UNIVERZITET U TUZLI
FAKULTET ELEKTROTEHNIKE

Zadaća br. 2

PREDMET:Obrada digitalnih signala

Tema:

Analiza govornog signala-Primjena Speech Recognition-a u Python-u

Studenti:
Amar Mehmedović
Mahir Terzić

Profesor:
Dr.Sc. Nermin Suljanović,red.prof

Tuzla,januar 2023. godine

Uvod

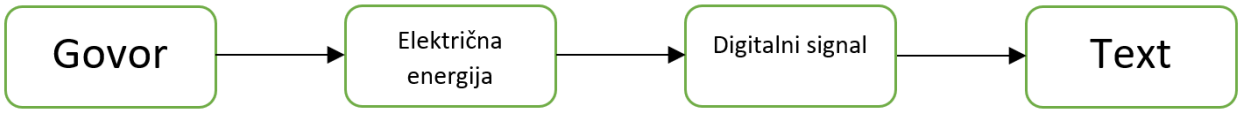
U dosadašnjem toku studija smo se upoznali sa korištenjem Python programskog jezika za obradu digitalnih signala.Pokazali smo kako možemo koristiti Python za ispitivanje osobina signala,množenje/sabiranje više signala, dodavanje šuma itd. Ipak,do sada nismo imali priliku koristiti govorni signal i upotrebljavati ga u Python-u.

Jedna od prednosti ovog programskog jezika u odnosu na ostale jeste što nudi mogućnost jednostavnog korištenja mikrofona odnosno govora unutar našeg programa.Sve što je potrebno jeste da korisnik pomoću eksternog mikrofona izgovara slova,riječi ili rečenice a računar će prepoznavati energiju njegovog glasa te zatim vršiti konverziju tog glasa u električnu energiju.Takva energija se poslije pretvara u digitalni signal koji možemo upotrebljavati po želji.

Da bi imali ovu mogućnost,potrebno je instalirati već unaprijed pripremljene pakete za konverziju govornog signala u digitalni signal odnosno u tekst. U nastavku rada ćemo pokazati kako zapravo funkcionišu ti paketi, šta je to Speech Recognition u Python-u te kako se može primjenjivati u našem kodu.

Speech Recognition

Korištenjem Speech Recognition-a u Python-u možemo kreirati programe koji će razumijevati govor te ga pretvarati u tekstualnu poruku.
Speech Recognition koristi moderne kompjuterske tehnologije i lingvistiku kako bi pretvorio sve što mikrofon registruje u digitalni signal,koji možemo iskorištavati i modifikovati lakše nego što bi to uradili sa govornim signalom.



Na prikazanoj blok šemi možemo vidjeti u kojem smjeru se izvršava konverzija. Korisnik izgovara riječi,a računar ima zadatak da identificira njegov glas i razumije ono što on govori. Da bi to bilo moguće moraju se u obzir uzeti mnoge mogućnosti,na kojima su programeri radili dugi niz godina: korisnikove godine, spol, akcenat, jezik, fraziranje i sl. Također ,vrlo je važno da sistem ima jasan pristup glasu korisnika bez pozadinskih smetnji(šuma) a u suprotnom će računar izvršiti pogrešan unos glasovne poruke. Ako to želimo izbjeći,potrebno je da imamo kvalitetan mikrofon,da budemo sami u prostoriji(ili da jedini govorimo), izbjegavamo teško razumljive riječi itd. ili da upotrebljavamo filtere koji će izdvojiti naš glas od svih ostalih zvukova.

Nakon prvog koraka,odnosno unosa glasa , vrši se pretvorba govora u električnu energiju. Kao što znamo, takva energija će nam predstavljati analogni signal a za našu upotrebu nam je potreban digitalni signal pa prelazimo na treći korak odnosno konverziju električne struje u digitalni signal.

