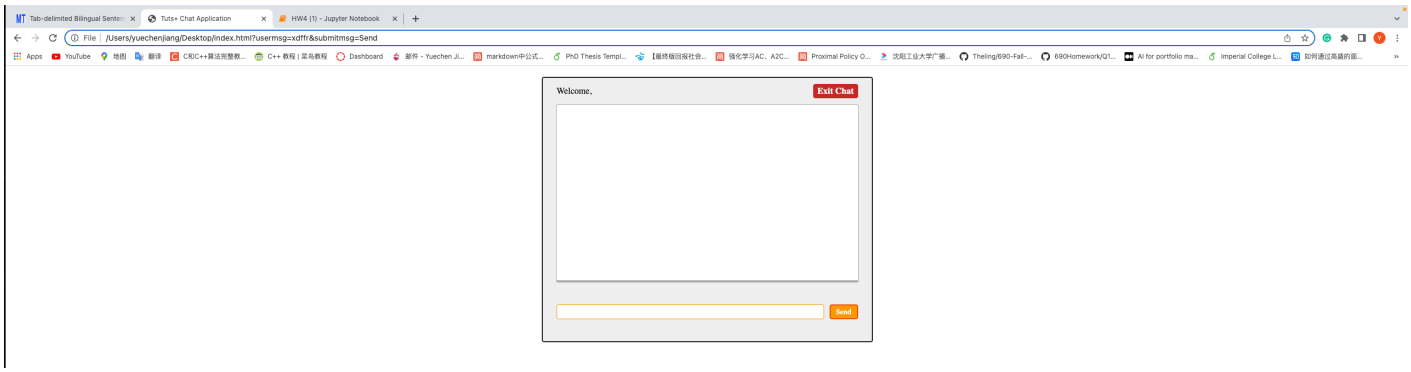
In this assisment I use IBM Watson Studio to build following 3 models and application:

- Convert Voice to Text
- Tanslate English to other language(Bi-LSTM Seq2Seq model Cognitive Application)
- Convert text to voice(Cognitive Application)

In this project, we will build a chatbox that can chat in multiple languages in real time. First, we will use SpeechRecognition to convert English speech into text, enter the chatbox, and then use the data provided by http://www.manythings.org/anki/ (http://www.manythings.org/anki/) , train a Bi-LSTM Seq2Seq model to translate the English output from the chatbox into Target Language, and finally convert the translated text into speech.

Preview of Chatbox



Convert voice to text

This will hear from your mic for 5 seconds, then try to convert that speech to text!

It's very similar to the previous code, but here we use the Microphone() object to read the audio from the default microphone, then use the duration parameter in the record() function to stop reading after 5 seconds, then upload the audio data to Google to get the output text.

You can also use the offset parameter in the record() function to start recording after an offset of a few seconds.

Also, you can identify different languages by passing the language parameter to the accept_google() function. For example, if you wanted to recognize Spanish speech, you could use:

In [ ]:
```
with sr.Microphone() as source:

# read the audio data from the default microphone

audio_data = r.record(source, duration=5)

# print("Recognizing…")

# convert speech to text

text = r.recognize_google(audio_data)

# print(text)


text = r.recognize_google(audio_data, language="es-ES")
```

Convert text to voice

This is a simple task using package to conver text to voice

In [ ]:
```
!pip3 install gTTS pyttsx3 playsound
```

```
Collecting gTTS
  Downloading gTTS-2.2.4-py3-none-any.whl (26 kB)
Collecting pyttsx3
  Downloading pyttsx3-2.90-py3-none-any.whl (39 kB)
Collecting playsound
  Downloading playsound-1.3.0.tar.gz (7.7 kB)
Requirement already satisfied: click in /opt/conda/envs/Python-3.9/lib/
python3.9/site-packages (from gTTS) (8.0.3)
Requirement already satisfied: requests in /opt/conda/envs/Python-3.9/l
ib/python3.9/site-packages (from gTTS) (2.26.0)
Requirement already satisfied: six in /opt/conda/envs/Python-3.9/lib/py
thon3.9/site-packages (from gTTS) (1.15.0)
Requirement already satisfied: charset-normalizer~=2.0.0 in /opt/conda/
envs/Python-3.9/lib/python3.9/site-packages (from requests->gTTS) (2.0.
4)
Requirement already satisfied: idna<4,>=2.5 in /opt/conda/envs/Python-
3.9/lib/python3.9/site-packages (from requests->gTTS) (3.3)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /opt/conda/env
s/Python-3.9/lib/python3.9/site-packages (from requests->gTTS) (1.26.7)
Requirement already satisfied: certifi>=2017.4.17 in /opt/conda/envs/Py
thon-3.9/lib/python3.9/site-packages (from requests->gTTS) (2021.10.8)
Building wheels for collected packages: playsound
  Building wheel for playsound (setup.py) ... done
  Created wheel for playsound: filename=playsound-1.3.0-py3-none-any.wh
l size=7037 sha256=d8f832c6ce6e78aec1b257931d0cc58987012012e6becc1c5f7b
39767d9949c7
  Stored in directory: /tmp/wsuser/.cache/pip/wheels/ba/39/54/c8f7ff9a8
8a644d3c58b4dec802d90b79a2e0fb2a6b884bf82
Successfully built playsound
Installing collected packages: pyttsx3, playsound, gTTS
Successfully installed gTTS-2.2.4 playsound-1.3.0 pyttsx3-2.90
```

In [ ]:
```
import gtts

from playsound import playsound
```

In [ ]:
```
# make request to google to get synthesis
tts = gtts.gTTS("Hello world")
```

