Effective Chatbots using Machine Learning and Natural Language Processing

John Ivanov,[1] Prajval Sharma,[2] and Yarwin Liu[3]

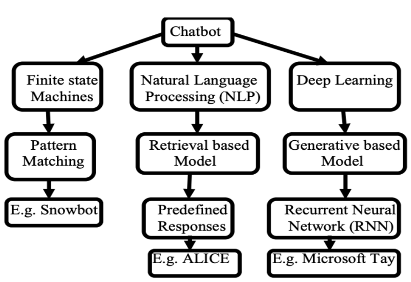
Tesoro High School, ivanov.john04@gmail.com,[1] Cupertino High School, prajrocks25@gmail.com,[2] Aliso Niguel High School, yarwinliu@gmail.com[3]

ABSTRACT

Our goal is to produce an effective chatbot that understands a user input of a sentence or question and responds with an appropriate output, improving on the current design of modern chatbots. Chatbots have the capacity to profoundly impact virtual communication for accommodation, assistance, and even emotional support because of their broad applicability. In addition, they are essential in bridging the gap between humans and machines: the perfection of chatbots would enable machines to understand human emotions, the stepping stone before human thoughts can be understood. In this paper, we have created a general generative chatbot, which makes use of a sequence to sequence model, consisting of an encoder and decoder, both of which are recurrent neural networks (RNN), and generates textual output mimicking a human response, with commonly used slang and acronyms. The encoder converts input text into indices, which are then converted back to text after going through the model. Through our experiments, we have observed that with a low learning rate of 0.00001, the accuracy, or similarity of the output to human response, increases with a downward concavity comparative to that of a logarithmic graph, and that loss decreases with an upward concavity.

I. INTRODUCTION

Chatbots, also known as bots, are software used to communicate with humans, through text as well as voice. In the last decade, they have been increasing in number due to technological advancements in machine learning. One of the first known bots, ELIZA, in 1966 matched user prompts to predetermined responses, marking the beginning of machine-human interaction [10]. Since then, traditional chatbots, with their scripted responses, have been replaced by conversational chatbots that have the ability to learn and adapt with the input of new information [6]. In 1995, the Artificial Linguistic Internet Computer Entity (ALICE) made use of natural language processing (NLP) to interpret user input [10]. Today's chatbots are built on generations of previous models and utilize deep neural networks that collaborate with NLP to produce a model responding to any user input, which is much closer to real human interaction.



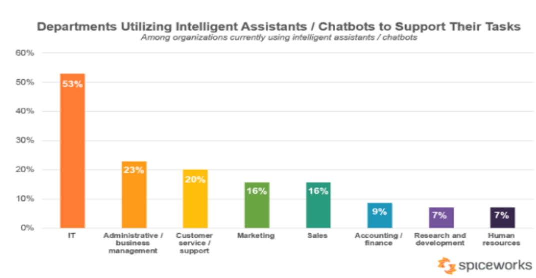


Fig 1: Classification of Chatbot types [6]

Fig. 2: Use of Chatbots by Department [2]

With increased development, the rules that frame chatbots are diminishing as networks can train the model to respond effectively to more comments or questions from the user [1]. There are currently many different types of chatbots. As Figure 1 shows, some bots generate text based solely on pattern matching. Our goal is to create a dynamic generative-based bot utilizing deep learning technologies and RNNs, similar to the third branch of the diagram.

As a result of the flexibility in chatbot structures, they have various uses. As shown in Figure 2, most of the chatbots and intelligent assistants are currently being used by the information technology (IT) industry, which focuses on technology and development. However, this paper aims to promote the use of bots in the human resources (HR) department, which only uses around 7% of bots [2]. This is because the HR department consists mainly of communication, so bringing chatbots to this field will verify that artificial intelligence (AI) can have natural conversations with humans.

For the model to effectively respond to a user, a machine learning model with a RNN that utilizes NLP techniques to understand the data would be necessary. This is because it is essential to preprocess the data down to its roots and intents, which uses less computational power and time. The overall process is to 1) make the model understand the input and then 2) make the model produce an appropriate output. Step 1 would require natural language understanding (NLU), which can be done through lexical, syntactic, semantic, and pragmatic analysis [1]. Once this is done, the next step would be natural language generation (NLG), in which the model will generate a response based on what has been learned from prior inputs. It is recommended to approach this through RNNs as they extend feedforward networks [6]. Using long short-term memories (LSTM) or gated recurrent units (GRU) are useful for remembering long sequences of data because information is updated, rather than replaced, as new inputs are given [10]. A support vector machine (SVM) can also be used to classify questions and answers [10]. A third step known as dialog management can be included between the NLU and NLG [5]. This step is essential because it helps the bot choose an appropriate response, but it is frequently merged with NLP. One paper also claimed a deep learning model that included both a convolutional neural network (CNN) and an LSTM/GRU [9]. Most datasets were freely available on the internet, and many research papers used Reddit feeds and replies; one article recommended using movie dialogues.

