ISSN: 2229-6956 (ONLINE)
DOI: 10.21917/ijsc.2022.0392

DESIGNING EFFECTIVE CHATBOT SYSTEM USING GRU WITH BEAM SEARCH

Sandeep A. Thorat[1], Vishakha D. Jadhav[2], and Dadaso T. Mane[3]
[1,3]Department of Computer Science and Information Technology, Rajarambapu Institute of Technology, India
[2]Department of Computer Science, D.A.V. Rajarambapu Institute of Technology, India

Abstract

Artificial Intelligence (AI) based Chatbot is a moderately new technology in the world. AI and Natural Language Processing (NLP) empowers a Chatbot to converse like a human being. Chatbots have become popular recently as they diminish human efforts by automating various tasks. AI-based Chatbot learns from the previous discussion and generates an appropriate response or action for the input given by the user. In the proposed research work we designed AI-based Chatbot system using the Sequence to Sequence (Seq2Seq) model. This system uses a Gated Recurrent Unit (GRU) for encoder and decoder. In the proposed model the GRU encoder accepts a query from the user. The GRU encoder uses an attention mechanism to consider only relevant information and convert it into the context vector form. A context vector is another input to the GRU decoder. The GRU decoder generates a response using the Beam search algorithm. The research work uses Cornell Movie Dialogue Corpus to train the proposed interactive Chatbot system. It is observed that the proposed model with the combination of GRU and Beam search gives better accuracy with the minimum loss for testing data. These experimental results are better than existing approaches that use LSTM Seq2Seq models to train Chatbot systems.

Keywords:

Chatbot, Deep learning, RNN, GRU, Attention Mechanism, Beam Search

1. INTRODUCTION

A Chatbot is a computer program that helps for communication between humans and machines. The key objective of a Chatbot is to make a user think that communication is going with another person. Chatbot is also known as a Talkbot, chatterbot, or interactive agent. Chatbot systems are being used in various sectors like government, businesses, financial organizations, etc. Two popular techniques are used to develop Chatbots; the first one is rule-based or template-based, and the second one uses machine learning and deep learning algorithms. In a rule-based approach, user questions and corresponding answers are predefined. Here the response given by Chatbot depends on the matching of rule or pattern between the user query and stored responses. Rule-based chatbots cannot work properly because of the lack of flexibility and inability to do summarization [21]. Machine learning and Deep learning-based techniques need a manually collected dataset for training purposes. Unlike rule-based AI-based Chatbots are able to learn on their own [20]. This research work focuses on the design of an effective AI-based Chatbot.

The biological neurons enliven Neural Network (NN) in the human brain. Components of NN involve neurons, connections, weights, biases, activation function, and learning rules. Deep learning is a specialization of machine learning algorithms which is based on NN. AI-based Chatbots use Deep Neural Network (DNN) algorithms. As compared to machine learning, deep learning works on a large dataset. Deep learning is designed with the help of connected layers. The input layer is first layer, and the last layer is the output layer. Hidden layers are the middle layers. There are more than two layers in the hidden layer in deep learning. Various types of deep learning models include ANN, Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), etc. [2].

The proposed research work uses Seq2Seq architecture for designing a Chatbot system. Earlier research focused on choosing appropriate hyper parameters for performance improvement in the Chatbot system using the LSTM (Long Short-Term Memory) Seq2Seq model [22]. Bi-directional LSTM contains few hidden layers; all information from sentences is passed through those layers. The proposed research work aims to improve the accuracy of Chatbots and produce high-quality responses. The proposed model improves the performance of the Chatbot system by replacing the LSTM model by using GRU with Beam search decoding. The LSTM encoder-decoder structure is complex compared to other models [14].

The working of the GRU encoder-decoder is the same as LSTM, but it uses a simplified structure. Unlike LSTM, GRU has only three gates: update, reset, and current memory gate. These gates are used to decide information which should be delivered to the output [17], [6]. Both LSTM and GRU models are useful to overcome vanishing and exploding gradient problems. The proposed model also uses the attention mechanism to focus on important words in the sentences. The input sequence is converted into the vector representation in GRU and then passed to the encoder. The encoder produces an internal representation which is then inputted to the decoder. Here output generated by the decoder is the target response [3]. The proposed model is trained using the Cornell Movie dataset. The experimental results show that GRU with Beam search decoding gives better results than existing models which depend on LSTM.