Nakon trećeg koraka već imamo željeni rezultat odnosno digitalni signal koji možemo uređivati,mjenjati,koristiti u programu i sl. Za našu upotrebu je pogodan tekst,pa u posljednjem koraku dobijamo konačan proizvod obrade odnosno tekstualnu poruku.

Da bi bolje shvatili upotrebu Speech Recognition-a ali i samu konverziju govornog signala u text, pogledati ćemo primjer Python Code-a za igru pogađanja riječi.

Python Code

Za uspješno izvođenje Python Code-a i korištenje govornog signala ,potrebno je instalirati tri paketa : pyaudio, pyttsx3, SpeechRecognition.

- pyaudio→instaliranjem ovog paketa možemo koristiti Python za slušanje i snimanje audio-a na različitim operativnim sistemima(Windows,Linux,...)
- pyttsx3→instaliranjem ovog paketa možemo vršiti konverziju tekstualne poruke u govorni signal
- SpeechRecognition→instaliranjem ovog paketa , računar dobija mogućnost da identificira riječi sa ulaznog uređaja(mikrofon)

```
import speech_recognition as sr
import pyttsx3
import random as rd
import time as t
import wave
import numpy as np
import matplotlib.pyplot as plt
```

Nakon instalacije paketa, vršimo importovanje Google-ovog SR(Speech Recognition-a) koji ćemo upotrebljavati da bi vršili konverziju iz govora u tekst. Importujemo i prethodno navedeni pyttsx3,random za randomiziranje vrijednosti u našem string-u te vrijeme odnosno time čiju upotrebu ćemo objasniti poslije. Numpy te maptplotlib.pyplot koristimo za plotanje odnosno grafički prikaz signala.

```python
#Funkcija za speech recognition
def prepozGovor(recognizer,microphone):
    if not isinstance(recognizer,sr.Recognizer):
        raise TypeError("Instance 'recognizer' must be 'Recognizer' type")
    if not isinstance(microphone,sr.Microphone):
        raise TypeError("Instance 'microphone' must be 'Microphone' type")
    with microphone as source:
        recognizer.adjust_for_ambient_noise(source)
        audio=recognizer.listen(source)
    response={"success":True,"error":None,"transcription":None}

    try:
        response["transcription"]=recognizer.recognize_google(audio,language="en-US",show_all=False)
    except sr.RequestError:
        response["success"]=False
        response["error"]="API was unreachable"
    except sr.UnknownValueError:
        response["error"]="Unable to recognize speech"

    return response
```

Dalji kod nam pokazuje funkciju za speech recognition.Ovaj dio koda je i najvažniji dio koji pokazuje razne mogućnosti za unos i obradu glasovnog signala. Definisali smo funkciju prepozGovor koja uzima dva parametra: microphone i recognizer. Kod vrši ispitivanje da li smo definisali dobar microphone i recognizer jer da bi program radio mora biti identifikovan uređaj za input glasa te mora postojati prepoznavač glasa inače naš kod gubi smisao. Source je izvor odnosno naš izvor je u ovom slučaju mikrofon koji koristimo za unos audio podataka. Audio pohranjuje ono što izgovorimo preko mikrofona tj. sačuva informaciju u obliku digitalnog signala. Response identifikuje da li je unos signala uspješan ili ne. Nastavak koda je pokušavanje upotrebe Google-ovog SR prilikom čega smo jezik postavili za engleski tako da ako želimo da unos radi, riječi moramo izgovarati na engleskom a ne našem jeziku. RequestError znači da je zahtjev neuspješan dok UnknownValueError ukazuje da SR nije razumio ono što smo izgovorili.

```python
#Funkcija za konvertovanje teksta u govor
def tekstUGovor(tekst):
    eng=pyttsx3.init()
    eng.setProperty('rate',160)
    eng.say(tekst)
    eng.runAndWait()
```