The highest success rate over time was the supervised learning + imitation learning + reinforcement learning model, with an accuracy of 84.57% [7]. In another fairly accurate model, a team used a transformer model that pushed their accuracy up to 61% [8]. However, due to our time constraint of four weeks, we have used a sequence to sequence model. It was also mentioned that cleaning data before feeding dramatically improved the sentence error rate [8]. We wish to integrate all of the advice given by these papers to minimize our error and perfect the model output through our research.

This article tries to improve upon this method by utilizing state-of-the-art machine learning algorithms to approach this issue. For example, we will make use of NLP, which uses restrictive vocabularies to simulate realistic conversations through pattern matching [6]. The central aspect that we are trying to improve on is understanding user input and framing sentences appropriate for the language it is trained on and communicated to with. This technology can easily be applied to virtual assistants such as Alexa or Google Assistant by converting from text to speech. These assistants focus on short interactions with users with specific intent as they are commercial consumer products [5]. However, these intentions can be changed to fit various needs, such as customer service.

II. METHODOLOGY

A. Datasets

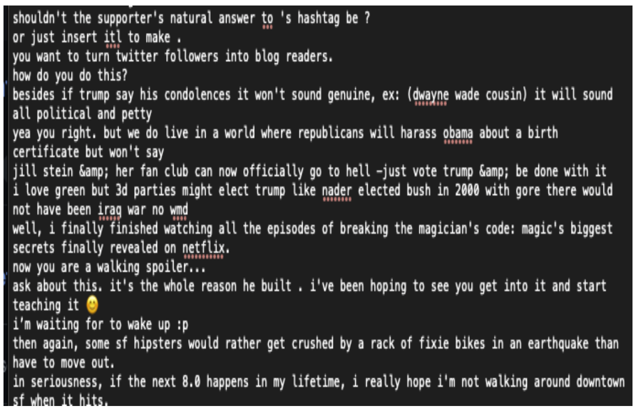


Figure 3: Twitter Dataset

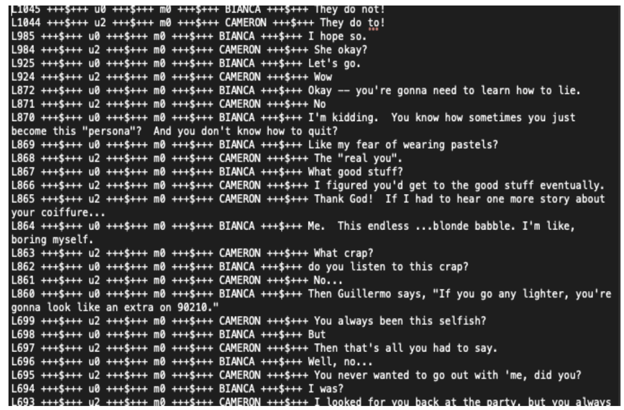


Figure 4: Cornell Dataset

We used a dataset from Twitter to train the model to recognize informal language and commonly used but improper spellings and abbreviations such as “u” and “lol.” The question and answer format allowed for a more natural conversation flow. A Cornell movie dialog dataset trained the chatbot to carry out conversations with more variation in sentence structure and language. It contains over 220,000 conversations collected from various movies.

B. Data Pre-Processing

Before training, it is essential to feed the data to the model in a format that can be processed. We have split our data into training, testing, and validation with a ratio of 0.5: 0.25: 0.25, respectively. The words are added to a vocabulary dictionary, and are then padded to make each word the same length as the one with maximum characters. This padded portion would appear as zeros. Capitalization and punctuation are also removed for easier machine comprehension [5].

C. Sequence-to-Sequence Model

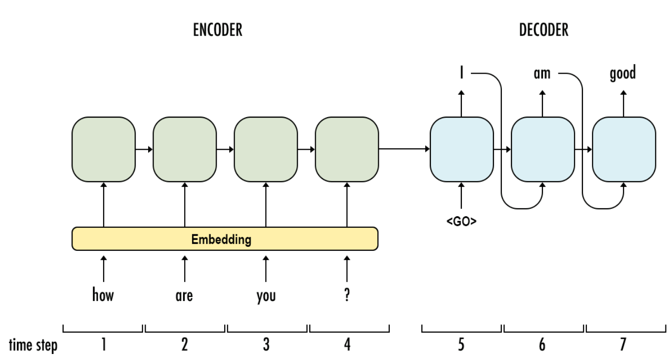


Figure 5: Sequence to sequence model with an encoder and decoder [3]

There are three layers to the decoder of the sequence to sequence model, which takes a sentence as input and returns another variable-length sequence as output. As shown by Figure 5, it is done through the use of an encoder and a decoder [1]. Both are RNN in which cells of each layer are connected to cells on the same layer and adjacent layers. This means that there may be self-feedback connections and decisions influenced by what has been learned from prior inputs. This encoder process is used to produce a vector that describes the essentials and intent of the input.