This paper is arranged as follows: Section 2 presents a literature survey about the development of different Chatbot systems. Section 3 discusses the design of Chatbot using GRU with Beam Search in detail. Section 4 presents the experimental results of LSTM and GRU models. The paper ends with the conclusion in Section 5.

2. LITERATURE SURVEY

R. B. Mathew et al. [10] used NLP and machine learning to create Chatbot for disease prediction. This Chatbot system identifies some symptoms and recommends the treatment. This system is trained on the symptoms-disease dataset. K-nearest Neighbor (KNN) algorithm can predict the disease depending on the user symptoms. Most of the companies deploy Chatbot systems on their websites and cloud services. In 2018 P. Rivas et al. [18] presented an experimental survey about perception people about bots and their trustworthiness for performing tasks. In this survey, various bots are included which is based on machine learning. The analysis was online and here the authors designed a plot based on customer experience.

B. Kohli et al. [8] described some agents who depend upon natural expressions. The authors analyzed artificial clients like Eliza, Zen, Iesha, and Rude. Use of the AI increases day by day thus in this paper web-based protocols are used to make a Chatbot system. A. Argal et al. [1] proposed a smart Chatbot system for the tourism domain on the Echo platform. M. Nuruzzaman et al. [13] presented a survey on existing DNN based Chatbots; authors have discussed their characteristics and limitations. The authors compared the 11 most popular Chatbot systems with their functionality. In 2018 A. Nursetyo et al. [12] proposed an intelligent Chatbot system based on AIML. AIML performs pattern matching on user query and give a response. This Chatbot system is useful as E-commerce assistant.

P. Kumar et al. [9] used the Seq2Seq model for designing Chatbot system. T. Nguyen et al. [11] build a Vietnamese Chatbot based on the Seq2Seq model with an attention mechanism. For testing purposes authors use the deep learning framework PyTorch with GPU. This Chatbot system generates simple and basic conversation however due to limited training size and inappropriate answers it has limitations. S. Prasompha et al. [16] proposed a Chatbot system that is useful in many businesses like call centers, train schedules, and reservations. The authors used a deep learning-based AI technique to understand the conversation between machines and users. The authors attempted to improve the performance of this Chatbot system using the Seq2Seq model.

Sometimes the Seq2Seq model generates generic answers which are inconsistent with the questions. K. Palasundram et al. [15] proposed some methods that identified the weakness of the Seq2Seq model. The authors observed that few factors are affecting the performance of the Seq2Seq model. These factors include insufficient input data, misuse of cross-entropy function, infrequent words used in data, and selecting wrong data processing methods. G. Dzakwan et al. [5] attempted to increase Chatbot performance using some advanced NN encoder-decoder model. The authors used combinations of RNN, GRU, and LSTM with their modifications for designing the Chatbot system. This model used a bi-direction encoder and attention mechanism-based decoder. The authors used the Papaya English dialogue dataset for training the Chatbot system. The experimental results of this model show that LSTM produced the highest score as compared to other models.

MHMC chitchat dataset used by the author Ming-Hsiang Su et al. [20]. This dataset contains daily conversations of older adults. This Chatbot is based on a multi-layered LSTM embedding system that extracts semantic information. The research used Euclidean distance to select proper question patterns that give the corresponding answer. H. Honda et al. [7] also attempted to use deep learning for improvement in the performance of Q&A system. Here symbolic processing is learned with the help of the Neural Machine Translation (NMT), Word2Vec training, deep learning models. For AI it is a difficult task to handle unknown data thus author used Word2Vec. The main problem with the standard LSTM model was the inability of a neural network to store all the information from the text. Therefore Y. Sharma et al. [19] proposed a dynamic memory network with LSTM model. The authors also used various approaches like basic NLP, Deep learning algorithm.

3. DESIGN OF CHATBOT USING GRU WITH BEAM SEARCH

The Fig.1 shows architecture of the proposed system. The proposed Chatbot system is based on Seq2Seq encoder-decoder architecture which is trained using the Cornell movie dataset. Initially, data is pre-processed by the proposed system. The input and output are in the form of dialogues during the training stage of the proposed model. The length of the input and output dialogues are different; hence training pre-processing adds a few tokens. Input sentence containing tokens is passed to the encoder. In the encoder, each input word is converted into its vector form. The output of the encoder is the input of the decoder. The attention mechanism is applied at the decoder stage, which focuses on the important part of the decoder. We applied the Beam search decoding algorithm in the proposed model at the decoder stage. The output of the decoder gives the proper response. The proposed model is tested by comparing the generated output and actual output.