Ovaj dio koda je zadužen za konvertovanje teksta u govor gdje je tekst neki niz riječi(string) koji prosljeđujemo u funkciju.Vrši se konvertovanje teksta u govorni signal gdje podešavamo setProperty odnosno podešavamo rate koji predstavlja

koliko brzo će biti izgovoren tekst kada se konvertuje u govorni signal.Sa runAndWait zapravo obustavlja sve ostale naredbe i očekuje unos kako bi nastavio dalje izvršavanje koda.

```
recognizer=sr.Recognizer()
microphone=sr.Microphone()
```

Podešavamo prepoznavač i mikrofon na sr odnosno da SpeechRecognition bude Google-ov te da on vrši indetifikaciju i upotrebu mikrofona. Postoje i opcije da mi definišemo vlastiti input uređaj po želji npr. ako imamo više mikrofona .

```
NUM_GUESS=3
PROMPT_LIM=5
TRY=["none","first","second","third"]
WORDS=["White","Yellow","Green","Blue","Orange","Black","Red","Purple","Gray","Brown"]
word=rd.choice(WORDS)
words=" ".join(WORDS)
print(words);
instructions="I'm thinking about one of these colors:{words}, You have 3 tries to guess which one".format(words=words)
tekstUGovor(instructions);
t.sleep(1);

for i in range(NUM_GUESS):
    for j in range(PROMPT_LIM):
        tekstUGovor("{num} guess. Speak!".format(num=TRY[i+1]))
        guess=prepozGovor(recognizer,microphone,i);
        if guess["transcription"]:
            break
        if guess["error"]=="Unable to recognize speech":
            tekstUGovor("I didn't quite catch that. What did you say?")
    if guess["error"]=="API was unreachable":
        tekstUGovor("ERROR, API not available for use")
        break;
    tekstUGovor("You said: {}".format(guess["transcription"]))
    s=wave.open(TRY[i+1]+"-guess-microphone-result.wav","r")
    signal=s.readframes(-1)
    signal=np.fromstring(signal,"int16")
    fs=s.getframerate()
    ti=np.linspace(0,len(signal)/fs,num=len(signal))
    plt.subplots_adjust(left=0.1,bottom=0.1,right=0.9,top=0.9,wspace=0.7,hspace=1.2)
    plt.subplot(3,1,i+1)
    plt.xlabel("Duzina signala")
    plt.ylabel("Frekvencija signala")
    plt.stem(ti,signal)
    plt.grid()
    if guess["transcription"]==word:
        tekstUGovor("Correct, you win")
        break;
    elif i<2:
        tekstUGovor("Incorrect, try again!")
    else:
        tekstUGovor("Sorry, you lose, i was thinking of the color {}".format(word))
```

Dosadašnji dio koda je vezan za pripremu SR i svih ostalih funkcionalnosti a mi dalje imamo mogućnost da ih upotrebljavamo po našoj želji. Nastavak programa

prikazuje Python Code za kreiranje igre pogađanja riječi. U jedan niz riječi navodimo ,na engleskom jeziku, boje koje će biti pohranjene na memorijskoj lokaciji WORDS. Pomoću random tj. rd opcije program vrši nasumični odabir jednog elementa stringa odnosno jedne boje i vrši ispisivanje svih ponuđenih boja. Naš cilj je pogoditi koja od navedenih boja je izbor računara odnosno naš unos treba biti jednak tom elementu stringa. NUM_GUESS predstavlja broj pokušaja,u našem slučaju 3, odnosno moramo imati 3 validna glasovna signala za koje će se izvršiti konverzija u tekst te zatim izvršiti poređenje sa elementom koji je slučajni odabir. Naravno, ukoliko prvi pokušaj bude uspješan,nema potrebe za izvođenjem drugog ili trećeg pokušaja. Upotrebljavamo tekstUGovor funkciju odnosno provlačimo je kroz petlju čije je ograničenje broj pokušaja odnosno koliko je glasovnih unosa dozvoljeno.

