AI Voice Assistant Chatbot

Final Report

By Alish Kadiwal (200394974), Dhruv Modi (200392219), Ria Chevli (200416701)

Table of Contents

Project Member (2) :

Alish Kadiwal
Dhruv Modi
Ria Chevli

Project Summary (A) :

We have developed a voice assistant chat-bot where instead of typing, the user(s) can send their request by speaking. Our AI chatbot will interact with the user based on general questions and likeability. We will be storing any information related to the user and based on the previous conversation, it would help our machine to search into specific information related to google and in our files. We will also utilize other libraries and packages which could help with searching for response through Google, Wikipedia, and Mathematical computations. We can also be performing OS based actions through our bot.

Description of the Problem (B) :

Not able to understand Natural Language

While chatbots can answer many of the questions that a user will ask, they are still not able to acquire the natural flow of how humans communicate. They cannot understand sarcasm, misspelling in the statements, and slang used by humans. The result of this is that chatbots cannot be used in public situations or a personal channel like social application. While this problem exists there has been improvement in chatbots to adapt how humans communicate.

Chatbots are not personalized and do not have emotions

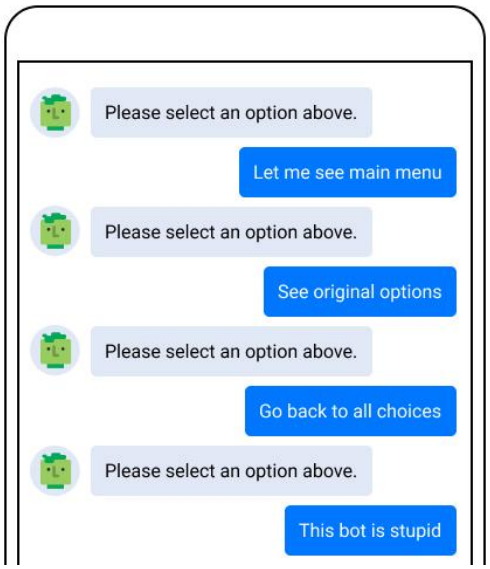
Chatbots also lack emotions and personalization, when a user has a problem they are looking for empathy, which is difficult for the chatbots to express. If the conversation between the chatbot and user is not able to follow a particular track, the chatbot is not able to improve the discussion, and it lacks the “human” element which results in an unhappy user and negative experience for the customer.

- Only answers general questions/simple questions
- Higher chances for misunderstanding if the question is outside of scope and/or the question is not clear

Limited Functionality

The goal of a chatbot is to provide a solution/output for simple questions that relies on a factual basis. Since the training of the chatbot is limited, it results in limited responses. For example, if there is a question with multiple parts or sub context that the AI model is not trained or is framed in a complex way then it fails to provide smooth functionality.

Hence, the chatbot sometimes does not work as expected and may require the user to re-frame the question or ask the multi part questions separately for the AI to grasp better. The response may or may not be as accurate. Therefore we notice that some chatbots tend to have limited functionality().

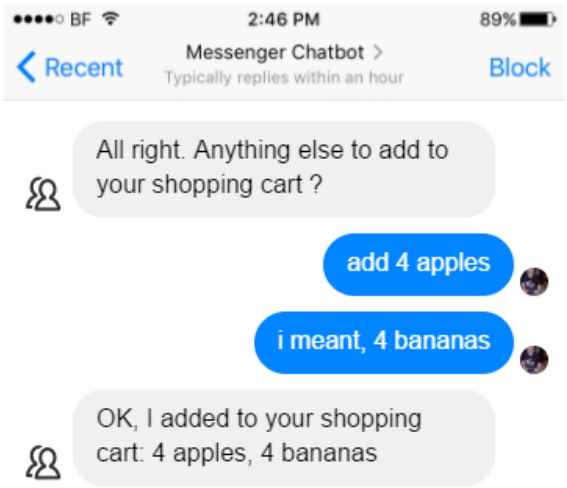


(Treml, 2017)

Higher chances for misinterpretation

In cases where the user is not providing responses in a manner that the AI expects or understands, then the bot may have higher chances to misunderstand what the user is trying to input. This happens in many cases where the responses from the user's end are not very clear or in cases that need correction/ alteration to be made. If the system database does not have the answer to the customer's questions, it may respond with whatever context the bot relates the question most appropriate to. This provides a bad experience to the user and can lead to frustration and negative impact of the bot creator ().

