**SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU**

**FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK**

Domagoj Rojnić

Diplomski studij

**ZADAVANJE NAREDBI WINDOWS SUSTAVU GLASOM**

Raspoznavanje uzoraka i strojno učenje

Osijek, 2021.

**SADRŽAJ**

[**1. Uvod** 1](#_Toc77427064)

[1.1. Opis projektnog zadatka 1](#_Toc77427065)

[**2. Prepoznavanje govora** 2](#_Toc77427066)

[2.1. Povijest 2](#_Toc77427067)

[2.2. Današnjica 3](#_Toc77427068)

[2.3. Glavne značajke sustava 4](#_Toc77427069)

[2.4. Algoritmi u sustavima raspoznavanja govora 4](#_Toc77427070)

[**3. Programsko rješenje** 6](#_Toc77427071)

[3.1. SpeechRecognition biblioteka 6](#_Toc77427072)

[3.2. PyAutoGUI biblioteka 7](#_Toc77427073)

[**4. Testiranje aplikacije** 10](#_Toc77427074)

[**5. Zaključak** 12](#_Toc77427075)

[**6. Literatura** 13](#_Toc77427076)

# **1. Uvod**

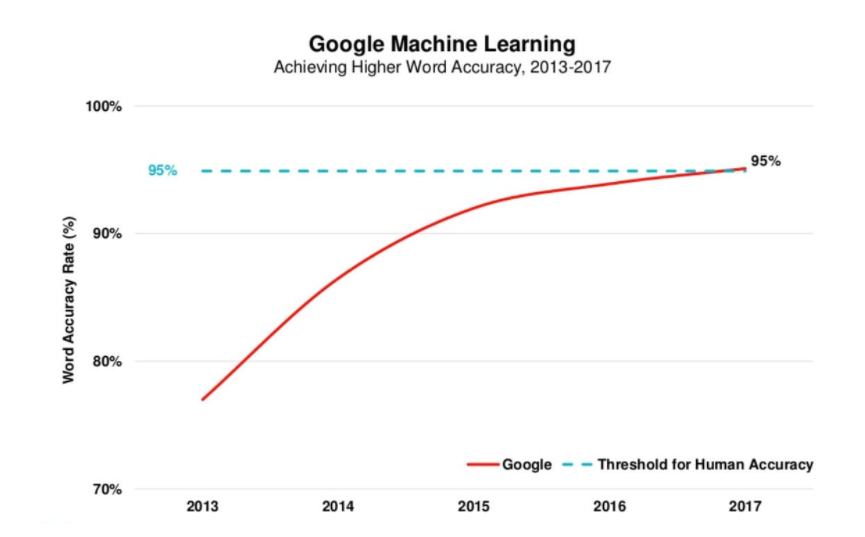
Prepoznavanje govora (engl. speech recognition), poznato kao i automatsko prepoznavanje govora (engl. automatic speech recognition, ASR), predstavlja sposobnost koja omogućava programu računalnog sustava obradu ljudskog govora i pretvorbu govora u tekstualni zapis. Iako se često miješa s prepoznavanjem glasa (engl. voice recognition), prepoznavanje govora fokusira se na pretvorbu govora iz verbalnog u tekstualni oblik, dok se kod prepoznavanja glasa želi identificirati glas pojedinog korisnika.

## 1.1. Opis projektnog zadatka

Cilj ovog projektnog zadatka bio je implementirati sustav koji omogućuje izvođenje Windows naredbi glasom. Potrebno je obraditi i transformirati glasovni oblik u tekstualni zapis rečenih naredbi. Odabrano je 8 naredbi koje se izvršavaju pomoću različitih biblioteka unutar Python skripte.

# **2. Prepoznavanje govora**

Računalna snaga i umjetna inteligencija uvelike stoje iza napretka u području obrade i prepoznavanja govora. Uz ogromne količine govornih podataka i kombinacijom s brzom obradom podataka, prepoznavanje govora dostiglo je točku gdje su njegove mogućnosti približno jednake ljudskim. Slikom 2.1. u nastavku dan je graf koji prikazuje napredak postotka točnosti prepoznavanja govora Google-ovog sustava za prepoznavanje ostvarenog metodama strojnog učenja.