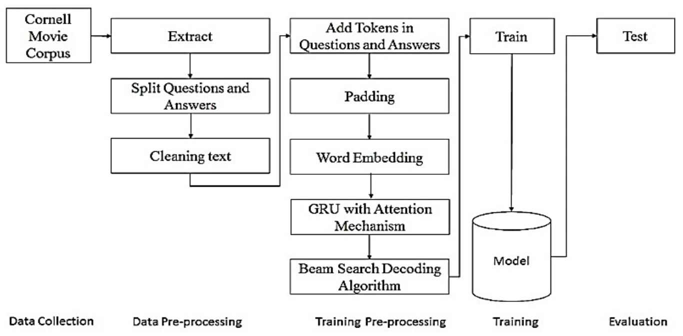

Fig.1. Proposed system architecture

In this section, subsection 3.1 introduces the Seq2Seq model. Later subsection 3.2 discusses the attention mechanism. Subsection 3.3 shows the training of the Seq2Seq model and Beam search decoding algorithm.

3.1 SEQUENCE TO SEQUENCE MODELS

The proposed system is based on Seq2Seq encoder-decoder architecture. The Seq2Seq model correlates the input of system with the output. Here, input length may be different from length of the output. This model is comprised of three important components viz. encoder, intermediate vector, and decoder. The encoder accepts a single element of the input. Here the input is the collection of the words from the user question. Then encoder vector is used to encapsulate the words of the input. The input word order helps the decoder give the correct response. The decoder calculates the response using a hidden state with respect to its weights. Sometimes Seq2Seq model fails due to the vanishing exploding gradients problem. Many variations are developed to solve this problem such as LSTM and GRU. In the proposed model, we explored both LSTM and GRU encoder-decoder architecture.

The LSTM depends on its gated structure. In the proposed model, any dialogue or sentence is a pass to the LSTM encoder. Each word from a given input sentence is converted with respect to its vector form. This vector form is a pass to the LSTM first layer which is known as the forget gate layer. Here LSTM encoder decides information that is going to be considered. Then the output of the forget gate layer passed to the input and output gate layer respectively. Finally, the LSTM encoder generates an output vector, which is an intermediate vector.

This intermediate vector is inputted to the LSTM decoder which contains all information about the input sequence. The LSTM decoder uses an intermediate vector to generate the target sequence. As shown in Fig.1 we perform tokenization in the data pre-processing part. LSTM stops decoding when the end of the sentence token arrives. The LSTM decoder gives a response as per the input given to the encoder.

The vanishing exploding gradients problem is effectively resolved using GRU [17]. The working of GRU and LSTM is similar but there is a difference in gate structure. Here vector form of the input sentence given to the GRU first layer. The first layer of the GRU encoder is the update gate layer. The update gate performs concatenation between the current input and the previous output. This can be done using the following Eq.(6).

$$Z_t = W^z * x_t + U^z * h_{t-1} \tag{1}$$

Here, x_t is current input given to the proposed model GRU encoder with respect to time t. The previous output with respect to time t-1 is h_{t-1}. In the update, gate multiplication performs with respect to its corresponding weights. Here the weights are W^z and U^z. After performing concatenation, the output of the equation i.e. Z_t is a pass to the activation function. The update gate is helpful to identify a part of past information with respect to its time that needs to be considered in the future. With the help of the update gate, the proposed model eliminates the risk of vanishing gradient problems. The output of the update gate is a pass to the reset gate. This gate is used to determine which information should be removed from the previous in the proposed model [6]. To calculate the output of the reset gate following equation is used.

$$r_t = W^\gamma * x_t + U^\gamma * h_{t-1} \tag{2}$$

The above equation is the same as the update gate equation. There is a difference in the gate usage and the weights. Here weights are W^γ and U^γ is multiplied with respect to x_t and h_{t-1}. This gate also removes unnecessary data from the proposed GRU encoder model. Here we need to store relevant information from the past using the below equation in the GRU encoder with respect to its current content. In the above equation, the output of the reset gate is used as input, and then in the proposed model tangent activation function is applied [6]. This is shown using equations number 3 and 4.

$$h_t' = \tanh(W * x_t + r_t * U * h_{t-1}) \tag{3}$$

$$h_t = Z_t * h_{t-1} + (1 - Z_t) * h_t' \tag{4}$$

Finally, a single vector is generated, which is the output of the proposed GRU encoder. The output of this equation depends on all the previous gate outputs. Then GRU decoder generates a proper response as per the input given. In the proposed model reset and update gate determines the information that should be passed to the output [17]. The GRU encoder-decoder model can keep all the data as long as feasible.

