A Smart AI Chatbot using Python and Machine Learning

Doctor Chat Bot on Coronavirus

Table of Content

Abstract: *This is a project-based term paper. This paper aims to present a design of 'Doctor Chat Bot on Coronavirus' that if I ask this chat bot about corona disease, it will come up with a reasonable response.*

Keywords: Chatbot, Coronavirus, AI, Python

Background of this project: Coronavirus disease (COVID-19) is an infectious disease caused by a newly discovered coronavirus.[1] Currently there's no cure for the 2019-nCOV virus. Recently many fake news came out. So, people get confused about coronavirus. That's why I thought to make a chatbot that would give them an accurate idea of the Coronavirus. This type of chatbot can used in any health-related websites.

1. Introduction

A Chatbot is a system that can interact with human users with natural language. The vast amount of information that is available on the internet allows Chatbots to provide accurate and efficient information based on the user's requirements. Chatbots are used in domains like Customer Support, Virtual Assistance, Online Trainings, and Online Reservations and also for general conversations. Many chat bots are created to simulate how a human would behave as a conversational partner.[2] Chat bots are in many devices, for example Siri, Cortona, Alexa, and Google Assistant. Many chat bots are used now a days for customer service.

There are broadly two variants of chat bots: **Rule-Based** and **Self-Learning**. A Rule-Based chat bot is a bot that answers questions based on some rules that it is trained on, while a Self-Learning chat bot is a chat bot that uses some Machine Learning based technique to chat.[2]

I have used a rule-based approach for responding back to greetings, and I have the chat bot respond to questions and queries by taking in some text and having the chat bot select the best response back from that text. This type of self-learning is called **retrieval-based** learning.

Recently I have learned Python using Google Colab. So, I have decided to make a chatbot in google colab. My proposed doctor chatbot read in the URL of an article then answer to the users based on that article.

2. Related Work

In this section Artificial Intelligence technologies that have been used in the development of chatbots are briefly presented. Also, this section includes a short review on the chatbot applications that have been implemented for Medical Bots or Doctor Bots.

2.1. Artificial Intelligence technologies and chatbots

Chatbots are rapidly evolving AI applications used by several companies and organizations all over the world in order to engage and serve the users' needs by exploiting the most common skill of them, which is chatting. A short selected (from the literature) definition of a chatbot is: "Chatbot (or chatterbot, Bot, smartbot, Conversational interface or Artificial Conversational Entity) is a computer program or an artificial intelligence which conducts a conversation via auditory or textual methods" (Bilange, 1991; Vassos, Malliaraki, Falco, Maggio, Massimetti, Nocentini & Testa, 2016; Følstad & Brandtzaeg, 2017; Valtolina, Barricelli, Gaetano & Diliberto, 2018). Their major advantage is that they can serve their users anytime and if they get smart enough, they can provide a satisfactory human-like assistance. Chatbot applications are using advanced AI technologies, as presented in the following subsections.[6]

2.1.1. Machine Learning

Machine Learning is the field of AI science that focuses on getting machines to "learn" and to continually develop autonomously. ML utilizes supervised or unsupervised algorithms, such as decision trees, neural networks, deep learning and others, that enable computer systems to optimize the ways of problem solving (Sukhbaatar, Weston & Fergus, 2015; Serban, Sordoni, Bengio, Courville & Pineau, 2016, Wehle, 2017). For chatbots, ML is used to understand what users are saying, learning from examples provided for training the system on what a user might say when interacting with the chatbot, analyzing and understanding user intent in real-time.
Many Doctor Chatbot designs have been proposed in the past few years which aim to provide the user with medicine recommendation after extracting the illness information from the user messages. [6]

2.2. AI Chatbot Platforms

Before using a readymade chatbot, a developer must define the reasons and the goals that the chatbot is designed for. There are mainly three types of chatbot platforms: No-programming platforms, conversation-oriented platforms, and platforms backed by the industry (Couto, 2017).