Slika 2.1. Graf postotka točnosti Google-ovog sustava prepoznavanja govora

Iako je u posljednje vrijeme bilo puno koraka prema naprijed, prepoznavanje govora datira jos od ranih 1950-ih. Nadalje su kratko navedeni ključni događaji koji su oblikovali ovu tehnologiju u posljednjih 70 godina.

## 2.1. Povijest

Prvi sustavi prepoznavanja govora bili su usmjereni na brojeve, a ne riječi. 1952. Bell Laboratories dizajnirali su „Audrey“ sustav koji je mogao prepoznati pojedine izgovorene znamenke. Deset godina kasnije, IBM je predstavio sustav „Shoebox“ koji je razumio i odgovarao na 16 riječi.

Diljem svijeta zemlje su razvijale vlastito sklopovlje i programe koje je obrađivalo i prepoznavalo govor. Krajem 60-ih, omogućeno je prepoznavanje riječi koje sadrže do 4 samoglasnika i čak do 9 suglasnika.

Najveći napredak tijekom 1970-ih ostvaren je zahvaljujući Ministarstvu obrane SAD-a i DARPA-i. Oni su proveli istraživanje razumijevanja govora kojim su stvorili „Harpy“ sustav kojime je omogućeno prepoznavanje do 1000 riječi što je slično vokabularu trogodišnjeg djeteta.

80-ih godina sustavi počinju raspoznavati i po par tisuća riječi, a najveći proboj u prepoznavanju govora dolazi zahvaljujući statističkoj metodi skrivenih Markovljevih modela (engl. Hidden Markov Models, HMM). Umjesto čistog korištenja riječi i traženja obrazaca, HMM izračunava vjerojatnost da su nepoznati zvukovi zapravo riječi.

Sve do 2001. godine, sustavi raspoznavanja govora ostvaruju oko 80% točnosti. Veliku prekretnicu predstavlja Google-ov *Voice Search*, aplikacija koja je dana milijunima ljudi na korištenje čime je sustav s vremenom uključivao raspoznavanje 230 milijardi riječi.

2010-ih Apple lansira Siri, Amazon Alexu, a Google izbacuje Google Home, gdje potrošačima postaje sve ugodnije razgovarati sa strojevima.

## 2.2. Današnjica

Danas se neke od najvećih tehnoloških tvrtki natječu u ostvarivanju što veće točnosti prilikom prepoznavanja govora. 2016. godine IBM je postigao stopu pogreške od 6,9%. 2017. godine Microsoft postiže 5,9% nakon čega je ubrzo IBM nadmašio njihovu stopu s 5,5%. Međutim, trenutnu pobjedu odnosi Google koji je uspio ostvariti stopu pogreške od 4,9%.

Korištenje sustava raspoznavanja govora danas nema granica, pri čemu se uvelike pomaže tvrtkama i klijentima u uštedi vremena. U automobilskoj industriji sustavi prepoznavanja poboljšavaju vozačevu sigurnost omogućavanjem izvođenja naredbi aktivacijom govora poput navigacije. Svakodnevno se koriste digitalni asistenti koji su ugrađeni u mobilne uređaje, time pomažući korsnicima s raznim zahtjevima. U zdravstvu omogućuju zapisivanje i vođenje dnevnika bolesti i dijagnoza pojedinih pacijenata.

## 2.3. Glavne značajke sustava

Mnogi programi su dostupni za prepoznavanje govora, ali naprednija rješenja koriste postupke strojnog učenja i umjetne inteligencije. Oni integriraju gramatiku, sintaksu, strukturu i sastav zvučnih signala kako bi razumjeli i obradili ljudski govor. Različite vrste sustava omogućuju različitim korisnicima da prilagode tehnologiju svojim specifičnim zahtjevima, od jezika i nijansi govora do prepoznavanja robnih marki. Ipak, svi ovi sustavi imaju zajedničke značajke koje je potrebno implementirati na neki od mogućih načina kako bi se izgradio kvalitetan i robustan sustav raspoznavanja.