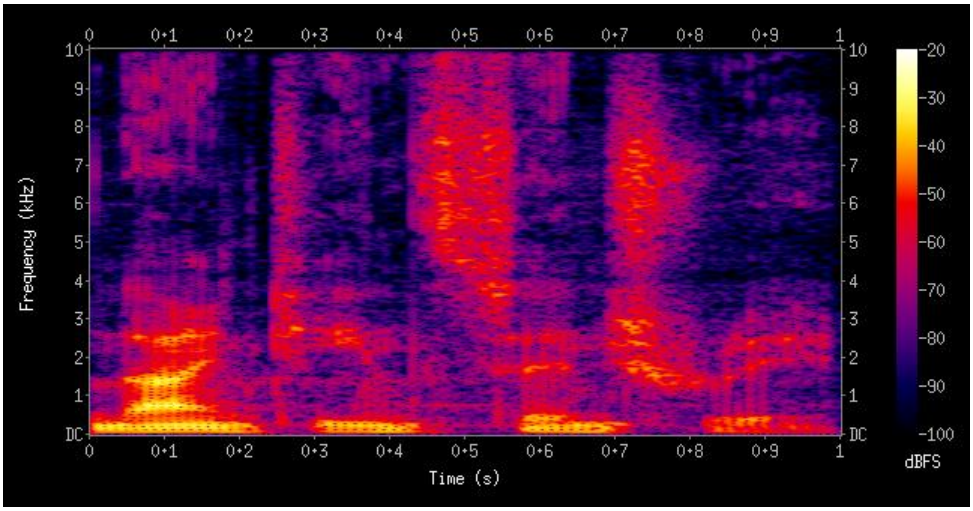
```
white yellow green blue orange black red purple gray brown
result2:
{   'alternative': [   {'confidence': 0.79887027, 'transcript': 'blue'},
                       {'transcript': 'glow'},
                       {'transcript': 'blow'},
                       {'transcript': 'Google'},
                       {'transcript': 'hello'}],
    'final': True}
C:\Users\falco\Downloads\Zadaca_SR (1).py:77: DeprecationWarning: The binary mode
of fromstring is deprecated, as it behaves surprisingly on unicode inputs. Use
frombuffer instead
  signal=np.fromstring(signal,"int16")
result2:
{   'alternative': [   {'confidence': 0.88712645, 'transcript': 'yellow'},
                       {'transcript': 'Yello'},
                       {'transcript': 'yelo'}],
```

Kod iznad pokazuje kako zapravo izgleda rezultat kompajliranja/pokretanja našeg programa. Vidimo da su ponuđene boje i da se od nas očekuje izbor . Nakon što preko input uređaja unesemo neku riječ,zadatak SR je da identifikuje tu riječ te zatim naša funkcija vrši poređenje. Ukoliko je jednakost netačna, traži se naredni pokušaj i taj proces se ponavlja dok ne postignemo limit pokušaja.Confidence govori kolika je vjerovatnoća da smo rekli tu riječ jer sve dok postoje i najmanje smetnje,Speech Recognition ne može biti siguran da je izgovorena ta riječ.

