# *Chatbot*

## *A Deep Neural Network Based Human to Machine Conversation Model*

G Krishna Vamsi
Computer science and engineering
Maulana Azad National Institute Of Technology
Bhopal, India

Akhtar Rasool
Computer science and engineering
Maulana Azad National Institute Of Technology
Bhopal, India

Gaurav Hajela
Computer science and engineering
Maulana Azad National Institute Of Technology
Bhopal, India

***Abstract*— A conversational agent (chatbot) is computer software capable of communicating with humans using natural language processing. The crucial part of building any chatbot is the development of conversation. Despite many developments in Natural Language Processing (NLP) and Artificial Intelligence (AI), creating a good chatbot model remains a significant challenge in this field even today. A conversational bot can be used for countless errands. In general, they need to understand the user's intent and deliver appropriate replies. This is a software program of a conversational interface that allows a user to converse in the same manner one would address a human. Hence, these are used in almost every customer communication platform, like social networks. At present, there are two basic models used in developing a chatbot. Generative based models and Retrieval based models. The recent advancements in deep learning and artificial intelligence, such as the end-to-end trainable neural networks have rapidly replaced earlier methods based on hand-written instructions and patterns or statistical methods. This paper proposes a new method of creating a chatbot using a deep neural learning method. In this method, a neural network with multiple layers is built to learn and process the data.**

***Keywords*— *machine learning, conversational agent, chatbot, machine learning classification technique, Neural Networks, Deep Learning, Natural Language Processing.***

## I. INTRODUCTION

A chatbot is a software program of a conversational interface that allows a user to converse in the same manner one would address a human. A virtual chatbot is a piece of software that is intelligent enough to mimic human interactions. Conversational bots are used in almost every customer interaction, like instantly messaging the client. Since the development of the first chatbot, they have evolved in functionality, interface, and their significance to the technical world cannot be neglected. However, modelling conversations remains a significant challenge in this field even today. Even though they are a long way from perfect, conversational agents are now used in various applications [1]. To understand the capabilities and limitations of current chatbot techniques and architectures, a detailed survey was conducted, where interrelated literature published over the past few years is studied, and a newly presented neural network model is now trained with conversational data. Deep learning and NLP techniques are being used in advanced research and development projects, AI and ML algorithms are implemented in development of conversations. R&D (Research and Development) are still under progress and experimentation in these fields. Conversation agents are predominately used by government administrations, businesses, and non-profit establishments. They are often organized by financial institutions like banking, insurance, startup companies, online stores and social service sectors. These chatbots are implemented by large corporations as well as small startup companies. However, chatbots are not under proper implementation in the medical field. A chatbot can help patients with medical-related works by assisting them via text messages, applications, or instant messaging. One can find many virtual bot development structures in the market, both interface-based and code-based. However, both models have limitations concerning flexibility and practicality in making real conversations. Most of the well-known intelligent personal assistants like Alexa by Amazon, Google Assistant by Google, and Cortana by Microsoft do have some limitations in functionality. Retrieval based agents are being introduced to hold conversions that match real interactions alike humans. Among many intelligent personal assistants currently available, they are structured on rule-based techniques or retrieval-based techniques that generate decent results [2].

Recently there is a significant increase in curiosity in the usage of chatbots. Many companies are successfully using these dialogue generation devices to fulfill customer requirements. Since companies are adopting the chatbot technology, there is a promising demand for the development and advanced research for the conversational agents.

## II. RELATED WORK

Machine learning and deep learning is useful for various set of problems. One of the applications of this technique is in predicting a dependent variable from the values of independent variables. Machine Learning is a part of Artificial Intelligence. ML finds a solution to the problems by recognizing patterns in

the databases rather than depending on rules. The Machine Learning techniques such as linear regression and naive Bayes methods study the correlation between features and the value of the output class. In other words, ML techniques help machines to understand some information about the real world. "The calculations hang on the correlation between features and output value, however, some modular designed features are restricted in these models, this may lead to many difficulties, like representing each entity as a different set of features, to verify that, consider the problem of face detection as an example, the modeller cannot represent a face on a pixel-to-pixel basis, but can easily represent with a set of features such as having a particular shape and structure" [3].