Pridruživanje težina određenim riječima koje se često govore izvan pojmova koji su već u osnovnom riječniku uvelike povećavaju preciznost sustava. Označavanje govornika također poboljšava sustave jer time se može kao izlaz dobiti slijed govora koji označava pojedine govornike u razgovoru s više sudionika, čime se određuje doprinos svakog govornika. Jedna od najvažnijih značajki je mogućnost prilgodbe okolišu i akustičnom okruženju, čime se mogu identificirati stalno pristuni zvukovi koji bi mogli ometati sustav raspoznavanja. Njih je potrebno prepoznati i kao takve ukloniti iz govora koji se prepoznaje kako bi se dobio što čišći i precizniji odgovor. Naposlijetku, potrebni su razni filteri koji će filtrirati određene riječi, fraze i sanirati sam govor.

## 2.4. Algoritmi u sustavima raspoznavanja govora

Sustavi raspoznavanja govora implementiraju različite algoritme i tehnike kojima se govor raspoznaje i transformira u tekst. U nastavku su kratko opisani najznačajniji algoritmi koji se danas koriste.

Obrada prirodnog govora (engl. Natural language processing, NLP) kao područje umjetne inteligencije zasniva se na poboljšanju interakcije čovjeka i računala putem pretvorbe govora u tekst različitim tehnikama obrade i transformacije riječi i rečenica.

Skriveni markovljevi modeli (HMM) predviđaju vjerojatnost zvukova da su riječi te se koristi kao model unutar prepoznavanja govora dodjeljujući oznake svakoj jedinici, odnosno riječima ili slogovima u rečenicama. Ove oznake mapiraju navedeni ulaz i omogućavaju sustavu da obradi najprikladniji slijed govora temeljem tih oznaka.

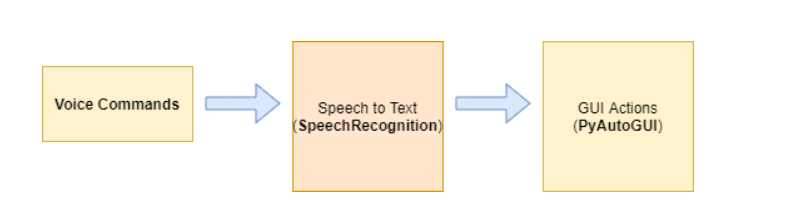
N-grami (engl. N-grams) znakova predstavlja najjednostavniji tip jezičnog modela koji rečenicama ili frazama dodjeljuje vjerojatnost. N-gram je zapravo slijed od n riječi u rečenici. Gramatika i vjerojatnost određenih sljedova riječi koriste se za poboljšanje prepoznavanja i točnosti.

Neuronske mreže (engl. Neural networks, NN) predstavljaju algoritam dubinskog učenja pomoću kojeg se pronalaze skrivene povezanosti među riječima i rečenicama čime se znatno povećava preciznost sustava prepoznavanja.

Diarizacija govornika (engl. Speaker diarisation) identificira i segmentira govor prema identitetu govornika pri čemu znatno pomaže sustavu u razlikovanju pojedinaca u razgovoru.

# **3. Programsko rješenje**

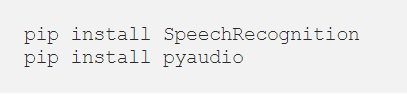
Prethodno naveden cilj projekta također se može prikazati dijagramom na slici 3.1. Korisnik kao ulaz daje glasovnu naredbu putem mikrofona, koji može biti ili ugrađen u Windows računalo ili eksterno povezan, nakon čega se glasovna naredba pretvara u tekst koristeći metode SpeechRecognition biblioteke. Konvertirani tekst je tada mapiran pojedinim GUI zadatcima, poput upravljanja mišem ili tipkovnicom, prilikom čega su aktivnosti izvedene pomoću PyAutoGUI biblioteke.



