the tokenize module. First, the text is tokenised into sentences and then further processes are performed.

STEMMING

Stemming is organising the words from their root forms such as mapping a group of words to the same stem even if the stem itself is not a valid word and does not exists in the Language.

TAGGING

In the world of NLP, the most basic and known models are based on Bag of Words. But such models failed to figure out the syntactic relations between words. POS tagging is the process of marking a word in a collection to a corresponding part of a speech tag, based upon its context and definition.

PARSING

Parsing suggests the breaking up of a sentence into its component parts so as to understand the meaning of a sentence properly. While parsing a sentence, the reader takes note of the sentence elements and their parts of speech.

JavaScript Object Notation (JSON)

JSON is both an open standard file format and data interchange format that uses human readable text to store and transmit data objects that consists of attribute value pairs and array data - types (or any other related value). It is a common data format, with huge range of applications, such as serving as a replacement for XML in AJAX systems.

JSON is independent of data format. It was derived from JavaScript. The JSON files uses extensions - .json

NumPY

NumPY is a library used in python for supporting large multidimensional arrays and matrices, with a large integration of highlevel mathematical functions for the operation on these arrays. The mathematical algorithms written in this version of Python runs much slower than compiled equivalents. It detects slowness in NumPY, by providing multidimensional arrays, functions and operators which can efficiently work on arrays, and requires rewriting some code, or may be inner loops using NumPy. The core functionality is for n-dimensional array data structure. These arrays are proceeded views on the memory.

Keras

Keras is also an open source library that is written in Python. It is designed to enable fast experimentation with deep neural network models. Basically, it focuses on being user friendly, modular, and extensible. Keras makes the implementation fast and as efficient as possible. It works with deep learning models. In this project, Dense neural network is used, which is a deep learning model.

Dense Neural Network

Dense neural network suggests that layers are fully connected or dense by the neurons in a network layer. Each neuron in a layer receives an input from all the neurons present in the previous layer and hence, they're densely connected with one another. Densely connected layers produce learning features from the combinations of features from previous layers. On the other hand, a convolutional layer depends on congruous features with small repetitive fields.

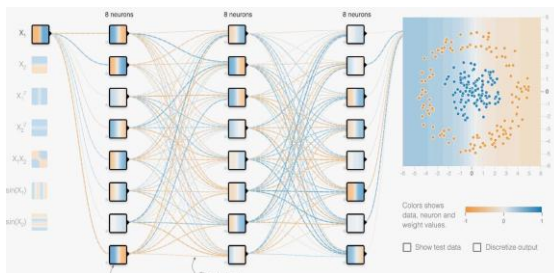


Fig 3.1 Dense Neural Network

4. INPUT DESIGN AND OUTPUT DESIGN

In this project, we have used python GUI with tkinter. It enables to develop fast and efficient GUI applications. Hence for the input and output design of chatbot A box whose dimensions and properties, which have been set up already, is made. Input and Output are in the form of the text.

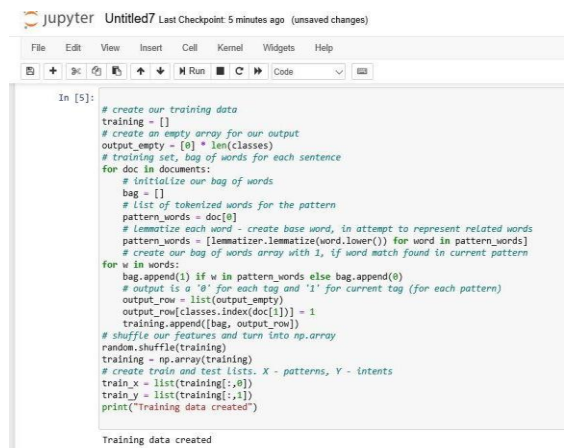
5. IMPLEMENTATION

The implementation of a chatbot starts with some of the steps, which are importing libraries, Lemmatization, creating training data, creating a model and cleaning sentences.

6. RESULTS

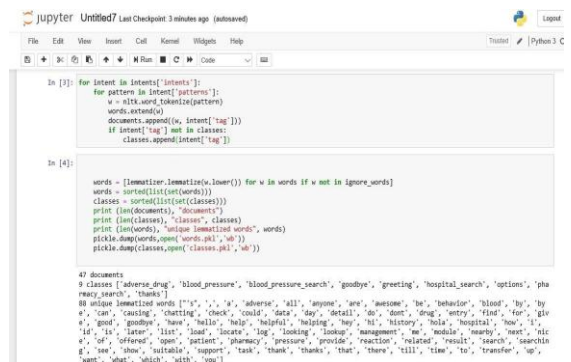
After successful compilation of code in Jupyter Notebook, the model gets trained in 200 epochs with 99% accuracy and GUI code compiles a user interface in python opens having textbox to enter input queries, a button to send