In this approach, the main drawback is the extreme importance of representation of data. A crucial task in designing a chatbot is to make the discussion between the system and the user feel human-like and natural. To increase human perception, several models with CUI (conversational user interfaces) like virtual bots are programmed to deliver delayed replies/responses to mimic the time taken by the humans to reply. However, a delayed response may negatively affect user fulfillment, mostly in times where quick responses are expected, for example, in customer interactions.

A novel method to implement word segmentation was introduced by Mohammed Javed et al. [4]. The algorithm counts/calculates the character spaces, including all types of gaps, including space between letters, punctuations, and words. The algorithm works on the number of gaps in the sentence. The average gap is calculated to determine the mean average gap between characters of the sentence. If the calculated space is above the calculated average space, these spaces are identified as tokenization points, this leads to tokenization at the blank spaces in-between words.

Word segmentation implemented by using Natural Language ToolKit (NLTK) was proposed by Naeun Lee et al. [5]. A Python package NLTK with inbuilt tokenizers provides services for Natural Language Processing. "A wide range of tokenizers are included which are as follows standard, letter, word, classic, lowercase, N-gram, pattern, keyword, path, etc. The word-punkt tokenizer is basic and simple, the sentences at the blank spaces are split. The accuracy, speed, and efficiency of the NLTK tokenizers is commendable" [5]. The package executes the algorithm at the backend.

Though chatbots are primarily question answering systems, if chatbots are used in medical-related tasks like notifying patients about an appointment or a regular check, to gather advice and check up on their health complaints, healthcare workers can spend time caring about patients.

For some minor health issues, the chatbot can be used for health advice and further guidance rapidly. These chatbots are particularly useful when one struggles to understand telephonic advice. Therefore, these conversational agents are seen as very useful, time-saving, and also cost-effective in healthcare services.

## III. Proposed Work

**Problem statement**: A chatbot is computer software that communicates with humans using natural language. In the past few years, several corporations are using chatbots to fulfil customer's needs. While chatbots are being used in many tech-related works, they are not under proper implementation in medical-related tasks. Chatbots entering the healthcare industry can help ease many of its difficulties.

The motivation for this research is to present a better way in modelling a chatbot that can be used for various medical-related tasks like to notify patients about a future appointment or a regular check, to gather advice, to check up on their health complaints, or to promote a healthcare business. In the meantime, healthcare workers can spend time caring about patients instead of going through a needless routine. The main challenge in designing a conversational agent is to make the conversation between the person and the software feel natural and human-like. A simple Graphic User Interface (GUI) to communicate with the chatbot.

**Contribution:** Chatbot typically uses sentences expressed by the user in natural language as input and delivers output response. Among the two main methods of generating responses, the old-style method uses hardcoded templates and instructions, whereas the advanced model that emerged with the growth of Deep Learning uses enormous data. Deep neural network models are trained on large amounts of data/information to learn the procedure of responding with relevant and grammatically accurate responses to input utterances. Some advanced models are developed to accommodate visual or spoken inputs.

This chatbot performs services like offering support for Adverse drug reaction, Blood pressure tracking through patient ID, Hospitals, and Pharmacies search. Indeed, creating a decent chatbot remains a tough challenge despite the advancements in Artificial Intelligence.

## IV. Dataset

The dataset is taken from the open-sourced Kaggle healthcare services competition. It is a simple dataset with 14 different tags, and each tag has patterns and responses. These patterns and responses take Question and Answer form. The dataset has a dictionary mapping of dictionaries. First, intents are mapped with tags, patterns, context, and responses, and then each of them is mapped with their queries and keywords [6].

This is a small dataset, and training it over deep neural networks model leads to overfitting the model, to find the optimal learning rate appropriate precautions must be taken while building the model.

TABLE I. Dataset description

| S. No | Attribute Name | Description |
|---|---|---|
| 1. | Tags | Tags are keywords assigned to the patterns and responses for training the chatbot .e.g., "greeting", "bye". |
| 2. | Patterns | These are the types of queries asked by the user to the chatbot. |
| 3. | Responses | Responses are the answers generated by the chatbot for the respective queries. |
| 4. | Context | Context is given for which the queries require a search and find operation. |

## V. FEATURE ENGINEERING

This dataset is a text data with 14 tags and multiple patterns and responses for each tag. When operating on text data, several preprocessing of the text data should be done before creating an ML or a deep learning model. Tokenizing is the simplest and a fundamental thing that one can work on text data. It is a simple process of making text data into small parts called words. Natural Language Processing is used for tokenizing the patterns. NLTK module provides natural language processing in Python. Then each word is lemmatized, and duplicate words are removed from the list of words. The converting of words to their lemma form and then creating a pickle file that stores the python objects used while predicting is known as lemmatizing.