Slika 3.1. Dijagram zadatka

## 3.1. SpeechRecognition biblioteka

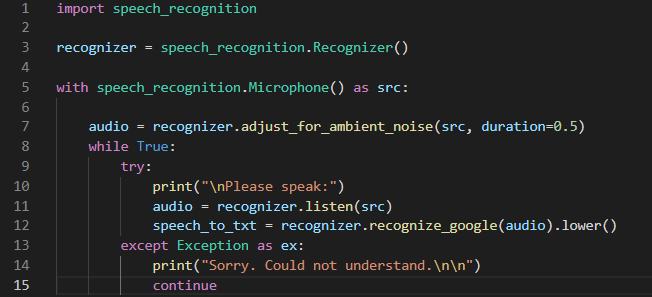
SpeechRecognition Python biblioteka nudi veliki niz mogućnosti u različitim aspektima kućnih automatskih sustava, učenja jezika pa sve do digitalnih asistenata. Korištenjem ove biblioteke omogućena je pretvorba govora u tekst. Podupire nekolicinu API-ja, *online* te čak i *offline*. Instalacija biblioteke u projekt izvodi se kako je prikazano na slici 3.1.1. prilikom čega se također instalira *pyaudio* biblioteka kako bi se moglo pristupiti mikrofonu.



Slika 3.1.1. Instaliranje biblioteka

Sljedećom slikom, vidljiv je kod kojim se uvodi biblioteka te inicijalizira *recognizer* objekt koji sluša glas putem mikrofona i pretvara ga u tekst. Pozivom funkcije *recognizer.adjust\_for\_ambient\_noise()* je opcionalno, ali vrlo korisno. Ova funkcija sluša ulazni zvuk bez govora onoliko sekundi koliko je predano u parametru *duration* i mijenja vrijednost *energy\_threshold* ovisno o količini buke koja je prisutna u sobi. Ovo radi kako bi prilikom raspoznavanja govora mogao odstraniti buku okoliša i što točnije klasificirati danu naredbu. Njome se također mjeri „osjetljivost“ *recognizer* objekta. Dobivena vrijednost varira u rasponu od 50 do 4000 gdje veće vrijednosti predstavljaju više buke i manju osjetljivost na ulazni signal govora.

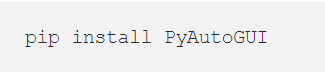
Funkcijom *recognizer.listen()* se sluša naredba putem mikrofona koja se sprema u varijablu audio. Zatim se varijabla prosljeđuje *recognizer* objektu koji pretvara zvuk u tekst.



Slika 3.1.2. Početni dio koda

## 3.2. PyAutoGUI biblioteka

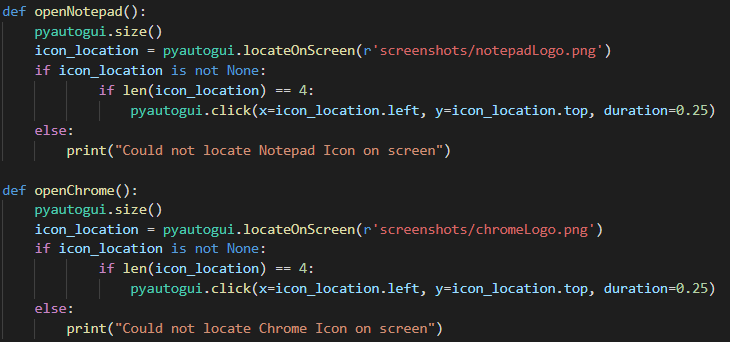
Python nudi biblioteku PyAutoGUI za GUI automatizaciju koja simulira klikanje miša i pritiskanje tipki na tipkovnici kao da ih i sam čovjek obavlja. Na primjer, moguće je sve od simuliranja kretnje miša, povećavanja i smanjivanja otvorenih prozora pa sve do mijenjanja razine zvuka i otvaranje instaliranih aplikacija. Instaliranje je prikazano na slici 3.2.1. te se izvodi na isti način kao i kod SpeechRecognition biblioteke.



Slika 3.2.1. Instaliranje PyAutoGUI biblioteke

Kretnje miša simuliraju se na način da se dohvaćaju x i y koordinate na zaslonu. Zaslon se može zamisliti kao 2D prostor čije se (0, 0) koordinate nalaze u gornjem lijevom kutu. Kretnjom udesno povećava se x vrijednost te kretnjom prema dolje povećava se y vrijednost. Funkcijom *pyautogui.size()* dohvaća se dimenzija zaslona.