Here is a very good example:



(Treml, 2017)

Project Objective (C) :

The objective of our project is to build a chat-bot that will be able to hold a two way conversation with the user. It performs basic functions like answering basic questions, performing tasks like fetching data/information from the internet, playing audio/video on platforms like youtube, calculating basic algorithms, etc. The goal is to make a bot which is able to hold a conversation with a live user. We want to build a chatbot that is intelligent and more lively than a regular chatbot, that we might interact with when we are visiting a website and/or looking at the application.We will be applying python modules/libraries to achieve this. We will also be accomplishing our goal through neural network strategies and algorithms, specifically recurrent neural networks (RNN) and its extension such as long short-term memory LSTM and convolutional neural network CNN. We also believe that this project will give all members a hands-on and extended exposure into the world of AI, and how neural networks operate.

Approaches or methodologies investigated by each student (D)

Recurrent neural network (RNN) - Dhruv

RNN is a type of neural network which is a very powerful algorithm and it is applied by Google's voice search as well as Apple's Siri (Donges, 2021). It is also the only algorithm with internal memory. It is one of the algorithms behind the recent achievements in deep learning. The internal memory that is part of RNN allows them to remember important elements in the input. This allows the RNN to predict precisely for the next input. This is the reason why they are the

likely selected algorithm for data that is sequential, for example, speech, text, time series, audio, video, etc.

How does RNN work?

For us to understand RNN, we will need knowledge about sequential data and feed-forward neural networks.

- Sequential Data: It ordered data where elements that are related follow each other. A very popular example of sequential data is time series data, where data points are recorded in time order.
- Feed-Forward Neural Network: the data will move in one direction, from input zone -> hidden zone -> output zone. There is also no memory connected to the input which results in prediction that is not good. This type of neural network is not aware of time order, this is because it only focuses on the present input. It is not able to remember the past and only remembers its training.

In RNN, there is a loop connected to data. When a RNN makes a decision it will take into account the current input as well as the past inputs. A standard RNN will have short term memory, however Long short-term memory (LSTM) which is an extension of the RNN and when RNN combines with LSTM, RNN will have long term memory (Donges, 2021).

RNN will have two inputs, the present and the past close to present. This is critical because the sequential data will have information about what the future situation will be, this differentiates RNN from other algorithms. RNN just like the feed-forward neural network also applies weights to both current input and the input before the current input.

Let's go through a quick example of RNN memory with the word “neuron”. As the character of the words get processed one by one, RNN will remember the characters because of the memory characteristic of the RNN. Then it will create an output, copy that output, and loop the output back into the network.

Types of RNN:

1. One to One
2. One to Many
3. Many to One
4. Many to Many

RNN are able to map:

- One to Many
- Many to Many (Translation)

- Many to One (Voice)

Now that we have an overview of what RNN is, the question arises of why we are applying it to our chatbot? The answer to this question is to make our chatbot smarter. The notion behind RNN is that we pass the data which is the most important from our imputed text to the hidden layers, this will result in the network learning and evolving itself (Erko, 2022).