3.2 SEQUENCE TO SEQUENCE WITH ATTENTION MECHANISM

In the previous section, we discussed the working of GRU encoder-decoder model. Sometimes this model fails to remember longer sentences. Another problem with these models is that they can't give more importance to specific words. Hence, we are using attention mechanisms in the proposed model. The main idea of this mechanism is to concentrate on the vital part of the information. The attention mechanism work at the encoder state where the most critical information is available in the proposed model [11]. The most important part of this mechanism is the alignment function, Softmax function, and context vector. In our model, we are using bi-directional GRU. At the encoder step, the attention mechanism concatenates forwarding and backward states.

The proposed model input sentence contains T_x number of the word and h_j is the number of vectors. In this model, only the last state of the GRU encoder is used as a context vector. The context vector is calculated using the following equation.

The proposed model input sentence contains T_x number of the word and h_j is the number of vectors. In this model, only the last state of the GRU encoder is used as a context vector. The context vector is calculated using the following equation.

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} * h_j \tag{5}$$

We can calculate output of the GRU encoder using the value of c_i. We can calculate α_{ij} using Softmax and the alignment function. The value of c_i is passed to the decoder which generates the most appropriate response to the user.

3.3 TRAINING SEQUENCE TO SEQUENCE MODEL AND BEAM SEARCH DECODING

The main goal of the proposed model in the training state is to reduce the difference between actual output and predicted output generated by this model. During the training of the proposed model, the predicted answer is given to the next time step, this makes the training process slow. The proposed model inputs the actual output sentence to the decoder to speed up this. At each time step, the decoder generates output using a set of vocabulary. Here the vocabulary size in this proposed model is 660.

In the previous section, we discussed specific behavior that speeds up the training of the proposed model. In the proposed model output of the decoder depends on its word selection. The word which has the highest probability is selected from our vocabulary size. But at all times this can't give the correct answer. Hence we perform decoding at a testing time using Beam search. This heuristic search investigates a graph by extending the node in a restricted set. For building a search tree, it uses breadth-first search. It considers only N (beam size) topmost nodes at every level in the memory. These N nodes are used to expand the next level. This makes a tree that contains top results. Beam search has a parameter called B, which is called beam width. The decoder generates the output in this proposed model with an overall 660 possible steps. Here we consider only the top 15 words because of beam size 15. Here, we find the most relevant response until the Beam search finds the end of sentence token.

4. EXPERIMENTAL WORK

We explored both LSTM and GRU architectures with the Beam search decoding algorithm as part of experimental work. In this section, we present a comparative study on the performance of both architectures. Section 4.1 discusses the dataset used for experimental work. Section 4.2 discusses performance measurement parameters. Section 4.3 provides information about experimental results and analysis of the LSTM and GRU.

4.1 TRAINING DATASET

In this work, we used the popular "Cornell Movie Subtitle Corpus" [4] as a dataset. Cristian Danescu-Niculescu-Mizil and Lillian Lee developed this corpus at Cornell University. The movie corpus has 220,579 conversations. These conversations are dialogues between 10,292 pairs of characters from movies. There are total 9,035 different characters from 617 movies. The total number of utterances are 304,713. Metadata of movies have some other information like genres of movie, IMDB rating, release year, number of users who voted IMDB, etc.

4.2 TRAINING MODEL AND PARAMETERS

This research work explored bi-directional GRU. A neural attention mechanism has been applied to improve performance. We used the Luong attention mechanism. Here, 16 samples were used before the model got updated with respect to hidden layers containing 256 vectors. In the proposed model, we want global minima, so we set the learning rate as 0.001 and use two hidden layers.

The word embedding is nothing but converting text into numbers; we used embedding size as 128. For optimization of the proposed model, we used the Beam search algorithm. The decoder can select the topmost 15 answers using the Beam search. In one Epoch entire dataset is processed through the NN in both directions i.e., forward and backward. We used 40 Epochs during the performance testing. These hyper parameters are shown in Table.1.