Now, the training data is produced in which the input and the output data is provided. The input will be the pattern, and output will be the class the input pattern belongs to. The input is first divided into small parts called tokens then they are all lemmatized so that strings can be compared from the chatbot database, this database contains all the responses, and an appropriate message is given as a response to the user. However, the machine does not understand the text, so the text is converted into numbers. This is similar to one-hot-encoding for categorical data. The output of this process is given as input to the neural network. The whole process is represented in fig. 1. Data is split into two categories, training, and testing dataset. The training and testing dataset is split into 80% and 20%, respectively. Training data is used to train the model and testing data to measure the accuracy of the model.

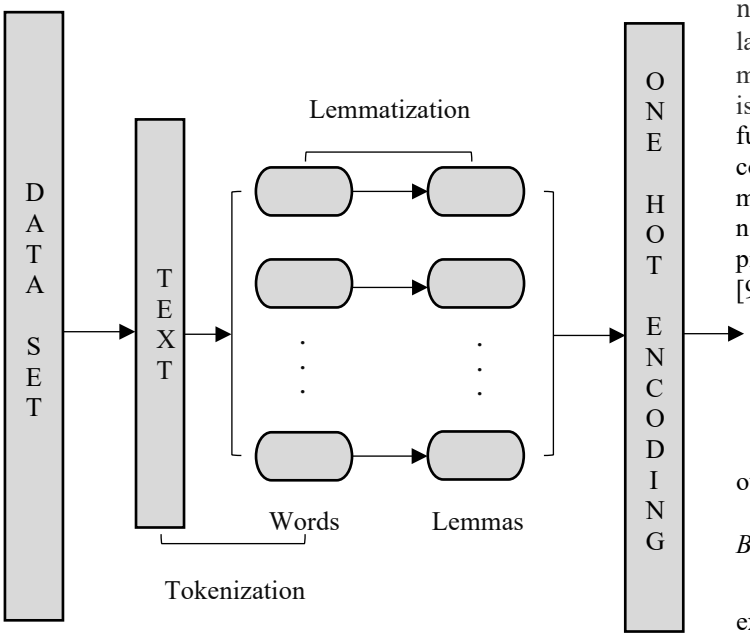


Fig. 1. *Flowchart of working of the chatbot*

### A. Architecture

In deep learning, a neural network with multiple layers is built for processing the input data or training data so that the model can extract higher-level features of the data. The conversational bot is trained on the dataset, which holds categories/classes intents, patterns, responses, and context. To classify in which category the user's message fits into, a special DNN is used, and then a random response from the list of responses is given. Natural Language Processing and Keras are used for creating Retrieval based chatbot [7].

The core aspects of Deep Learning are successive layers of representations, which helps in learning the features from low-level to high-level in a hierarchical manner, Deep Learning which is also known as a hierarchical representation of learning invalidated the feature engineering's higher-level dependence of shallow networks [8].

A feed-forward neural network type architecture is used in this model building in which the information moves only forward direction in the network. In each hidden layer, the input is transformed, and output is transferred to the following layers. In the following equations, the DNN model is interpreted mathematically.

$$z^{(n)}=a^{(n-1)}w^{(n)}+b^{(n)}$$
$$a^{(n)}=g(Z^{(n)}) \quad (1)$$

In (1): "n ∈ [1,…, n] denotes the $n^{th}$ layer of the neural network with $Z^n$ is the vector of activations of $n^{th}$ layer, $a^{(n-1)}$ is the output of the preceding $(n-1)^{th}$ layer and input to $n^{th}$ layer. $W^n \in \mathbb{R}^{l_i \times l_o}$ is a matrix of learnable biases of $n^{th}$ layer, $a^{(n)} \in \mathbb{R}^{l_o}$ is the output of $n^{th}$ layer, $a^{(0)}$ is the input to the model and $a^{(n)}$ is the output of the last layer of the network, $g(.)$ is the non-linear activation function. Non-linear activation function ReLU is used in the hidden layers because of its computational efficiency and faster learning convergence over most of the other activation functions. The output layer of the network uses a Softmax Non-linearity activation function to provide a probabilistic interpretation of the network's output" [9].

$$\text{Softmax}(z^{(L)})=\frac{e^{z_L}}{\Sigma_{k=1}^{k} e^{z_k}} \quad (2)$$

In (2), no. of output classes is denoted by k, that is, the output layer contains k number of neurons.