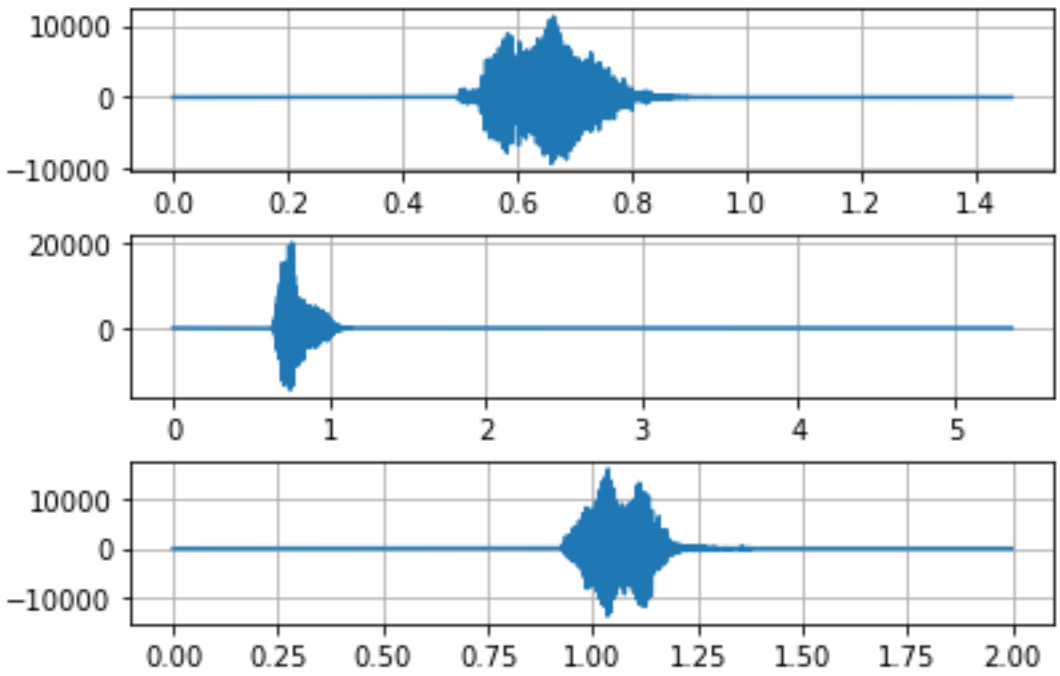
Analiza govornog signala

Ljudski glas kao i svaki zvuk ima svoje karakteristike. Kada govorimo, najvažnije osobine našeg glasa su vremenske pauze između svakog slova,riječi,rečenice zatim amplituda ,frekvencija glasa i sl. Sve ove osobine su vrlo važne kako bi druge osobe mogle razumiti ono što mi govorimo pa tako je isto i sa računarima. Da bi Speech Recognition uspješno identifikovao našu glasovnu poruku, potrebno je da analizira mnogobrojne aspekte. Jedan od najvažnijih aspekta je frekvencija glasa koja je važna karakteristika svakog analognog signala.Najjednostavniji način za analizu govornog signala jeste podjela u najsitnije dijelove kako bi se uspješno analizirao svaki segment .

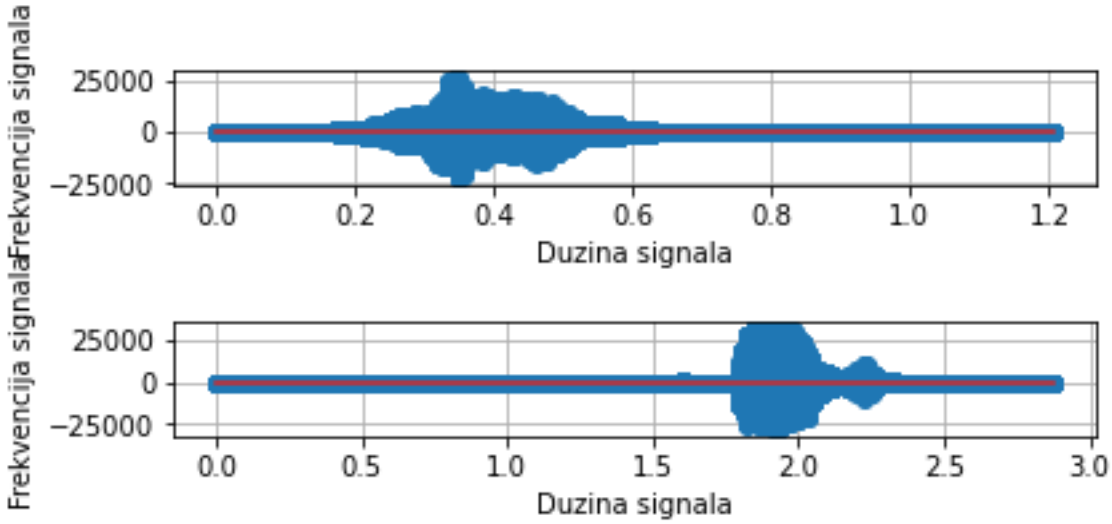
Kada u našem programu izgovorimo neku riječ, SR će izvršiti analizu svakog najsitnijeg dijela te riječi i na osnovu amplitude,frekvencije,glasnoće, i raznih faktora odlučiti o kakvom analognom signalu se radi te kasnije takav signal konvertuje u digitalni. Većina sistema za detekciju glasa koristi pristup komparacije kao i Google SR. S obzirom da nikada osoba neće izgovoriti neku riječ na isti način i uvijek će postojati razlike pa makar one bile u najsitnijim jedinicama frekvencije, potrebno je da za svaki glas postoji određen interval talasa koji će se porediti kako bi utvrdili kojem intervalu pripada izgovorena riječ ili slovo. Kao što ljudski mozak ima mogućnost razumijevanja raznih jezika,glasova,akcenata tako i računari koriste spektogram koji prikazuje vizualne spektre raznih frekvencija. Ovo je zapravo najbolja kopija ljudskog razumijevanja, i najjednostavniji način da sistem izvrši komparaciju raznih ulaznih poruka.