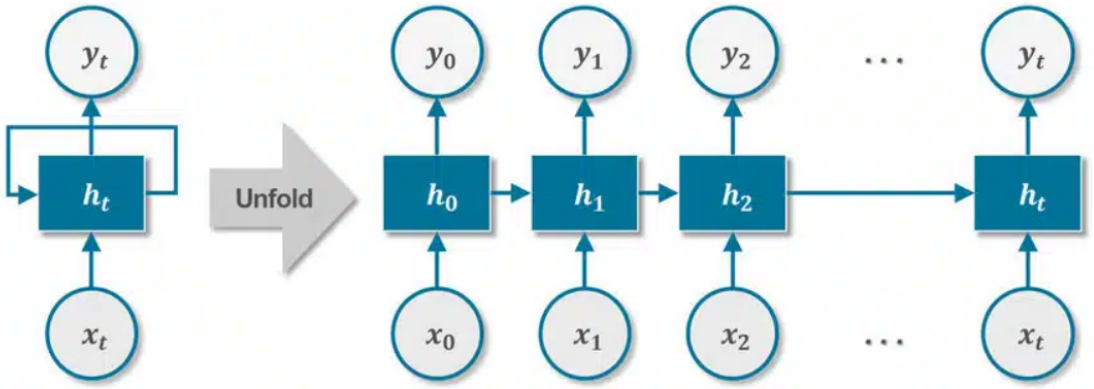


Figure 1. Recurrent neural network architecture

Recurrent neural network architecture. builtin. What Are Recurrent Neural Networks? | Built In

In the figure above, input x is passed to the hidden layer "h" and then looped several times, this looping allows the network to learn and evolve.

Going into more details with an example, suppose a user/client asks a chatbot "what is the current weather", we will break the sentence in to each word and input to the RNN, similar to the figure 1 where x0 will be word "what" and x1 will be word "is" and this continues for next word in the statement. Let's follow the steps:

1. We input the first word to the RNN which in this situation is "what" [Figure 1 - x0"]
2. Next the RNN will encode the word - [Figure 1 - "h0"]
3. Then a output will be created which is the prediction of the next word - [Figure 1 - "y0"]
4. Next we input the word "is" [Figure 1 - "x1"]
5. We can see that the hidden layers are connected to each other and the hidden layer for the second word will be combination of first hidden layer and the second input - [Figure 1 - "h1"]
6. Again our output will be the prediction of the next word [Figure 1 - "y1"]
7. All the steps are repeated for next words in the statement, but current hidden layer will be combination of the previous hidden layer and the current input

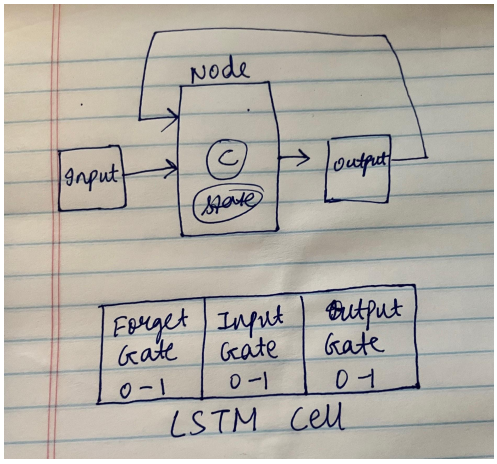
At the end of the statement our hidden layer [Figure 1 - "ht"] will have information from all the previous words and since all the hidden layers are connected, with this information a final output will be generated [Figure 1 - "yt"] (Erko, 2022).

Long short-term memory (LSTM) - Ria

Long short term memory allows a neural network to remember and store the information that it needs to keep the context but also to delete or discard the information or data that is no longer required and relevant (2021). It is very similar to the concept of RNN. A recurrent neural network is where an LSTM lives, so essentially LSTM is a type of a recurrent neural network.

For example, there is a node. This node receives some kind of input.This input is then processed in some way, so there is some kind of computation which results in an output. Although what makes a RNN node a little bit different is that it is recurrent which means it takes a loop around. Hence the output of a given step is provided alongside the input in the next step.Thus it allows RNN to remember the previous steps in a sequence

Although RNN does suffer from what is known as the long term dependency problem. It means that overtime as more and more information piles up, then RNN becomes less effective at learning new things and back tracking more and more to get context for the information to be processed. Here the concept of LSTM - Long short term memory comes handy as it provides the solution to this long term dependency problem. To add to this, we introduce what we call an internal state to the RNN node. Now when an RNN input comes in, it is receiving the internal state information as well. So a step receives the information from the previous step, the input of the new step and also some state information from the LSTM state (2021).



Each state is an LSTM cell and it consists of three parts. Each part has a gate.

