Association for Computing Machinery
NITK Student Chapter

CSEA

Spider R&D Club

TRI - NIT

HACKATHON

29th January,2022

CodeZero

Varada Praveen

Bhogi Harshavardhan

Talagana Amarendra

ML01

During this pandemic, the healthcare industry has developed rapidly in order to combat covid in the best way possible. One can have several questions regarding Covid-19 and one's health which should be addressed quickly. However, doctors and healthcare workers have been working on the frontline, due to which, they cannot answer these questions for everyone. In order to tackle the problem, you have to design a chatbot that will answer all people's questions related to Covid-19.

Objectives:

The task is to design a chatbot using machine learning, specifically trained on Covid-19 data that can answer questions regarding Covid-19 accurately.

Solution:

We've written a python code for a chat bot which responds for basic Covid – 19 questionnaires based on the data we've already provided for the training.

Libraries used:

- Numpy
- Pandas
- Sklearn
- NLTK
- Metrics
- Warnings

Platform:

We've used google colab for the project as it has inbuilt libraries and installing new libraries is simple. The interface and environment of colab is intriguing and work friendly.

Code:

```
import pandas as pd
import numpy as np
import random
import nltk
import string
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics.pairwise import cosine_similarity
import warnings
warnings.filterwarnings('ignore')
```

The libraries are first installed and imported

```
from google.colab import drive
drive.mount('/content/drive')
data = pd.read_csv('/content/drive/MyDrive/Covid-Utterances-en_US.csv')
ans = pd.read_csv('/content/drive/MyDrive/ans.csv')
```

the data used for training is then imported from the csv. This is further stored in a series of lists or dataframes.

```
data.dropna(subset = ['1. covid_intro '], inplace=True)
list1 = data['1. covid_intro '].tolist()
data.dropna(subset = ['2. covid_origin'], inplace=True)
list2 = data['2. covid_origin'].tolist()
data.dropna(subset = ['3. symptoms_info'], inplace=True)
list3 = data['3. symptoms_info'].tolist()
data.dropna(subset = ['4. symtptoms_have'], inplace=True)
list4 = data['4. symtptoms_have'].tolist()
data.dropna(subset = ['5. emergency_contact'], inplace=True)
list5 = data['5. emergency_contact'].tolist()
data.dropna(subset = ['6. precautions'], inplace=True)
list6 = data['6. precautions'].tolist()
data.dropna(subset = ['7. current_numbers'], inplace=True)
list7 = data['7. current_numbers'].tolist()
data.dropna(subset = ['8. medication'], inplace=True)
list8 = data['8. medication'].tolist()
```

The columns in the data are saved in individual lists/dataframes. Additional empty elements and its rows are removed.

```
ans1 = ans['1'].tolist()
ans2 = ans['2'].tolist()
ans3 = ans['3'].tolist()
ans4 = ans['4'].tolist()
ans5 = ans['5'].tolist()
ans6 = ans['6'].tolist()
ans7 = ans['7'].tolist()
ans8 = ans['8'].tolist()
```

the Answers of chatbot through which the chatbot communicates are also divided into individual lists/dataframes, such that mapping will be easy

```
listkk = [list1,list2,list3,list4,list5,list6,list7,list8]
anskk = [ans1,ans2,ans3,ans4,ans5,ans6,ans7,ans8]
```

the lists are then saved in a two dimensional answer lists for easy mapping

```
def greetingresponse(text):
  text = text.lower()

  botgreetings = ['hii','hey','Howdy','Hello']
  usergreeting = ['hi','hii','hey','hello','wassup!']

  for word in text.split():
    if word in usergreeting:
      return random.choice(botgreetings)
```

the introductory part, i.e, the greetings part of the chat is then written. However, considering the time we have, we were not able to set more examples for greetings.

```
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
```

this is a sort function which will be used in the further code

```
def bot_response(user_input):
  user_input = user_input.lower()
  bot_response = ' '
  for k in range(8):
    listkk[k].append(user_input)
    cm  = CountVectorizer().fit_transform(listkk[k])
    similarity_scores = cosine_similarity(cm[-1],cm)
    similarity_scores_list = similarity_scores.flatten()
    index = index_sort(similarity_scores_list)
    index = index[1:]
    response_flag = 0


    j = 0
    for i in range(len(index)):
      if similarity_scores_list[index[i]] > 0.6:
        bot_response = bot_response + ' ' + random.choice(anskk[k])
        response_flag = 1
        j = j+1
      if j>3:
        break
    listkk[k].remove(user_input)
    np.delete(similarity_scores_list, -1)
    if response_flag == 1:
      break


  if response_flag == 0:
    bot_response = bot_response +' '+ "I apologise,I don't Understand"

  return bot_response
```

This is the main function which extracts the answers from chatbot according to the type or domain of question the user asks.

- The function first takes the user input as an argument and then appends it to each element of the questionnaire list(which is essentially the concatenation of all the other group of lists) in a loop.
- The count vectorizer then uses fit transform to encode it for the machine to understand and learn and to find similarities with the other questions in the list.
- The similarity scores are then formulated using cosine similarity of metrics library
- The similarity scores are ranged from 0 to 1 in which the exact user input gives us a score of 1
- Then the user input is removed from the lis and flag is set to zero

- According to the similarity scores obtained, the answer from bot is taken from the answer list of the same column. The index numbers are sorted according to the similarity scores thus the answer will be obtained easily.
- If the answer is found flag is set to 1 and the loop breaks
- If the answer is not found the flag is zero and chat bot answers default.

```
exit_list = ['exit','bye','ttyl','cultr']

while(True):
  userinput = input()
  if userinput.lower() in exit_list:
    break
  else:
    if greetingresponse(userinput) != None:
      print("Bot: "+greetingresponse(userinput))
    else:
      print("Bot: "+ bot_response(userinput))
```

this is the final part of code and execution part

one of the chat went on like this.

```
hii
Bot: Howdy
emergency?
Bot:   nan nan you can find the details here: don?t panic, this is something most of the people experiencing nowadays
what is the emergency?
Bot:   Coronaviruses are a large family of viruses that are known to cause illness ranging from the common cold to more severe diseases such as Middle East Respiratory Syndrome (MERS)
how can we avoid it?
Bot:   Wash your hands thoroughly with soap and warm water or with an alcohol-based hand sanitizer
Take hot shower or steamy shower to relieve sore, scratchy throat and cough
Keep your hands and fingers away from your eyes, nose, and mouth
Setup humidifier in your home
Drink plenty of fluids
bye
```

The code is pasted in the repository along with the pdf and dataset
The answer dataset is not found anywhere online so we prepared it.

Repository - https://github.com/AmarendraTalagana/CodeZero

Questionnaire – https://drive.google.com/file/d/1I_fHtFSv9GzVYJNu_KY0VxrB0t7B3YdB/view?usp=sharing
Answer csv - https://drive.google.com/file/d/1CWOexfGuQJSlbKB0K_HoG4gfhpXlUK8H/view?usp=sharing
Code - https://colab.research.google.com/drive/1u4ZdDJYFFWmnnmEFsZYOlC-2cZWkaD5t?usp=sharing