The encoder iterates through the input one token at a time, outputting an output vector, recorded, and a hidden state vector, which is passed onto the next time step. Our encoder is where we include an embedding layer that takes the one hot coded input of the words and trains the embedded network to place words with similar meaning close to each other in the vector space thus giving them similar representation in numerical form for the model to process. After this process has been finished, the data is fed to a multi-layered bidirectional GRU, which has two RNNs. One is given the input sequence in the standard order, and the other is fed the same series, reversed. This has the effect of encoding both past and future contexts [4]. Figure 6 shows the steps of the encoding process.

The decoder creates the chatbot's response token by token, as shown in Figure 7. Using context vectors generated by the encoder and hidden internal states to predict the next word in the sequence generates words until the end of a sentence. The decoder takes an input word and the context vector and returns a guess for the next word in the sequence and a hidden state to use in the next iteration. Using the context that the encoder transforms, the decoder will use it to generate a meaningful output for the given task.

Computation Graph:

1. Convert word indexes to embeddings.
2. Pack padded batch of sequences for RNN module.
3. Forward pass through GRU.
4. Unpack padding.
5. Sum bidirectional GRU outputs.
6. Return output and final hidden state.

Figure 6: Encoder computation graph [4]

Computation Graph:

1. Get embedding of current input word.
2. Forward through unidirectional GRU.
3. Calculate attention weights from the current GRU output from (2).
4. Multiply attention weights to encoder outputs to get new "weighted sum" context vector.
5. Concatenate weighted context vector and GRU output using Luong eq. 5.
6. Predict next word using Luong eq. 6 (without softmax).
7. Return output and final hidden state.

Figure 7: Decoder computation graph [4]

LSTM networks are an extension of RNN, which maximizes the chance of generating a response given the previous conversations [10]. In the encoder-decoder architecture that we use, the encoder and decoder are both GRU.

The benefit of using a GRU instead of an LSTM is that the GRU has two gates, whereas the LSTM has three. Therefore, even though a GRU is less complex, it exposes the complete memory and the hidden layers. GRU is better in the sense that it yields quick results.

D. Optimization

An Adam optimizer, or an optimization algorithm that uses individual learning rates for each parameter and updates learning rates based on search results, is used to optimize the results. The advantage of this approach is that the learning rate adapts to different parameters, in contrast to the stochastic gradient descent method.

Cross-Entropy is widely used in machine learning as a loss function. Cross-Entropy is a measure from information theory that builds upon entropy and generally calculates the difference between two probability distributions. In our model, the Cross-Entropy loss function calculates the average negative log-likelihood of the elements that correspond to a 1 in the masking tensor.

E. Conducted Experiments

We evaluated our chatbot on 20 different inputs and saw the top 3 responses to test our chatbot. In addition, we have created a table to show the bot's evaluation and a graph to show the relationship between loss and the number of iterations. This will help in showing where the curve flattens, and additional training will become computationally expensive.

III. ANALYSIS

A. Results

Input	**Output**
How is life	Its wild lol
Do you know me	I was just thinking about it
How is life	its seriously wild
Do you want to be president	I don't know what to say
Can i help you	ill be doing this more than you
Can i help you	Follow me back
Are you smart	No, not yet, lol

Figure 8: Positive example inputs and outputs from the chatbot

Figure 8 shows examples of output that correctly responded to each input respectively.