the query and a display area which shows input and output messages.



```
In [5]: # create our training data
training = []
# create an empty array for our output
output_empty = [0] * len(classes)
# training set, bag of words for each sentence
for doc in documents:
    # initialize our bag of words
    bag = []
    # list of tokenized words for the pattern
    pattern_words = doc[0]
    # lemmatize each word - create base word, in attempt to represent related words
    pattern_words = [lemmatizer.lemmatize(word.lower()) for word in pattern_words]
    # create our bag of words array with 1, if word match found in current pattern
    for w in words:
        bag.append(1) if w in pattern_words else bag.append(0)
    # output is a '0' for each tag and '1' for current tag (for each pattern)
    output_row = list(output_empty)
    output_row[classes.index(doc[1])] = 1
    training.append([bag, output_row])
# shuffle our features and turn into np.array
random.shuffle(training)
training = np.array(training)
# create train and test lists. X - patterns, Y - intents
train_x = list(training[:,0])
train_y = list(training[:,1])
print("Training data created")
```

Fig 6.1 Creating training data



```
In [1]: for intent in intents['intents']:
for pattern in intents['patterns']:
w = nltk.word_tokenize(pattern)
words.extend(w)
documents.append([w, intent['tag']])
if intent['tag'] not in classes:
classes.append(intent['tag'])

In [4]: words = [lemmatizer.lemmatize(w.lower()) for w in words if w not in ignore_words]
words = sorted(list(set(words)))
classes = sorted(list(set(classes)))
print(len(documents), "documents")
print(len(words), "unique lemmatized words", words)
pickle.dump(words, open('words.pkl', 'wb'))
pickle.dump(classes, open('classes.pkl', 'wb'))

47 documents
9 classes ['adverse_drug', 'blood_pressure', 'blood_pressure_search', 'goodbye', 'greeting', 'hospital_search', 'options', 'pharmacy_search', 'thanks']
88 unique lemmatized words ['a', 'adverse', 'all', 'anyone', 'are', 'awesome', 'be', 'behavior', 'blood', 'by', 'case', 'causing', 'chatting', 'check', 'could', 'data', 'day', 'detail', 'do', 'don't', 'drug', 'entry', 'find', 'for', 'give', 'good', 'goodbye', 'have', 'hello', 'help', 'helpful', 'helping', 'how', 'hi', 'history', 'hold', 'hospital', 'how', 'i', 'id', 'is', 'later', 'list', 'lose', 'locate', 'log', 'looking', 'lookup', 'management', 'me', 'mobile', 'nearby', 'next', 'nic', 'of', 'opened', 'open', 'patient', 'pharmacy', 'pressure', 'provide', 'reaction', 'related', 'result', 'search', 'sensitive', 'see', 'show', 'suitable', 'support', 'task', 'thank', 'thanks', 'that', 'there', 'till', 'time', 'to', 'transfer', 'up', 'want', 'what', 'which', 'with', 'you']
```

Fig 6.2 Documents, Classes and Lemmatized words

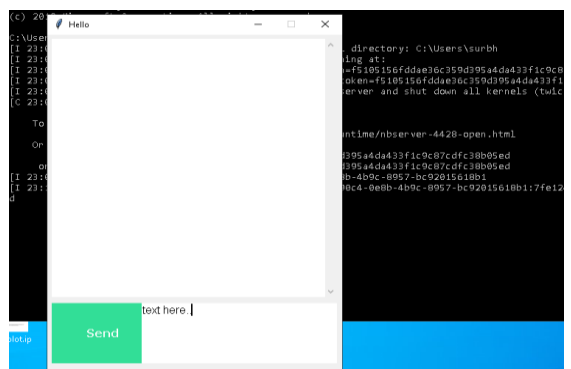


Fig 6.3 Actual Instance of our Chatbot

7. CONCLUSION

We conclude that a chatbot developed using deep neural network with tensorflow works better than simple machine learning based chatbots. Also, the overall code gets significantly reduced using libraries. GUI made using tkinter in python is sufficient to develop such an interface without the need of frontend technologies like HTML, CSS and JavaScript.

As future work, with the advances in AI development will impact Chatbots with AI making iterations fast, Voice

experiences going mainstream, will use Blockchain which is the surprising ally for chatbot. Our work on chatbot does not get over with this. We shall continue to make it better and take it to the advancement.

REFERENCES

[1] Akshay Kumar, Pankaj Kumar Meena, Debiprasanna Panda, Ms. Sangeetha, "Chatbot In Python", SRMIST, Chennai[1]

[2] <https://en.wikipedia.org/wiki/chatbot>

[3] Bayan Abu Shawar and Eric Atwell, 2007 "Chatbots: Are they Really Useful?"