Table.1. GRU Training Hyper-Parameters

Hyper-parameter Name	Value
Vocabulary size	660
Batch size	16
Number of Epochs	40
Size of Layer	256
Number of Layers	2
Embedded size	128
Beam size	15
Learning size	0.001
Encoder Type	Bidirectional

4.3 EXPERIMENTAL RESULTS

Section 4.3.1 discusses the experimental results of Epoch vs Loss analysis for LSTM and GRU. Section 4.3.2 shows graphical results of Epoch Vs Accuracy and results of Epoch vs Time required for training.

4.3.1 Comparing Loss on Various Epoch:

As a major aspect of the optimization algorithm, errors for the present state of the model must be assessed frequently. This requires us to choose an error function; the function is called a loss function. The weights can be updated to diminish the loss in the next state. Here we chose the weighted Softmax cross-entropy loss function. Eq.(6) and Eq.(7) give details of the loss function and cost function used in the proposed work.

$$\textit{Loss function} = (\textit{actual value-predicted value})^2 \quad (6)$$

$$\textit{Cost function} = \sum(\textit{actual value-predicted value})^2 \quad (7)$$

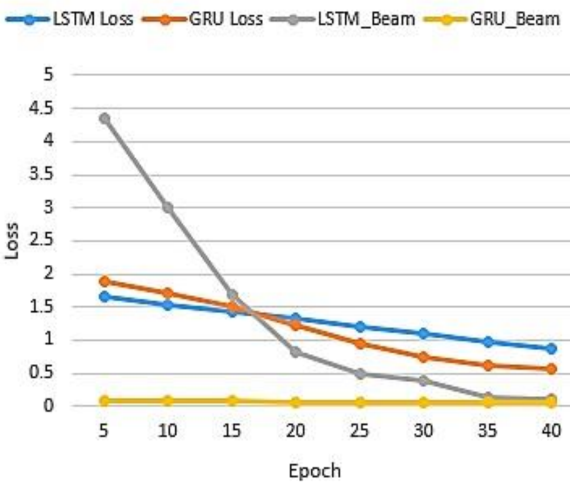

Fig.2. Loss Comparison on Various Epoch for LSTM and GRU

The Fig.2 shows the graphical results of loss comparison on various epochs for LSTM and GRU with and without using the Beam search algorithm. Compared to LSTM, the loss of GRU decreases rapidly with an increase in the number of epochs. The loss of LSTM at epoch number 40 is 0.873368, which is higher than GRU. The loss of GRU at epoch number 40 is 0.565367. After using the Beam search algorithm loss of LSTM at epoch number 40 reduces to 0.117084, and the loss of GRU becomes 0.068763. Here GRU with Beam search shows the minimum loss compared to other approaches.

4.3.2 Comparing Accuracy for Various Epoch:

The formula to derive accuracy is shown in Eq.(8). The Fig.3 shows the graphical results of LSTM and GRU accuracy with and without using a Beam search algorithm. As the number of epochs increases, accuracy also increases in LSTM and GRU. The accuracy of GRU is higher than LSTM. We observed the accuracy of LSTM at epoch number 40 as 0.837000. The accuracy of GRU at the same epoch is 0.880000. After using Beam search accuracy of LSTM is 0.986832, whereas for GRU it is 0.991980.

$$\textit{Accuracy}=((\text{No. of correct predictions}))/(\text{Total number of Questions}) \quad (8)$$

We measured the training time required for both LSTM and GRU. The experimental results are shown in Fig.4. The training time required for LSTM is greater than GRU. Training time per epoch in GRU is constant after some epoch. We observed the training time of LSTM at epoch number 40 as 25 seconds, whereas it is 21 seconds for GRU. After using a Beam search algorithm training time gets reduced. The training time of LSTM with the Beam at epoch number 40 is 5 seconds and for GRU with Beam it is 4 seconds.

We applied LSTM and GRU to generate a Chatbot responses in this research work. In experimental work, we observed that GRU with Beam search decoding gives better results compared to LSTM with Beam search. The GRU with Beam search model has the ability to keep all the information as long as feasible. This model takes less training time and performs better than LSTM.