- Forget gate
    - Keeps track of what sort of state information that is stored in this internal state can be forgotten.
- Input gate

    - The input gate keeps note of what new information can we add or update into this working storage state information.
- Output gate
    - The output gate keeps note of all the information that is stored in that state, which part of that should be output in this particular instance.

All these three states can be assigned numbers between 0 and 1 out of which 0 meaning that the gate is effectively closed and that nothing gets through whereas 1 means the gate is wide open and that everything can possibly go through.

Now when we are processing in our RNN cell, we have this additional state information that can provide us with some additional context. LSTM can be applied to any sort of series where we have a sequence prediction that is required and some long term dependency data to go alongside it (2021).
Some typical use cases:

- Machine translation
- Simple Q&A chatbot where we see the need for some information to be retrieved that we used in the previous step for us to recall it later on and make use of it as a context. For example: What is the weather in regina. How cold will it get tomorrow?
  Indicating that we are still talking about the weather aspect in the location of Regina.
  Hence LSTM comes handy where it potentially stores information like:
  Aspect: Weather
  Location: Regina.

Convolutional neural network (CNN) - Alish

CNN is mostly used for recognizing objects and image processing (International, 2021). However, it can also be used for detecting patterns within the sequence of data that is encoded into digits. CNN has multiple layers that process the data and extract features from them. CNN has convolution layer, Rectified Linear unit layer, pooling layer and fully connected layer. Convolutional layer performs filter, ReLU performs operations and the generated data is rectified feature map. Polling layer reduces the dimension from generated data in the ReLU layer and converts the 2d array to a single large linear vector (International, 2021). Finally, Fully connected layer gets the vector as input and identifies the data. Since CNN is very popular with image processing, the pixels in the images are segregated and that small section is processed separately using non linear processing and they are also connected (International, 2021).

Although in today's AI field there is very little difference between RNN and CNN, they are not exactly the same (International, 2021). The Major difference between RNN and CNN is, RNN is feed forward neural network and processes on sequential data. However, CNN is fully connected

or pooled and it can work on non linear processing. CNN also offers more filtering options and provides higher accuracy. RNN is useful for predicting data and reiterating on previous nodes to provide better results over time (International, 2021).

Implementation Details (E)

Uptil now, Ria Chevli and Dhruv Modi worked on the voice assistant- personal assistant part of the simple ChatBot. It usually includes the usage of various packages using the pip command. Few examples of the packages we used are: wikipedia, pyttsx3, datetime, time,json, request etc. We use the “Sapi5” microsoft text to speech engine for the voice recognition.

Dhruv Modi worked on -

- Fetching data from wikipedia
- Accessing the web browsers
- AI answering geographical and computational question
- To log off the PC

Ria Chevli worked on -

- Predicting current time and date
- Searching data from the web
- Questions and extra features
- Weather forecast (Dhruv helped with this too)

Alish has worked on with help of Ria and Dhruv -

- Training model using tensorflow
- Filtering out the words in our intent json file
- Sorting those sentences into categories like tags, pattern and response
- Adding words that are inside the tags and saving response and tags into training model (tags, words)
- Building Deep Neural Network model using tensorflow and fully connecting them with training data of tags([0]) and words in pattern([1])
- Training the model using above variables for at least 1000 variations or epoch and batch size of processed sample of 8 before the model is updated.
- Saving the model into training_data
- Creating functions to tokenize sentences, returning words from our model within the matching words in sentence, classifying and predicting the tags based on probabilities from the input sentence and return list of tags, and generating and returning random response from the filtered out data in the classify function.
- Adding threshold limit for the data that is classified to return tags. Current threshold limit is 0.2, which means any sentence that does not return an accuracy of 80% (0.8 probability) or more would not be recommended by our chatbot.

- We have added a simple default route in case the probability is less than 80% and it will ask us to search on google for response just like Google voice assistant does.

From our approaches and methodologies that we researched, we have applied RNN [Recurrent Neural Network] and CNN [Convolution Neural Network] to our chatbot. We have not implemented the LSTM [Long short-term memory] approach/methodology into our chatbot instead we are applying temporary data storage for one iteration. We have also researched and applied the ADAM optimizer by tensor flow.