### B. Learning

The learning of a deep neural network is to improve the existing performance of the neural network. In deep learning, cost-function is used for the model network with the best parameters. The model used is a classification type, so cross-entropy function is used in the network.

$$L=-\hat{y}\log(y)-(1-\hat{y})\log(1-y) \quad (3)$$

$\hat{y} \in \{0, 1\}$ in (3), and it is a one-hot-encoded label, and the model's output is denoted by y. With the aim of training the model, "The gradients are computed by differentiating the cost-function with respect to the model parameters using a mini-batch of data sampled from the training data and backpropagated to prior layers using the backpropagation algorithm" [10].

### C. Optimizers

Optimizers are the algorithms used to change the weights and learning rate of hidden layer neurons to minimize the losses of the network. The losses of the network are represented by Error function E(x), a mathematical function which is also known as Objective function invariably depends on the model's internal learnable parameters. The same function is used for computing the target values(Y) from the set of predictors(X) used in the model. "For example— the Weights(W) and the Bias(b) values of the neural network as its internal learnable parameters which are used in computing the output values and are learned and updated in the direction of an optimal solution, i.e., minimizing the Loss by the network's training process and also play a major role in the training process of the Neural Network Model" [11].

The internal parameters which are crucial in any model play a vital role in training a model efficiently and effectively in order to produce results accurately. "This is why various Optimization strategies and algorithms are used to update and calculate appropriate and optimum values of such model's parameters which influence our model's learning process and the output of a Model" [11].

### D. Activation Function

In DNN, the activation function of artificial neurons takes a set of input values or inputs and produces the output according to the activation function used. Some of the efficient functions are Gaussian, Multiquadric, and Inverse activation function.

The most important properties of activation function include

1. An activation function is continuously differentiable.
2. An activation function is desirable when it is bounded.
3. An activation function is a smooth function with monotonic derivatives.
4. An activation function is a monotonic function.

According to the Cybenko theorem [12], it states that it is possible to achieve approximately any continuous function with a feed-forward neural network with a single hidden layer of finite neurons and a sigmoidal activation function.

### E. Weight Initialization

Weight initialization is a fundamental initialization framework for developing a neural network, and an appropriate initialization parameter is chosen so that optimization can be reached in the optimal time. Developing an efficient initialization of a model is a vital step. Often, training is performed by initializing the parameters, specifying an optimization algorithm, computing the cost function, gradients of a cost function using the backpropagation, and so on updating the initialized parameters. Finally, it ends at updating the initialized parameters. Therefore, initialization is the foremost step in better performance of the model, Sartaj Singh Sodhi et al. [13] came up with an idea of assigning equal initial values to the weight vector. Still, it was discovered that the weights change in groups during the weight update and maintain symmetry, this causes the problem in training the network systematically. To address this problem the random weight initialization technique was introduced. "The initial weight values are chosen uniformly from a range of $[-\delta, \delta]$. Xavier and He normal weight initialization methods have been extensively applied since they submitted it" [13].

## VI. Experimental Results

In the deep neural network, the most crucial task is allocating weights to the hidden layers. In accordance with fig. 2, it can be seen that different weight parameters are earmarked for each neuron with the different activation function value at each level. Activation functions and weight values are adjusted during the training of the model to provide the most optimal value at the output. A keen analysis of the output is done for evaluating the proposed work on each parameter, especially related to the accuracy and cross-entropy error.

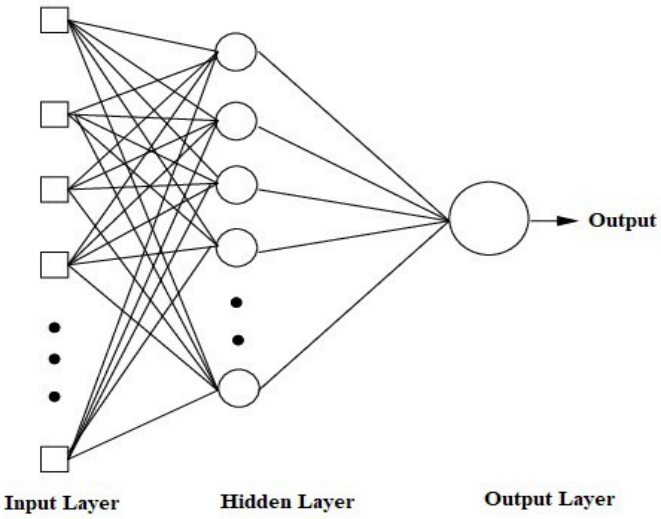


Fig. 2. *Deep Neural Network*

The analysis is divided into three parts, in the first part, a general network which consists of one hidden layer and most widely used activation function and optimizer algorithm are used. Then in the next section, analysis of different optimizer algorithms is performed, their performance is scaled on the basis of performance metrics.