1. The no-programming platforms are simple task-oriented platforms that do not have NLP or ML capabilities. There are easy in use and in construction of chatbots and can answer to simple questions concerning simple tasks such as order a pizza or buy a ticket. The generated chatbots can be easily integrated in social media and webpages. The most famous platforms are Chatfuel (2019), Octane.ai (2019) and Motion.ai (2019). [5]
2. The conversation-oriented platforms on the other hand are not built to serve a task but are designed to make a conversation. Most of them use the Artificial Intelligence Modelling Language (AIML) (2019) in order to model the conversation with a user and manage to conduct good discussions. The don't use NLP or ML techniques so the developers must keep them manually updated. The most famous conversational bots' platform is PandoraBots (2019). [5]

3. The platforms backed by the industry are advanced chatbot platforms that are using ML and/or NLP technology, and may integrate large external sources of data/information/knowledge such as Web pages, ontologies, and Knowledge Graphs. The most famous of these platforms are Google Dialogflow (2019), Facebook Wit.Ai. (2019), KITT.AI (2019), IBM Watson Assistant (2019), Microsoft Luis (2019) and Amazon Lex (2019). [5]

2.3. Doctor Chatbot or Medical Chatbot

Many Doctor Chatbot designs have been proposed in the past few years which aim to provide the user with medicine recommendation after extracting the illness information from the user messages.
A similar paper "Pharmabot: A Pediatric Generic Medicine Consultant Chatbot" proposed by Benilda Eleonor V. Comendador, Bien Michael B. Francisco, Jefferson S. Medenilla, Sharleen Mae T. Nacion, and Timothy Bryle E. Serac provides a design for a stand-alone medical Chatbot that is implemented using MS Access and Visual C# [3]. For using the proposed design, the user has to navigate using the four options provided by the application. This design aims to work by converting the user input to SQL queries and execute it on MS Access to retrieve the solution to the illness [3].
Also, a research paper "MedChatBot: An UMLS based Chatbot for Medical Students" proposed by Hameedullah Kazi, B.S. Chowdhry and Zeesha Memon focuses on a design for an AIML based Medical Chatbot. This Chatbot design is implemented using a JAVA based AIML interpreter called Chatter bean [4]. To use the proposed design, the user has to type a message that should contain the illness name and it detects the illness names using AIML patterns. Once the illness is detected, the Chatbot provides the user about the necessary information about the problem [4].

However, the previous proposed designs in the past did not focus in understanding the intensity of the illness that the user is suffering through. But their project was more advanced and powerful than mine.

3. Proposed Work

My proposed work was on a smart AI chatbot using Python and machine learning. And the chatbot is a Doctor chatbot on coronavirus that that if I ask this chat bot about corona disease, it will come up with a reasonable response. I have implemented the chatbot in Google Colab. **Google Colab** is a free cloud service and now it supports free GPU! You can: improve your Python programming language coding skills. develop deep learning applications using popular libraries such as Keras, TensorFlow, PyTorch, and OpenCV.[7]

3.1. Natural Language Processing Vocabulary

When it comes to Natural Language Processing (NLP), it will come across some terms. I have used NLP throughout the code. A few of those terms and definitions are below:

1. **Natural language processing (NLP)**

   Natural language processing (NLP) is a sub field of linguistics, computer science, information engineering, and artificial intelligence concerned with the interactions between computers and human (natural) languages.[9]

2. **Text Corpus**

   A corpus or text corpus is a large and structured set of texts of a particular author or a body of writing on a particular subject.

3. **Bag of Words (BoW)**

   Bag of Words (BoW) is a Natural Language Processing technique of text modeling. It is called Bag of Words because any information about the order or structure of the word document is removed and the model is only worried about the frequency of the known words in the document.