Input	Output
Where are you	We have one before the sun very badly
Are you funny	Do you think youll make sure I want one
Do you want to be president	I didnt get the message

Figure 9: Negative example inputs and outputs from the chatbot

Figure 9 shows examples of output thats do not properly responded to each input respectively

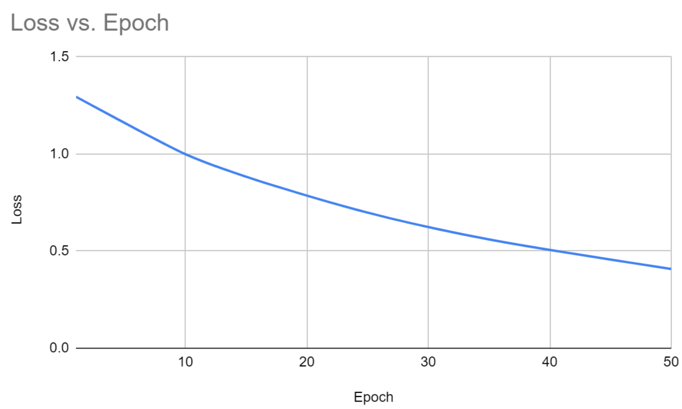


Figure 9: Graph demonstrating the change in loss over the epochs

The graph in Figure 8 shows the decrease in the loss as the model is trained over the 50 epochs. It is illustrated that as the model reaches a higher epoch number, the loss decreases. At epoch one, the loss is 1.3 at epoch 5, the loss is 1.1, at epoch 15, the loss is 0.89, at epoch 25, the loss is 0.7, at epoch 35, the loss is 0.56, and at epoch 50, the loss is 0.41.

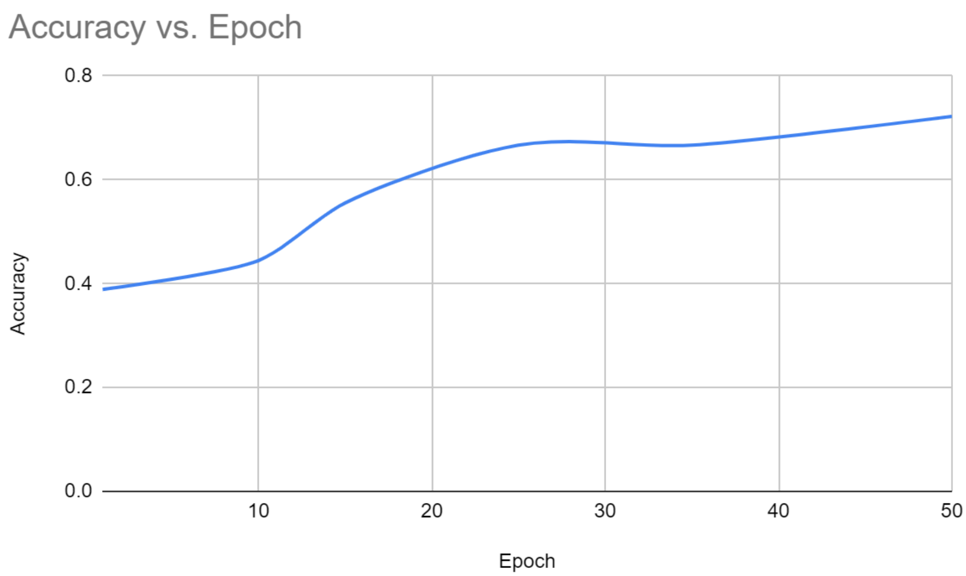


Figure 10: Graph demonstrating the change in accuracy over the epochs

The graph in Figure 9 shows the increase in accuracy as the model is trained over the 50 epochs. The graph highlights how the accuracy increases as the model reaches a higher epoch number. At epoch 1, the accuracy is 38.9%, at epoch 10, the accuracy is 44.4%, at epoch 15, the accuracy is 55.6%, at epoch 35, the accuracy is 66.7%, and at epoch 50, the accuracy is 72.2%.

B. Evaluation

Since our experiments deal with a chatbot attempting to mimic human communication, it made sense to include sample output for questions a human might ask to judge the capability of the model.

Besides the end output, we have also tried predicting the rate at which the model will learn when it is trained over more iterations. This is why we have included a graph that shows the learning loss in comparison to the number of epochs that the model was trained on. We noticed an expected decrease in loss as the epoch number increased. Our accuracy versus a number of iteration graphs also showed an increase with a negative concavity as expected. What we found fascinating is that the first time we ran our experiments, even though the accuracy was increasing, the loss was increasing as well.

Our loss was cross-entropy loss and we used the Adam optimizer for minimizing our loss.

C. Limitations

As we had just four weeks to plan and get our results, we could not experiment with various models. We used the sequence to sequence model that was common for the work we are doing. Therefore, we are expecting similar results to other papers. Nevertheless, we have been adding informal language to our dataset in hopes that the model grasps trends and slang during the time period. We have presented our output, loss rate, and accuracy and how they depend on the number of iterations the model is trained on.

Another challenge that arose from the short time frame was not being able to train the model on a high number of iterations. Our upper limit, as shown by the graph, was 50 epochs. Thus, the curves could not be analyzed beyond this point.