Then in the next section, different kinds of weight initializers are used, and their performance is measured, this is the third part of the analysis. For proper communication with the chatbot, a user-friendly graphic user interface (GUI) is

created. A software GUI library called 'Tkinter' for Python is used to create a simple user interface.

Tkinter is one of the fastest ways to create a simple GUI application. It is cross-platform, so the same code works for different operating systems. Tkinter is Python's de-facto standard GUI package [14], so it is easy to use when compared to other GUI creating applications. A sample conversation with the chatbot is shown in fig. 3.

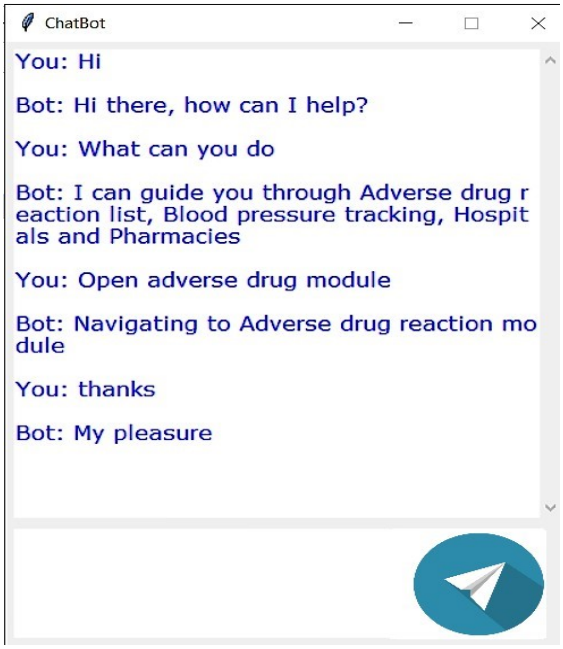


Fig. 3. *GUI of the Chatbot*

### *A. Accuracy of different Optimizers*

The train and test data are split into 0.2 proportion. In this analysis, there are different models created for each optimizer algorithm, and analysis is performed on each one of them. Here each model has two hidden layers and Relu as an activation function in hidden layers. Five different optimizers were chosen for this purpose. They are Stochastic Gradient Descent (SGD), AdaGrad, AdaDelta, RmsProp, and Adam. The graph (fig.5) is training loss and validation loss vs. the number of epochs. From this graph, one can analyze the performance of different optimizers using the minima generated in their learning span.

SGD optimizer outperforms other optimizer algorithms in this specific neural network model. The first minima generated by the model is with zero validation and training loss, which implies that the network adapts fast using SGD. It works the same as regular gradient descent, but updates for every epoch. SGD is prone to noise in the data, but here it performs better, which implies the outliers in the data is much less than expected. The dataset is small. Hence, something from the SGD optimizer is expected, and it gives the best performance in terms of accuracy. Somehow, it finds the minima, which is higher than other optimizers, but SGD gives the best results while solving the problem. SGD converges at the second epoch. RMSprop and Adadelta have the same minimas and give close accuracies. Adadelta automatically makes the learning rate larger or smaller in an inversely proportional way to the gradient. In this way, the network does not learn too slowly and does not skip the minima by taking a large step. RMSprop accumulates gradients from most recent iterations and uses exponential decay. It just falls behind the SGD. Both RmsProp and AdaDelta converge at the eighth epoch. The number of epochs vs. training loss for various optimizers and validation loss graph is shown below.