   Consider the document:
   "It was the best of times"
   "It was the worst of times"
   "It was the age of wisdom"
   "It was the age of foolishness"
   The dictionary contains the words:
   { *'It', 'was', 'the', 'best', 'of', 'times', 'worst', 'age', 'wisdom', 'foolishness'*
   If we want to vectorize the text *"It was the best of times"*, we would have the following vector: **[1, 1, 1, 1, 1, 1,0, 0, 0,0]**.
   The frequency of the words from the 10 unique words in the dictionary are below for the text *"It was the best of times".*
   "it" = 1
   "was" = 1
   "the" = 1
   "best" = 1
   "of" = 1
   "times" = 1
   "worst" = 0
   "age" = 0
   "wisdom" = 0
   "foolishness" = 0 [8]

4. **Stemming:**

   Stemming is the process of reducing inflected words to their word stem, base or root form — generally written word form. For example, if we were to stem the word "dance", "dancing", "dances", the result would be the single word "dance"

5. Lemmatization:

Lemmatization is the process of grouping together the different inflected forms of a word so they can be analysed as a single item and is a variation of stemming. For example, "feet" and "foot" are both recognized as "foot".

6. Tokenize:

One common task in **NLP** (**Natural Language Processing**) is **tokenization**. "Tokens" are usually individual words (at least in languages like English) and "**tokenization**" is taking a text or set of text and breaking it up into its individual words or sentences.

7. CountVectorizer:

CountVectorizer works on Terms Frequency, i.e. counting the occurrences of tokens and building a sparse matrix of documents x tokens.

8. TF-IDF Vectorizer:

TF-IDF stands for term frequency-inverse document frequency. TF-IDF weight is a statistical measure used to evaluate how important a word is to a document in a collection or corpus. The importance increases proportionally to the number of times a word appears in the document but is offset by the frequency of the word in the corpus.

- **Term Frequency (TF)**: is a scoring of the frequency of the word in the current document or another way of saying it measures how frequently a term occurs in a document. Since every document is different in length, it is possible that a term would appear much more times in long documents than shorter ones. The term frequency is often divided by the document length to normalize.

$$TF(t) = \frac{Number\ of\ times\ term\ t\ appears\ in\ a\ document}{Total\ number\ of\ terms\ in\ the\ document}$$

- **Inverse Document Frequency (IDF)**: is a scoring of how rare the word is across documents or another way of saying it measures how important a term is. IDF is a measure of how rare a term is. Rarer the term, more is the IDF score.

$$IDF(t) = log_e(\frac{Total\ number\ of\ documents}{Number\ of\ documents\ with\ term\ t\ in\ it})$$

Thus,

$$TF - IDF score = TF * IDF$$

TF-IDF is thus the product of TF and IDF or TF * IDF.

3.2. Chatbot Design Programming

This program takes text from an online website [10] and uses it to chat and answer queries. I have created a 'smart' chat bot program to answer queries on Coronavirus.

```
# Description: This is a 'smart' chatbot program
```

I install a few packages **nltk** and **newspaper3k**. NLTK is the Natural Language Tool Kit package, which is a popular package for NLP with Python. Newspaper3k is a python package used for extracting and parsing newspaper articles.

```
pip install nltk
pip install newspaper3k
```

3.2.1. Import the Libraries & Packages

Next, I import the libraries. I use the **newspaper** library to extract the text from the website by using the **Article** class. I use the **random** library to generate a random number for our greeting response. I the **string** library to process the standard Python string. The **sklearn.feature_extraction.text** library I used to get the count vectorizer class **countVectorizer** to vectorize the text and evaluate how important a word is to a document. From the **sklearn.metrics.pairwise** library I got the **cosine_similarity** method to see how similar the text is to the users queries. I have also import the **numpy** library to use some of its methods like the sort() method. Last but not least I use the warnings library to ignore the **warnings** I get with this program.

```
#import libraries
from newspaper import Article
import random
import string
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics.pairwise import cosine_similarity
import nltk
import numpy as np
import warnings
warnings.filterwarnings('ignore')
```

Downloaded the **punkt** package. Punkt is a pre-trained tokenizer model for the English language that divides the text into a list of sentences. I install the **popular** package which is a subset of NLTK data, or all of the packages using **nltk.download()**

```
nltk.download('punkt', quiet=True) # Download the punkt package
```

3.2.2. Download the Data from a Website

Read in the URL [10] of the article to get the text corpus. A text corpus or corpus is a large and structured set of texts of a particular author or a body of writing on a particular subject.