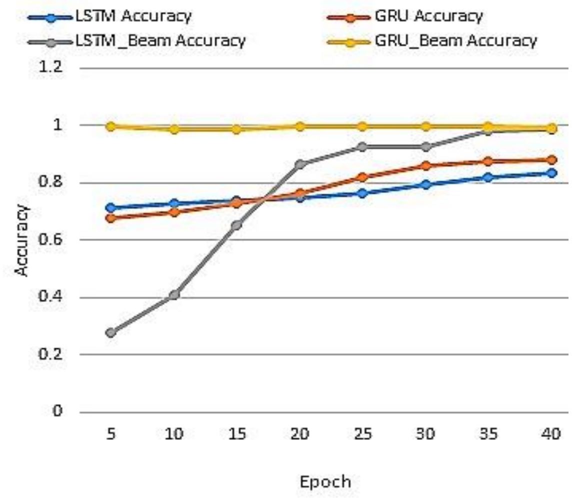

Fig.3. Accuracy Comparison on Various Epoch for LSTM and GRU

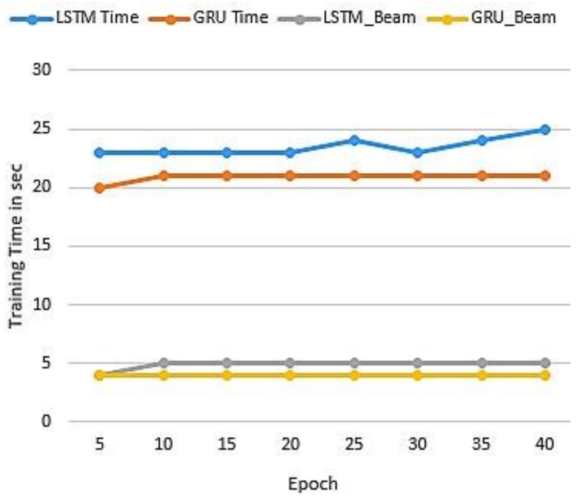

Fig.4. Training Time Comparison on Various Epoch for LSTM and GRU

5. CONCLUSION

Nowadays Chatbot systems are based on Seq2Seq architectures. There are a few challenges in using Seq2Seq encoder-decoder model for building Chatbot systems. The Seq2Seq models cannot handle variable-length sequences and slow down the training process. As the length of the input sentence increases the Seq2Seq model loses information. In this research work, we addressed these limitations by designing a Chatbot system using the GRU model and Beam search decoding mechanism. The research work used Cornell Movie Corpus as a training dataset for the Chatbot. The GRU encoder takes sequence by sequence input and processes one at a time. With the help of an attention mechanism, GRU is able to focus on only important information from the input sequence. The GRU decoder using Beam search algorithm generates a proper response.

After analyzing experimental results, we conclude that a number of epochs have a direct impact on the accuracy of the model. The accuracy of the GRU with the Beam search algorithm for 40 epochs is higher compared to other models. The GRU with the Beam search algorithm gave 99.198% accuracy which is better than the LSTM. The GRU with Beam search gave approximately 6.8763% loss which is better than the LSTM. Though LSTM gives us the most control ability it is more complex than GRU and exhibits higher cost. GRU uses fewer parameters therefore it required minimum memory. It is also observed that GRU is faster than LSTM. Also, GRU with Beam search decoding shows improvement in the responses generated by the Chatbot system. Using the Beam search algorithm this model selects only the topmost predicted answers and keeps it in memory at each time step. The Chatbot System can be further improved with high-quality real-life conversation datasets.

REFERENCES

[1] A. Argal, S. Gupta, A. Modi, P. Pandey, S. Shim and C. Choo, "Intelligent Travel Chatbot for Predictive Recommendation in Echo Platform", *Proceedings of IEEE International Conference on Computing and Communication*, pp. 176-183, 2018.

[2] L. Chen, P. Chen and Z. Lin, "Artificial Intelligence in Education: A Review", *IEEE Access*, Vol. 8, pp. 75264-75278, 2020.

[3] Andry Chowanda and Alan Darmasaputra Chowanda, "Generative Indonesian Conversation Model using Recurrent Neural Network with Attention Mechanism", *Proceedings of IEEE International Conference on Computer Science and Computational Intelligence*, pp. 433-440, 2018.

[4] Cristian Danescu-Niculescu-Mizil and Lillian Lee, "Chameleons in Imagined Conversations: A New Approach to Understanding Coordination of Linguistic Style in Dialogs", *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics*, pp. 1-12, 2011.