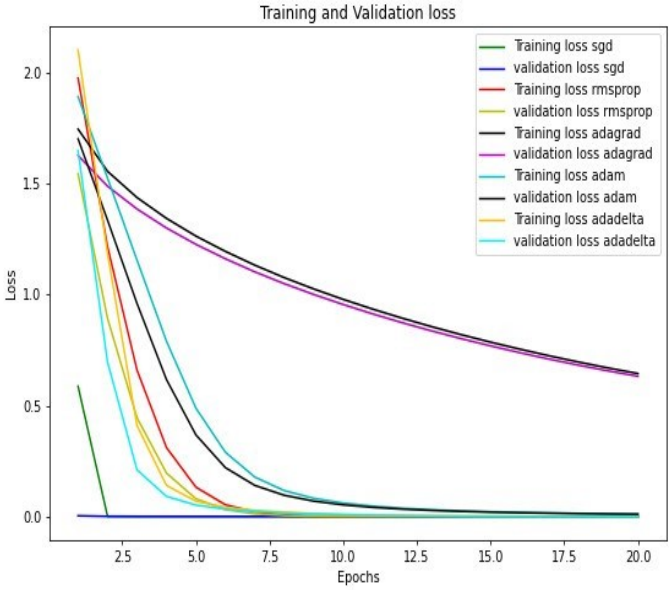


Fig. 4. *Number of epochs vs. training loss for various optimizers and validation loss*

In general, Adam performs better than most optimizer algorithms, but here it is not the best optimizer. It may be due to a small dataset. It converges at the fifteenth epoch. With the use of AdaGrad, the learning rate decays slower, which implies the dimensions have a gentler slope for learning. AdaGrad performs worst in this case. It never converges below 20 epochs and approaches global minima, but at a slower rate. The overall performance of optimizer algorithms in increasing order is AdaGrad, Adam, RMSprop, AdaDelta, and SGD. The accuracy obtained from different models is given in the table below.

TABLE II. Accuracy of different optimizer algorithms

| | **SGD** | **AdaGrad** | **AdaDelta** | **RmsProp** | **Adam** |
|---|---|---|---|---|---|
| **Accuracy** | 100% | 89.57% | 99.89% | 99.64% | 95.55% |

### *B. Accuracy of different Weight Initializers*

Weight initialization is used to control the output of the layers either from increasing or decreasing dramatically. The same neural network with SGD optimizer is used, and the weights in each neural network are initialized by using different initialization techniques. Seven different kinds of initialization techniques initialize the weights in this neural network, and they are Glorot Normal, He Normal, Zeros, Ones, Random Normal, Identity, and Orthogonal and this observation is given in table III.

TABLE III. Accuracy of different weight initializers

| Weight Initializers | Accuracy |
|---|---|
| Glort_normal | 100% |
| He_normal | 100% |
| Zeros | 88.12% |
| Random_normal | 98% |
| Identity | 100% |
| Orthogonal | 98% |
| Ones | 100% |

Initialization with Ones and Identity begins with less training loss and converges at the global minima by the second epoch. This implies that the network does not have exploding/ vanishing gradient problems with simple symmetrical weight initialization. Random normal and He normal look similar in the graph, they start with the same validation loss and converge similar to Ones and Identity weight initializers. Both converge at the second epoch. The large values of validation loss at the start of the training are due to random initialization of weights, resulting in inconsistent variance.

Random normal, He normal, Ones and Identity initialization of weights results in 100% accuracy of the model. Orthogonal initialization of weights is similar to Random normal, but with the Orthogonal property of the vectors, this initialization begins with a higher validation loss. Nonetheless, orthogonal initialization also converges at the second epoch. Zero initialization is worst at the beginning because all weights are initialized to zero. It converges slowly but has many minimas, initializing with zeros may harm the learning gradient. The global minima for Zero initialization is achieved around the fifteenth epoch. Form this. It is evident that Random normal, He normal, Identity, and ones initialization of weights works best for this network. In general, Random normal initialization gives better results when compared to other initializers in different networks. A graph is plotted between the training loss vs. the number of epochs (fig. 5).