Metodom *pyautogui.locateOnScreen()* danu sliku pronalazi na zaslonu tako da traži točan raspored piksela kao na slici. Pronalaskom ikonice moguće je odrediti x i y koordinate te simulirati lijevi klik miša na te koordinate metodom *pyautogui.click().* Na taj način ostvareno je otvaranje Google Chrome i Notepad aplikacija. Definicije funkcija prikazane su na slici 3.2.2. Veliki nedostatak ove funkcije je taj da ukoliko postoji razlika i u jednom pikselu prilikom uspoređivanja slike i tražene ikonice, funkcija neće vratiti očekivani izlaz.

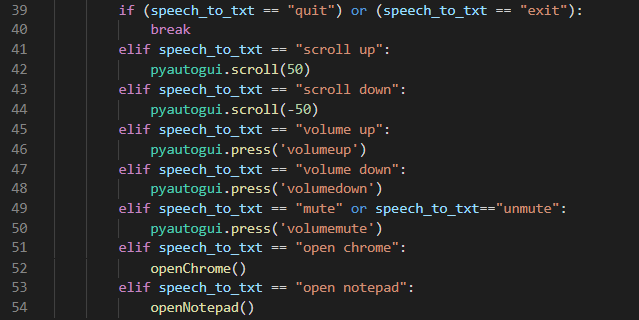


Slika 3.2.2. Definicije funkcija za otvaranje aplikacija

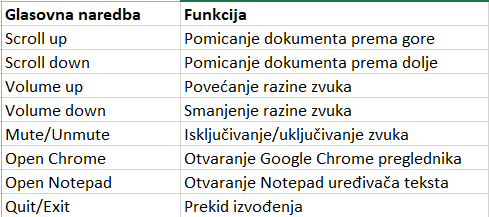
Simulacija pritiskanja tipki ostvarena je funkcijom *pyautogui.press()* kojoj se kao argument zadaje ime radnje koju tipka izvodi, dok se pomicanje srednje tipke miša ostvaruje funkcijom *pyautogui.scroll()* prilikom čega se kao argument predaje vrijednost za koju je potrebno pomaknuti miš u pikselima.

Slikom 3.2.3 danom u nastavku prikazan je dio koda kojim se provjerava koja je naredba rečena i izvođenje pripadajuće GUI operacije ovisno o vrijednosti koja je spremljena u *speech\_to\_text* varijabli.

Nadalje, mapiranje funkcija koje se izvode na određene glasovne naredbe prikazano je slikom 3.2.4.



Slika 3.2.3. Izvođenje operacija ovisno o izrečenoj naredbi



Slika 3.2.4. Naredbe aplikacije

# **4. Testiranje aplikacije**

Kao što je prethodno navedeno, SpeechRecognition biblioteka nudi više API-ja kojima se vrši obrada i transformacija glasa u tekst. Kako bi se utvrdilo koji najbolje odgovara danome problemu, izvršeno je testiranje aplikacije korištenjem sljedećih API-ja: CMU Sphinx, Google Speech Recognition, Google Cloud Speech API, Microsoft Bing Voice Recognition i IBM Speech to Text.

Prvo svojstvo koje je potrebno uvažiti je mogućnost korištenja API-ja s ili bez internetske povezanosti. Kako bi pristupili određenim *recognizer* objektima, potrebna je internet konekcija koja šalje serveru glasovni oblik, prilikom čega server obrađuje podatke, pretvara ih u tekst te ga vraća klijentu kao izlaz. Sljedećom tablicom prikazani su korišteni pristupi i je li potreban pristup internetu za njihov rad.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **CMU Sphinx** | **Google Speech Recognition** | **Google Cloud Speech API** | **Microsoft Bing Voice Recognition** | **IBM Speech to Text** |
| **Potreban internet** | NE | DA | DA | DA | DA |