[5] G. Dzakwan and A. Purwarianti, "Comparative Study of Topology and Feature Variants for Non-task-oriented Chatbot using Sequence to Sequence Learning", *Proceedings of IEEE International Conference on Advanced Informatics: Concept Theory and Applications*, pp. 135-140, 2018.

[6] Hisham El-Amir and Mahmoud Hamdy, "*A Tour Through the Deep Learning Pipeline*", Oxford Press, 2020.

[7] H. Honda and M. Hagiwara, "Question Answering Systems with Deep Learning-based Symbolic Processing", *IEEE Access*, Vol. 7, pp. 152368-152378, 2019.

[8] B. Kohli, T. Choudhury, S. Sharma, and P. Kumar, "A Platform for Human-Chatbot Interaction using Python", *Proceedings of IEEE International Conference on Green Computing and Internet of Things*, pp. 439-444, 2018.

[9] P. Kumar, M. Sharma, S. Rawat and T. Choudhury, "Designing and Developing a Chatbot using Machine Learning", *Proceedings of IEEE International Conference*

on System Modeling Advancement in Research Trends, pp 87-91, 2018.

[10] R.B. Mathew, S. Varghese, S.E. Joy and S.S. Alex, "Chatbot for Disease Prediction and Treatment Recommendation using Machine Learning", *Proceedings of IEEE International Conference on Trends in Electronics and Informatics,* pp.851-856, 2019.

[11] T. Nguyen and M. Shcherbakov, "A Neural Network Based Vietnamese Chatbot", *Proceedings of IEEE International Conference on System Modeling Advancement in Research Trends*, pp. 147-149, 2018.

[12] A. Nursetyo, D.R.I.M. Setiadi and E.R. Subhiyakto, "Smart Chatbot System for E-Commerce Assitance Based on AIML", *Proceedings of IEEE International Conference on Research of Information Technology and Intelligent Systems,* pp. 641-645, 2018.

[13] M. Nuruzzaman and O. K. Hussain., "A Survey on Chatbot Implementation in Customer Service Industry Through Deep Neural Networks", *Proceedings of IEEE International Conference on E-Business Engineering*, pp. 54-61, 2018.

[14] E. Ozsarfati, E. Sahin, C.J. Saul and A. Yilmaz, "Genre Classification Based on Titles with Comparative Machine Learning Algorithms", *Proceedings of IEEE International Conference on Computer and Communication Systems*, pp. 14-20, 2019.

[15] K. Palasundram and A. Azman, "Enhancements to The Sequence-to-Sequence-based Natural Answer Generation Models", *IEEE Access*, Vol. 8, pp. 45738-45752, 2020.

[16] S. Prasomphan, "Improvement of Chatbot in Trading System for SMEs by using Deep Neural Network", *Proceedings of IEEE International Conference on Cloud Computing and Big Data Analysis*, pp. 517-522, 2019.

[17] D. Ren, Y. Cai, W. H. Chan and Z. Li, "A Clustering Based Adaptive Sequence-to-Sequence Model for Dialogue Systems", *Proceedings of IEEE International Conference on Big Data and Smart Computing*, pp. 775-781, 2018.

[18] P. Rivas, K. Holzmayer, C. Hernandez and C. Grippaldi. "Excitement and Concerns about Machine Learning-based Chatbots and Talkbots: A Survey", *Proceedings of IEEE International Conference on Technology and Society*, pp. 156-162, 2018.

[19] Yashvardhan Sharma and Sahil Gupta, "Deep Learning Approaches for Question Answering System", *Proceedings of IEEE International Conference on Computational Intelligence and Data Science*, pp. 785-794, 2018.

[20] M. Su, C. Wu, K. Huang, Q. Hong and H. Wang, "A Chatbot using LSTM-based Multi-layer Embedding for Elderly Care", *Proceedings of IEEE International Conference on Orange Technologies*, pp. 70-74, 2017.

[21] M. Tabiaa and A. Madani, "The Deployment of Machine Learning in E-Banking: A Survey", *Proceedings of IEEE International Conference on Intelligent Computing in Data Sciences*, pp. 1-7, 2019.

[22] Q. Yang, Z. He, F. Ge and Y. Zhang, "Sequence-to-Sequence Prediction of Personal Computer Software by Recurrent Neural Network", *Proceedings of IEEE International Conference on Neural Networks,* pp. 934-940, 2017.