Results (F)

We have achieved 80% of our proposed goal. Our bot is able to respond to issues related to google questions, search on wikipedia, and perform OS based actions. In our Project proposal, where we had some issues with response credibility. We have resolved that issue by adding additional parameters from different functions in tensor flow which gives us accuracy for the result. Based on that result accuracy, we have added the default route in case the accuracy of the result found is less than 70%. Just like any other Voice assistant like Google or Siri, it asks for searching on the internet for anything it doesn't find anything related to in our database. Based on our second response, it will either iterate through the beginning or it will search on the internet based on our *"Yes"* response. We were not able to implement 2 ways conversation and it did not save any data based on user response. We have switched different models to provide us with better accuracy and it took some research time.

Conclusion / Future Work (G)

On personal levels we believe that this project enhanced our coding skills in python as well as we have gained knowledge about Artificial Intelligence. It helped us go out of our bubbles and explore more, research about the concepts related to the tensor flow, voice recognizer module and getting better insights on how neural networks works and passing in our data to train the model. Overall, this class provided us the opportunity to do projects based on real world concepts.

For future work we have decided that we would develop a GUI for our Chatbot to enhance the user interactions. We also would also implement a two way conversation to make discussion between our user and chatbot more interactive and natural.

References (H)

(2021). *YouTube*. Retrieved December 3, 2022, from https://youtu.be/b61DPVFX03I. https://youtu.be/b61DPVFX03I

Funniest chatbot fails & bad Chatbot Responces. Boxmode Blog. (2021, September 22). Retrieved December 3, 2022, from https://blog.boxmode.com/funniest-chatbot-fails/ https://blog.boxmode.com/funniest-chatbot-fails/

Treml, F. (2017, April 24). *3 things your chatbot fails at (but shouldn't)*. Medium. Retrieved December 3, 2022, from https://chatbotsmagazine.com/3-things-your-chatbot-fails-at-but-shouldnt-1be87e81f6f

M.Mirthula. (2020, July 30). *How to build your own AI personal assistant using python*. Medium. Retrieved December 3, 2022, from https://towardsdatascience.com/how-to-build-your-own-ai-personal-assistant-using-python-f57247b4494b

Sykes, D. (2018). *Tensorflow chat bot!!* replit. Retrieved December 2, 2022, from https://replit.com/talk/learn/Tensorflow-chat-bot/8342

International, T. (Ed.). (2021, January 21). *What's the difference between CNN and RNN?* What's the Difference Between CNN and RNN? | TELUS International. Retrieved December 2, 2022, from https://www.telusinternational.com/articles/difference-between-cnn-and-rnn

Donges, N. (2019). Recurrent neural networks 101: Understanding the basics of RNNs and LSTM. Built In. https://builtin.com/data-science/recurrent-neural-networks-and-lstm

Kostadinov, S. (2019, November 10). How Recurrent Neural Networks work - Towards Data Science. Medium. https://towardsdatascience.com/learn-how-recurrent-neural-networks-work-84e975feaaf7

Viraj, A. (2021, December 16). How To Build Your Own Chatbot Using Deep Learning - Towards Data Science. Medium. https://towardsdatascience.com/how-to-build-your-own-chatbot-using-deep-learning-bb41f970e281

Biswal, A. (2022, November 30). Recurrent neural network (RNN) tutorial: Types and examples [updated]: Simplilearn. Simplilearn.com. Retrieved December 3, 2022, from https://www.simplilearn.com/tutorials/deep-learning-tutorial/rnn#:~:text=RNN%20works%20on%20the%20principle,the%20output%20of%20the%20layer.

Breaking down the pros and cons of Chatbots as a customer experience solution. RSS. (n.d.). Retrieved December 3, 2022, from https://www.chatdesk.com/blog/pros-and-cons-of-chatbots

YouTube. (2018). YouTube. Retrieved December 3, 2022, from https://www.youtube.com/watch?v=LHXXI4-IEns&ab_channel=TheA.I.Hacker-MichaelPhi.

YouTube. (2022). YouTube. Retrieved December 3, 2022, from https://www.youtube.com/watch?v=DFZ1UA7-fxY&ab_channel=ritvikmath.