**Raspberry Pi Chat Bot**

Program to listen for a wake word then ask for a question which is sent to Openai chatGPT and speaks the response.

Program uses

1. picovoice for the wake word
2. Google TTS for the speech
3. Openai for the chat bot

**Hardware**

* Raspberry Pi 4
* Speakers or headphones
* USB microphone

**Install:**

* pip install openai
* sudo apt install python3 python3-pip python3-all-dev python3-pyaudio portaudio19-dev libsndfile1
* sudo pip3 install pvporcupine
* pip install gTTS
* pip install playsound

**Access keys:**

You will need access keys from:

picovoice: https://picovoice.ai/ (only if you want to use a wake word)

openai: . You need to sign up to it. The best place to start us:

https://platform.openai.com/docs/introduction/overview

**Setup tests:**

1. Headphones / Speakers

Plug your speakers or headphones into the 3.5 mm jack socket

Install the playsound if you haven’t done so.

Find a wav or mp3 file and play it. (If you don’t have an mp3 file, download one or wait for the next section). Try the python code:

from playsound import playsound

playsound('/path\_to\_your\_file.wav')

1. Microphone

I used a USB microphone. This is used for the wakeword and speech recognition.

Recording uses pyaudio. Make sure it’s installed as above.

A quick way to test that the microphone works is to find out the microphone’s hardware address with the following code:

import pyaudio

p = pyaudio.PyAudio()

for i in range(p.get\_device\_count()):

print(p.get\_device\_info\_by\_index(i).get('name'))

I get:

bcm2835 Headphones: - (hw:0,0)

USB PnP Sound Device: Audio (**hw:1,0**)

Now try a recording using the hw address and the command:

arecord -D **hw:1,0** -d 5 -f cd test.wav -c 1

This records for 5 seconds and saves to text.wav. Play back the test.wav file (see section 1). If it works you can ignore any jack server warnings.

1. Wakeword

Test the wakeword is detected. Install pvporcupine and obtain key from https://picovoice.ai/

1. Check detection - run the following:

**porcupine\_demo\_mic --access\_key "YOUR\_KEY" --keywords picovoice**

Say ‘picovoice’ and you should see: *Detected picovoice*

You can show the available wakeword using the command:

pvporcupine.KEYWORDS

They are:

'grasshopper', 'alexa', 'ok google', 'jarvis', 'terminator', 'porcupine', 'picovoice', 'computer', 'americano', 'grapefruit', 'hey barista', 'blueberry', 'hey google', 'hey siri', 'bumblebee', 'pico clock'

We’ll use ‘blueberry’, but you can change that in your code. You can now test wakeword detection using the code:

import waitForWakeWord

success = waitForWakeWord.wait()

print(success)

Make sure waitFOrWakeWord.py is in the same folder.

If success is True, the wakeword has been detected.

1. Make your Pi speak

Google Text to Speak a pleasant voice (if a bit slow). It saves the audio to a file, which then has to be read. Create three functions:

1. Save text to a file

from gtts import gTTS

def speak(text, filename):

tts = gTTS(text=text, lang="en")

tts.save(filename)

1. Play the file

import playsound

def play(filename):

playsound.playsound(filename)

and one to call them:

def Gspeak(speech, filename):

speak(speech, filename)

play(filename)

Test them using:

speech = "Waiting for the wake word - blueberry"

filename = "voice.mp3"

Gspeak(speech, filename)

You should hear “Waiting fro the wakewprd ‘blueberry’”

1. Call openai

Test the openai call using the following code:

import openai

openai.api\_key = ‘YOUR KEY HERE’

def openai\_create(prompt):

response = openai.Completion.create(

model="text-davinci-003",

prompt=prompt,

temperature=0.9,

max\_tokens=200

)

return response.choices[0].text

query = "how do i get to London"

response = openai\_create(query)

print (response)

You should get some information on getting to London.

1. Bring it all together:

Here’s all the code you need. Save keys.py, waitforwakeword.py and chatbot.py in the same folder and run chatbot.py

Full Code

**keys.py:**

key = {

'OPEN\_AI\_KEY' : ‘YOUR\_OPENAI\_KEY,

'PORCUPINE\_KEY' : ‘YOUR\_PVPORCUPINE\_KEY'

}

**waitForWakeword.py**

import keys

def wait():

import struct

import pyaudio

import pvporcupine

success = False

porcupine = None

pa = None

audio\_stream = None

access\_key = keys.key['PORCUPINE\_KEY']

try:

porcupine = pvporcupine.create(access\_key=access\_key,keywords=["blueberry"])

pa = pyaudio.PyAudio()

audio\_stream = pa.open(

rate=porcupine.sample\_rate,

channels=1,

format=pyaudio.paInt16,

input=True,

frames\_per\_buffer=porcupine.frame\_length)

while not success:

pcm = audio\_stream.read(porcupine.frame\_length)

pcm = struct.unpack\_from("h" \* porcupine.frame\_length, pcm)

keyword\_index = porcupine.process(pcm)

if keyword\_index >= 0:

success = True

finally:

if porcupine is not None:

porcupine.delete()

if audio\_stream is not None:

audio\_stream.close()

if pa is not None:

pa.terminate()

return (success)

**chatbot.py**

# 1. detect wake word,

# 2. prompt for question,

# 3. pass query to OpenAi and

# 4. speak response

import keys

import waitForWakeWord

import SpeechRec

import callOpenai

import openai

from gtts import gTTS

import playsound

def speak(text, filename):

tts = gTTS(text=text, lang="en")

tts.save(filename)

def play(filename):

playsound.playsound(filename)

openai.api\_key = keys.key['OPEN\_AI\_KEY']

filename = "query.mp3"

def Gspeak(speech, filename):

speak(speech, filename)

play(filename)

return

speech = "Waiting for the wake word - blueberry"

Gspeak(speech, filename)

success = waitForWakeWord.wait()

while success:

Gspeak("Ask me a question or say quit", filename)

query = SpeechRec.recognise()

if query != "quit":

Gspeak("I think you said " + str(query) + ". Asking chat g p t", filename)

response = callOpenai.openai\_create(query)

Gspeak("The answer is", filename)

Gspeak(response, filename)

else:

success = False

Gspeak("goodbye", filename)

Now ask chat GPT a question. See my video at:

I hope it all works for you