Tablica 4.1. Potrebnost interneta za korištenje API-ja

Glavna značajka koja se testira je koliko dobro pojedini API klasificira govor i pretvara ga u tekst. Kako bi se dobili što precizniji rezultati, testiranje je izvršeno u 4 kruga. Svakim krugom prolazi se kroz sve naredbe i kroz sve API-je. Svaka naredba izrečena je 10 puta te se na temelju točnih i netočnih klasifikacija utvrđuje stopa pogreške. Također, svaki krug naredbe zadaje drugi govornik. U testiranju su sudjelovala 4 govornika različite dobi i spola, stariji i mlađi muškarac te starija i mlađa žena. Rezultati testiranja prikazani su u tablicom 4.2.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **CMU Sphinx** | **Google Speech Recognition** | **Google Cloud Speech API** | **Microsoft Bing Voice Recognition** | **IBM Speech to Text** |
| **Scroll up** | 7 | 2 | 1 | 3 | 2 |
| **Scroll down** | 6 | 3 | 3 | 4 | 2 |
| **Volume up** | 2 | **0** | 1 | 2 | 1 |
| **Volume down** | 3 | 0 | 0 | 2 | 0 |
| **Mute/unmute** | 2 | 1 | 0 | 1 | 0 |
| **Open Chrome** | 5 | 1 | 2 | **0** | 3 |
| **Open Notepad** | 4 | 1 | 0 | 2 | 1 |
| **Quit/Exit** | 1 | 0 | 0 | 0 | 2 |
| **Missclassified** | 30 | 8 | 7 | 14 | 11 |
| **WER (%)** | 37,5 | 10 | **8,75** | 17,5 | 13,75 |

Tablica 4.2. Rezultati testiranja

U tablici je prikazan broj pogrešno klasificiranih naredbi te u zadnjem retku postotak pogrešno klasificiranih naredbi (engl. Word error rate, WER). Kao što je vidljivo, najveći problem predstavljaju naredbe *Scroll up* i *Scroll down*, ponajviše iz razloga što se riječ *Scroll* često krivo klasificira kao *Call*. Riječi *Up* i *Down* u svim naredbama nisu predstavljala problem prilikom klasifikacije što je također vidljivo kod boljih rezultata *Volume* *up* i *Volume* *down* naredbi. *Open* *Chrome* i *Open* *Notepad* naredbe nisu predstavljale problem prilikom klasifikacije osim kod CMU Sphinx koji nije u potpunosti razumio izgovorenu naredbu. Najmanji broj krivo klasificiranih riječi vidljiv je kod naredbe *Quit*/*Exit*. Google Speech Recognition jedini je točno klasificirao sve primjere naredbe *Volume* *up*, dok je Microsoft Bing Voice Recognition jedini naredbu *Open* *Chrome* u potpunosti točno klasificirao svaki puta.

Bitno je napomenuti kako niti jedan API nije pokazao razliku između klasificiranja naredbi pojedinih govornika, što govori kako su sustavi kvalitetno izrađeni i jednako prepoznaju različite tonalitete glasova.

Iz tablice je također vidljivo kako je CMU Sphinx pokazao najlošije rezultate što se može interpretirati činjenicom da jedini radi bez potrebne internetske povezanosti. Ostala 4 API-ja ostvaruju slične rezultate, iako se za ovaj projektni zadatak najboljim pokazao Google Cloud Speech jer je postigao najmanju stopu pogreške.

# **5. Zaključak**

Ovim projektom vidljivo je kako postoji više mogućnosti upravljanja Windows sustavom te je prikazana i korištena mogućnost izvođenja naredbi govorom. Testiranjem različitih API-ja s više govornika utvrđeno je kako nema razlike u klasificiranju naredbi ovisno o govorniku te da Google, IBM i Bing sustavi raspoznavanja govora pokazuju vrlo dobre i slične rezultate. CMU Sphinx pokazuje znatno lošije rezultate što je objašnjivo činjenicom da ne treba internet prilikom klasificiranja. Daljnim radom valjalo bi testirati veći skup API-ja kao i veći skup naredbi kako bi se utvrdili detaljniji i opširniji rezultati.

# **6. Literatura**

https://pypi.org/project/SpeechRecognition/

https://pypi.org/project/PyAudio/

https://pypi.org/project/PyAutoGUI/

https://towardsdatascience.com/audio-deep-learning-made-simple-automatic-speech-recognition-asr-how-it-works-716cfce4c706

https://sonix.ai/history-of-speech-recognition

https://www.ibm.com/cloud/learn/speech-recognition

https://summalinguae.com/language-technology/guide-to-speech-recognition-technology/