In [ ]:
```
# save the audio file
tts.save("hello.mp3")
```

In [ ]:
```
# in spanish
tts = gtts.gTTS("Hola Mundo", lang="es")
tts.save("hola.mp3")
```

In [ ]:
```
# all available languages along with their IETF tag
print(gtts.lang.tts_langs())
```

```
{'af': 'Afrikaans', 'ar': 'Arabic', 'bg': 'Bulgarian', 'bn': 'Bengali',
'bs': 'Bosnian', 'ca': 'Catalan', 'cs': 'Czech', 'cy': 'Welsh', 'da':
'Danish', 'de': 'German', 'el': 'Greek', 'en': 'English', 'eo': 'Espera
nto', 'es': 'Spanish', 'et': 'Estonian', 'fi': 'Finnish', 'fr': 'Frenc
h', 'gu': 'Gujarati', 'hi': 'Hindi', 'hr': 'Croatian', 'hu': 'Hungaria
n', 'hy': 'Armenian', 'id': 'Indonesian', 'is': 'Icelandic', 'it': 'Ita
lian', 'iw': 'Hebrew', 'ja': 'Japanese', 'jw': 'Javanese', 'km': 'Khme
r', 'kn': 'Kannada', 'ko': 'Korean', 'la': 'Latin', 'lv': 'Latvian', 'm
k': 'Macedonian', 'ms': 'Malay', 'ml': 'Malayalam', 'mr': 'Marathi', 'm
y': 'Myanmar (Burmese)', 'ne': 'Nepali', 'nl': 'Dutch', 'no': 'Norwegia
n', 'pl': 'Polish', 'pt': 'Portuguese', 'ro': 'Romanian', 'ru': 'Russia
n', 'si': 'Sinhala', 'sk': 'Slovak', 'sq': 'Albanian', 'sr': 'Serbian',
'su': 'Sundanese', 'sv': 'Swedish', 'sw': 'Swahili', 'ta': 'Tamil', 't
e': 'Telugu', 'th': 'Thai', 'tl': 'Filipino', 'tr': 'Turkish', 'uk': 'U
krainian', 'ur': 'Urdu', 'vi': 'Vietnamese', 'zh-CN': 'Chinese', 'zh-T
W': 'Chinese (Mandarin/Taiwan)', 'zh': 'Chinese (Mandarin)'}
```

Build the Chatbox

Rmark: Sometimes it has issues connect to the server

In [1]:
```
import _thread
import socket
import threading
```

In [ ]:
```
"""AF_INET is the address domain of the
socket. This is used when we have an Internet Domain with
any two hosts The 2nd context of the code is the type of socket. """
s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
s.setsockopt(socket.SOL_SOCKET,socket.SO_REUSEADDR,1)
# piece of code to allow IP address & Port
host="127.0.0.1"
port=5000
s.bind((host,port))
s.listen(5)
clients=[]
#code to allow users to send messages
def connectNewClient(c):
     while True:
        global clients
        msg = c.recv(2048)
        msg ='Online ('+str(clients.index(c)+1)+'):  '+msg.decode('asci
i')
        sendToAll(msg,c)
def sendToAll(msg,con):
    for client in clients:
        client.send(msg.encode('ascii'))

while True:
    c,ad=s.accept()
    # Display message when user connects
    print('*Server Connected ')
    clients.append(c)
    c.send(('Online ('+str(clients.index(c)+1)+')').encode('ascii'))
    _thread.start_new_thread(connectNewClient,(c,))
```

In [2]:
```
import tkinter
import socket
import _thread
import sys
```

In [3]:

```
# Code to create a new client socket  and connect to the server

i = 3
client = 0
start = True
def sendMessage ():
    msg = txt.get()
    client.send(msg.encode('ascii'))


def recievingMessage (c):
    global i
    while True :
        msg=c.recv(2048).decode('ascii')
        if not msg :
            sys.exit(0)
        global start
        if (start) :
            start = False
            #tkinter codes starts
            window.title(msg)
            continue
        msglbl = tkinter.Label(window,text=msg)
        msglbl['font']=("Courier",10)
        msglbl['bg']='black'
        msglbl['fg']='#0aff43'
        msglbl['width']=50
        msglbl.grid(columnspan=2,column=0,row=i,padx=5)
        i += 1
#Socket Creation
def socketCreation ():
    c = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    c.setsockopt(socket.SOL_SOCKET,socket.SO_REUSEADDR,1)
#Local Host
# import all functions /
#  everthing from chat.py file
    host = '127.0.0.1'
    port = 5000
    c.connect((host,port))
    global client
    client = c
    send['command'] = sendMessage
    _thread.start_new_thread(recievingMessage, (c,) )




#Creating a window
window = tkinter.Tk()
window.title('Chatbox')
window['bg']='#242424'

window['padx']=10
window['pady']=10
#Adding Elements
#Entry
```

```
txt = tkinter.Entry(window)
txt['width']=50
txt['relief']=tkinter.GROOVE
txt['bg']='#f5f6f7'
txt['fg']='red'
txt['font']=("Courier",12)
txt.grid(column=0,row=1,padx=5,pady=15)
#Button
send = tkinter.Button(window,text="Send")
send['relief']=tkinter.GROOVE
send['bg']='red'
send['fg']='white'
send['activebackground']='#404040'
send['padx']=3
send['font']=("Courier",10)
send.grid(column=1,row=1,padx=5,pady=15)



_thread.start_new_thread(socketCreation, () )



window.mainloop()
```