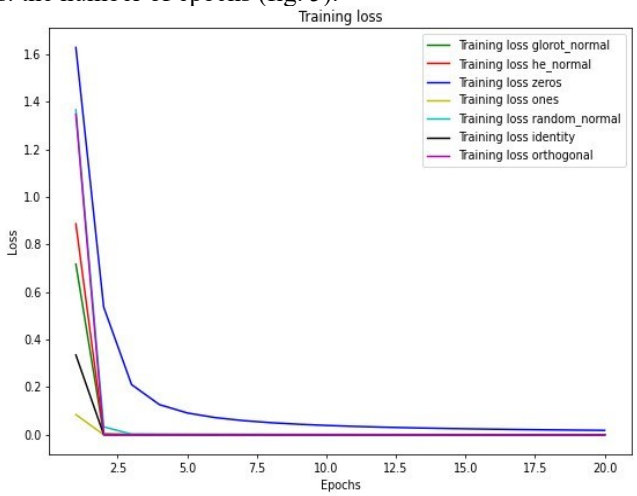


Fig. 5. *Number of epochs vs. training loss for various weight initializers*

## VII. DISCUSSION

A DNN is used to evaluate the analysis of the dataset. A machine learning toolbox 'Keras' based on Python is used to evaluate the performance based on the analysis. The main aim is to study neural networks and the different algorithms used in the neural network. The research shows that SGD is the best optimizer here, producing 100% accuracy, and the He normal weight initialization gives fast results in terms of the number of epochs. The activation function used for input and the hidden layer is ReLu. The activation function used for the output is Softmax. A sequential type of model building is used in this model. A graphic user interface (GUI) is created to communicate with the chatbot using Tkinter (Python).

## VIII. CONCLUSION

Conversations/interactions between services and people are streamlined by the chatbot applications, whereby developing the costumer's experience in a better way. These chatbot applications also give companies innovative opportunities to maximize the customer's engagement process and their productivity by reducing the inevitable costs of customer service.

The research discovered that a chatbots' performance could be improved by using neural networks and different algorithms. It is important to acknowledge the limitations like the accuracy of the model, the lack of empathy, and privacy concerning users' data, and the chatbot could be improved by providing the availability of a voice version, which helps visually impaired or illiterate people. While many tasks are done by the chatbots, they can never replace humans until chatbots understand human emotions and human perception. This is even truer in the healthcare sector.

In future research, investigation of other enhanced methods can be done, that would further raise the standards of chatbots. Furthermore, the current advanced developments in neural networks, including RNN, deep CNN, and LSTM, a deep brief network based on restricted Boltzmann machines, and deep auto-encoder should be used for further development of conversational bot.

## REFERENCES

[1] B. Setiaji and F. W. Wibowo, "*Chatbot Using a Knowledge in Database: Conversation Modeling*" pp.71–78,*2016.*

[2] M. T. Mutiwokuziva, M. W. Chanda, P. Kadebu, A. Mukwazvure and T. T. Gotora, "*A neural-network based chat bot*" 2017 2nd International Conference on Communication and Electronics Systems (ICCES), Coimbatore, pp.211– 217,2017

[3] Kyunghyun, "*Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation*", 2014

[4] Mohammed Javed, P. Nagabhushan, B.B. Chaudhari, "*A Direct Approach for Word and*

*Character Segmentation in Run-Length Compressed Documents with an Application to Word Spotting*", 13th International Conference on Document Analysis and Recognition (ICDAR), 2015.

[5] Lee, Naeun & Kim, Kirak & Yoon, Taeseon, *"Implementation of robot journalism by programming custombot using tokenization and custom tagging",*(ICACT.2017.), pp.566-570, 2017.

[6] Chatbot-project,google.drive,12.dec: https://drive.google.com/file/d/1763Y5zy7HmRYsOoBL0gUxQRGY6xCgQiN/view. [Accessed: Dec, 2019].

[7] Deepti Chopra, Jacob Perkins, and Nitin Hardeniya, "*Natural Language Processing: Python and NLTK*", pp.21–33,2016.

[8] Y. LeCun, Y. Bengio, G. Hinton," *Deep learning*", Nature 521 (7553), pp.436–444, 2015.

[9] Jeff Heaton, *"Introduction to the math of neural networks"*, pp.29–64,2012.

[10] Michael Nielsen, *"Neural networks and Deep learning: Introduction to core principles"*, 2(3):pp.39–53, 2019.

[11] Xiang-Sun Zhang, "*Neural networks in optimization*", pp.199–236,2013.

[12] G. Cybenko, "*Approximation by superpositions of a sigmoidal function. Mathematics of Control, Signals and Systems*", 2(4):pp.303–314, 1989.

[13] Sartaj Singh Sodhi and Pravin Chandra, "*Interval based weight initialization method for sigmoidal feed-forward artificial neural networks*". AASRI Procedia, pp.6:19–25, 2014.

[14] Burkhard A. Meier, "*Python GUI Programming Cookbook*", pp.1–67,2015.