IV. DISCUSSION

In this paper, we experimented with different learning rates and epoch numbers as well as various styles of datasets. We demonstrated that with an increase in epoch number, the accuracy of the model increased; meanwhile, the loss decreased as the epoch number increased. Due to the use of the Twitter dataset and a movie dataset, which contained numerous data points incorporating slang, we noticed that our chatbot was able to respond to the users' questions using modern-day slang. For example, Figure 8 shows the user inputting the question "How is life," to which the chatbot responds, "its wild lol."

Initially, as we were running the model with a learning rate of 0.001 which was 10 times faster than the rate we used, we found that the accuracy was increasing and the statements were becoming more human-like even as the learning loss was increasing. We assumed that the model was becoming overconfident in its responses so that in the instances where it was wrong, the loss was high. Since this was the case, the second time, we ran the model with a learning rate of 0.00001 which was 100 times smaller.

Future work may entail studying the effectiveness of different models and using more data points to find more optimal learning rates, which would not only decrease the training time but also the loss significantly.

Another trend that was observed was that the model was having trouble answering questions that were personal to it. Such questions would include "who are you?", "What is your name?" and "Tell me about yourself?". Since this is an artificial network, it would help to create a "resume" stored in the memory that includes sentences about the bot and its personality. For this, a simpler retrieval-based model could be used. Even though this seems counterintuitive, it would help the model increase accuracy in those certain questions.

REFERENCES

[1] Ayanouz, Soufyane, et al. "A Smart Chatbot Architecture Based NLP and Machine Learning for Health Care Assistance." *ResearchGate*, Association for Computing Machinery, 31 Mar. 2020, www.researchgate.net/publication/340678278_A_Smart_Chatbot_Architecture_based_NLP_and_Machine_Learning_for_Health_Care_Assistance.

[2] Brain. "Chatbot Report 2019: Global Trends and Analysis." *Medium*, Chatbots Magazine, 19 Apr. 2019, chatbotsmagazine.com/chatbot-report-2019-global-trends-and-analysis-a487afec05b.

[3] Chablani, Manish "Sequence to Sequence Model: Introduction and Concepts." *Medium*, Towards Data Science, 23 June 2017 towardsdatascience.com/sequence-to-sequence-model-introduction-and-concepts-44d9b41cd42

[4] “Chatbot Tutorial¶.” *Chatbot Tutorial - PyTorch Tutorials 1.9.0+cu102 Documentation*, Pytorch, pytorch.org/tutorials/beginner/chatbot_tutorial.html?highlight=chatbot+tutorial.

[5] Fang, Hao, et al. “Sounding Board: A User-Centric and Content-Driven Social Chatbot.” *Arxiv*, Cornell University, 26 Apr. 2018, arxiv.org/abs/1804.10202.

[6] Jwala, K. "(2019, June)." (n.d.): Jwala, K., Sirisha, G. N. V. G., & Raju, G. V. P. (2019, June). *Developing a Chatbot using Machine Learning*. https://www.ijrte.org/wp-content/uploads/papers/v8i1S3/A10170681S319.pdf.

[7] Liu, Bing, et al. “Dialogue Learning with Human Teaching and Feedback in End-to-End Trainable Task-Oriented Dialogue Systems.” *Aclanthology*, Association for Computational Linguistics, June 2018, aclanthology.org/N18-1187/.

[8] Mazarè, Pierre-Emmanuel, et al. “Training Millions of Personalized Dialogue Agents.” *Arxiv*, Cornell University, 6 Sept. 2018, arxiv.org/abs/1809.01984.

[9] Siddhant, Aditya, et al. “Unsupervised Transfer Learning for Spoken Language Understanding in Intelligent Agents.” *Arxiv*, Carnegie Mellon University, 13 Nov. 2018, arxiv.org/pdf/1811.05370.pdf.

[10] Suta, P., Lang, X., Wu, B., Mongkolnam, P., & Chan, J. H. (2020, April 4). *An Overview of Machine Learning in Chatbots*. http://www.ijmerr.com/uploadfile/2020/0312/20200312023706525.pdf.

ACKNOWLEDGEMENTS

We would like to thank Dr. Ryan Solgi, our mentor and teacher whose knowledge and insight has made this paper possible. Dr. Solgi has introduced us to many models and methods to solve deep learning problems which have helped us in our project. We would also like to thank Laboni Sarker and S. Shailja, who have helped revise the paper and provided useful input on how to improve it.

AUTHOR CONTRIBUTION STATEMENT

P.S. conceived experiments and ran them. J.I and Y.L. analyzed results, revised the paper, and created the charts as well as data tables. All authors reviewed and wrote the paper.