Spektogram



Za naš primjer,imamo grafički prikaz tri različita unosa po jedne riječi preko ulaznog uređaja. Amplituda nam predstavlja frekvenciju signala odnosno glasa što je najvažnija karakteristika prilikom identifikacije govornog signala. Dok ordinata predstavlja frekvenciju signala,apcisa predstavlja dužinu signala.Kolika je dužina signala zavisi naravno od toga koliki je naš glasovni unos odnosno koliko dugo je trebalo SR da registruje i identifikuje naš glas preko ulaza.



Slika iznad prikazuje grafički prikaz dva signala,odnosno bez zadnjeg signala što znači da smo našu riječ pogodili iz drugog pokušaja te nije bilo potrebe za trećim.

Zaključak

Korištenje govora u računarskim tehnologijama je jedna od vrlo korisnih i efikasnih metoda. Zahvaljujući razvoju informacione nauke, upotreba glasovnog signala kod elektronskih uređaja postala je svakodnevnica. Ako razmotrimo, svaki dan imamo priliku viditi konverziju govornog signala u digitalni signal. Aplikacije koje prepoznavaju muziku, multimedija u vozilima, virtualni asistenti kod mobilnih uređaja(umjetna inteligencija) kao npr. Siri,Alexa te online prevodioci jezika su sve primjeri pretvorbe glasovnih naredbi u digitalne signale.

Sve navedene aplikacije svoj zadatak obavljaju na sličan princip,kao što smo gore objasnili , i predstavljaju jednostavan način korištenja govora(glasa) za upravljanje elektronskih uređaja prilikom čega se vrši konverzija iz analognog u digitalni signal. Ipak,većina ovih uređaja vrše i obrnutu operaciju npr. Siri odgovara korisniku tako što pretvara tekstualnu poruku u glas koji zatim šalje na izlaz odnosno zvučnik uređaja što je zapravo suprotno od onoga što smo mi uradili.

Kroz naš primjer smo objasnili koliko je zapravo primjena govornog signala jednostavna ali definitivno olakšava unošenje podataka koje bez ikakvih problema možemo izmjenjivati u obliku digitalnog signala.

Zaključujemo da je govorni signal vrlo važna prednost kako i u Python-u tako i u svakodnevnom životu , i ako je pravilno iskorišten može predstavljati veliku olakšicu prilikom korištenja elektronskih uređaja,aplikacija,programa te svih ostalih polja vještačke inteligencije.