Downloaded the article, then parse the article and apply NLP to the article. Then stored the articles text into a variable **corpus**.

```
#Get the article URL
article = Article('https://medicalaid.org/coronavirus-what-you-need-
to-
know/?gclid=Cj0KCQjw7qn1BRDqARIsAKMbHDYc0mbVeyDAJEizV5WPqZ39sVH1kR8iMN
wCj-B3iSajJT1gO6LLgQUaAgk3EALw_wcB')
article.download() #Download the article
article.parse() #Parse the article
article.nlp() #Apply NLP
corpus = article.text #Store the article text into corpus
```

Print the corpus

```
print(corpus)
```

```
The 2019 novel coronavirus has the potential to be a global pandemic.
Health officials say it originated in a market in Wuhan, China that sold
live and dead wild animals that people ate for food, improved health and
vitality and a number of other purposes. The virus has now been detected
in Australia, Canada, Finland, France, India, Italy, Japan, Nepal,
Russia, Singapore, Spain, Taiwan, Thailand, Vietnam, the United States
and over a dozen other countries.
```

A sample of printed text/corpus

3.2.3. Tokenize the Data

Next, I have **tokenize** the text by getting a list of sentences from the text.

```
#Tokenization
text = corpus
sent_tokens = nltk.sent_tokenize(text)# txt to a list of sentences
```

Print the list of sentences.

```
#Print the list of sentences
print(sent_tokens)
```

```
['The 2019 novel coronavirus has the potential to be a global pandemic.',
```

Sample of the printed sentence token

Next, I use keyword matching (a rule-based approach) to check for greeting type words as input from the user and respond back with a randomized greeting as output.

To do this I create a list of greeting inputs (the greetings we expect from the user) and then I create a list of greeting responses (the greetings that our chat bot will use).

Then I create a function to check for the user's greetings and randomly choose a greeting response back.

```
#Function to return a random greeting response to a users greeting
def greeting_response(text):
  #Convert the text to be all lowercase
  text = text.lower()
  # Keyword Matching
  #Greeting responses back to the user from the bot
  bot_greetings = ["howdy","hi", "hey", "what's good",  "hello","hey
there"]
  #Greeting input from the user
  user_greetings = ["hi", "hello",  "hola", "greetings",
"wassup","hey"]

  #If user's input is a greeting, return a randomly chosen greeting
response
  for word in text.split():
    if word in user_greetings:
      return random.choice(bot_greetings)
```

Create a function to return the indices of the values from an array in sorted order by the arrays values. This function will help return the chat bot response.

```
#Return the indices of the values from an array in sorted order by
the values
def index_sort(list_var):
  length = len(list_var)
  list_index = list(range(0, length))

x  = list_var
  for i in range(length):
    for j in range(length):
      if x[list_index[i]] > x[list_index[j]]:
        temp = list_index[i]
        list_index[i] = list_index[j]
        list_index[j] = temp

return list_index
```

3.2.4. Generating the Chatbot response

I have created a function which will take in a users response or queries, and then send back the best response(s) selected from the corpus.

```
# Generate the response
def bot_response(user_input):
    user_input = user_input.lower() #Convert the users input to all
lowercase letters
    sentence_list.append(user_input) #Append the users response to
the list of sentence tokens
    bot_response='' #Create an empty response for the bot
    cm = CountVectorizer().fit_transform(sentence_list) #Create the
count matrix
    similarity_scores = cosine_similarity(cm[-1], cm) #Get the
similarity scores to the users input
    flatten = similarity_scores.flatten() #Reduce the dimensionality
of the similarity scores
    index = index_sort(flatten) #Sort the index from
    index = index[1:] #Get all of the similarity scores except the
first (the query itself)
    response_flag=0 #Set a flag letting us know if the text contains
a similarity score greater than 0.5
    #Loop the through the index list and get the 'n' number of
sentences as the response
    j = 0
    for i in range(0, len(index)):
      if flatten[index[i]] > 0.5:
        bot_response = bot_response+' '+sentence_list[index[i]]
        response_flag = 1
        j = j+1
      if j > 2:
        break
    #if no sentence contains a similarity score greater than 0.5 then
print 'I apologize, I don't understand'
    if(response_flag==0):
        bot_response+' '+"I apologize, I don't understand."
```

3.2.5. Start the Conversation

Now I create a continuous loop for the chat bot to converse with the user. We will run this loop until the users response is 'exit'.

```
#Start the chat
print("Doc Bot: I am DOCTOR BOT or Doc Bot for short. I will answer
your queries about Chronic Kidney Disease. If you want to exit, type
Bye!")


exit_list = ['exit', 'see you later','bye', 'quit', 'break']


while(True):
    user_input = input()
    if(user_input.lower() in exit_list):
      print("Doc Bot: Chat with you later !")
      break
    else:
      if(greeting_response(user_input)!= None):
        print("Doc Bot: "+greeting_response(user_input))
      else:
        print("Doc Bot: "+bot_response(user_input))
```

```
DOCBot: I am Doctor Bot or DOCBot for short. I will answer your queries
about Coronavirus. If you want to exit, type Bye!
hi
DOCBot: hey
what are coronavirus?
DOCBot: The 2019 novel coronavirus has the potential to be a global
pandemic.
What is the symptoms of coronavirus?
DOCBot: Symptoms Of The Illness

People infected with the 2019 novel coronavirus begin to experience mild
cold or flu-like symptoms in two to four days.
How people can protect themselves from coronavirus?
DOCBot: How People Can Protect Themselves

According to the Centers for Disease Control and Prevention, the best way
people can prevent themselves from becoming infected with the 2019-nCoV
virus is to avoid exposure to people or animals sickened with it.
```

Sample of chatting with DOCBot.

4. Future Work

The Doctor Bot is the first step for the creating a AI chatbot. But it is not fully developed chatbot. Later I will implement this chatbot in Python GUI. And it canbe used in medical related websites.

5. Conclusion

My Doctor Chatbot will have a great impact on the life of its users. It would provide them the advantage of carrying a virtual Doctor in their pockets. It would also give them the freedom to chat with doctor and also can get a real doctor's advice if needed. This can be a most popular tool for people with busy schedule as they won't have to hamper their schedule to consult a doctor for minor health queries. This would also be a tool with high utility among elderly and physically disabled people as this can help them get solutions to all their health-related issue at their fingertips.

6. References

[1] https://www.who.int/health-topics/coronavirus#tab=tab_1

[2] https://medium.com/@randerson112358

[3] Comendador, B. E., Francisco, B. M., Medenilla, J. S., Nacion, S. M., & Serac, T. B. (2015). Pharmabot: A Pediatric Generic Medicine Consultant Chatbot. Journal of Automation and Control Engineering, 3(2), 137-140. doi:10.12720/joace.3.2.137-140

[4] Kazi, Hameedullah & S. Chowdhry, B & Memon, Zeesha. (2012). MedChatBot: An UMLS based Chatbot for Medical Students. International Journal of Computer Applications. 55. 1-5. 10.5120/8844-2886.

[5] Couto, J. (2017). Building *a Chatbot: Analysis and Limitations of Modern Platforms - DZone AI*. Retrieved from https://dzone.com/articles/building-a-chatbot-analysis-amp-%20limitations-of-mod

[6]https://www.researchgate.net/publication/318900216_Machine_Learning_Deep_Learning_and_AI_What's_the_Difference

[7] https://www.kdnuggets.com/2018/02/google-colab-free-gpu-tutorial-tensorflow-keras-pytorch.html

[8] https://medium.com/greyatom/an-introduction-to-bag-of-words-in-nlp-ac967d43b428

[9]https://en.wikipedia.org/wiki/Natural_language_processing#targetText=Natural%20language%20processing%20(NLP)%20is,amounts%20of%20natural%20language%20data.

[10] https://medicalaid.org/coronavirus-what-you-need-to-know/?gclid=Cj0KCQjw7qn1BRDqARIsAKMbHDYc0mbVeyDAJEizV5WPqZ39sVH1kR8iMNwCj-B3iSajJT1gO6LLgQUaAgk3EALw_wcB