Detekcija gesti prstiju/ruku zasnovana na nečujnom audio-signalu

Jurkić, Ena

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Sveučilišni studij elektrotehnike

DETEKCIJA GESTI RUKU ZASNOVANA NA NEČUJNOM AUDIO-SIGNALU

Diplomski rad

Ena Jurkić

Osijek, 2024.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Obrazac D1: Obrazac za ocjenu diplomskog rada na sveučilišnom diplomskom studiju

Ocjena diplomskog rada na sveučilišnom diplomskom studiju

Ime i prezime pristupnika:	Ena Jurkić
Studij, smjer:	Sveučilišni diplomski studij Elektrotehnika, Komunikacije i
Mat. br. pristupnika, god.	D-1483, 07.10.2022.
JMBAG:	0165073317
Mentor:	prof. dr. sc. Marijan Herceg
Sumentor:	
Sumentor iz tvrtke:	Zvonimir Kaprocki
Predsjednik Povjerenstva:	izv. prof. dr. sc. Ratko Grbić
Član Povjerenstva 1:	prof. dr. sc. Marijan Herceg
Član Povjerenstva 2:	prof. dr. sc. Mario Vranješ
Naslov diplomskog rada:	Detekcija gesti prstiju/ruku zasnovana na nečujnom audio-signalu
Znanstvena grana diplomskog rada:	**Telekomunikacije i informatika (zn. polje elektrotehnika)**
Zadatak diplomskog rada:	Prepoznavanje i razlikovanje gesti prstiju/ruku korisnika postalo je važno zbog velikog porasta upotrebe uređaja nosive tehnologije (engl. wearables), poput pametnog sata. Jedan od načina detekcije gesti prstiju/ruku korisnika je korištenje nečujnog audio-signala na način da se takav signal generira zvučnikom te se analizira reflektirani signal korištenjem mikrofona. U okviru ovog rada potrebno je razviti algoritam zasnovan na dubokom učenju koji će koristiti komercijalne zvučnike i mikrofone za generiranje nečujnog audio-signala i analizu reflektiranog nečujnog-audio signala.

Datum ocjene pismenog dijela diplomskog rada od strane mentora:	23.09.2024.
Ocjena pismenog dijela diplomskog rada od strane mentora:	Izvrstan (5)
Datum obrane diplomskog rada:	30.09.2024.
Ocjena usmenog dijela diplomskog rada (obrane):	Izvrstan (5)
Ukupna ocjena diplomskog rada:	Izvrstan (5)
Datum potvrde mentora o predaji konačne verzije diplomskog rada čime je pristupnik završio sveučilišni diplomski studij:	01.10.2024.

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

IZJAVA O IZVORNOSTI RADA

Osijek, 01.10.2024.

Ime i prezime Pristupnika:	Ena Jurkić
Studij:	Sveučilišni diplomski studij Elektrotehnika, Komunikacije i informatika
Mat. br. Pristupnika, godina upisa:	D-1483, 07.10.2022.
Turnitin podudaranje [%]:	15

Ovom izjavom izjavljujem da je rad pod nazivom**: Detekcija gesti prstiju/ruku zasnovana na nečujnom audio-signalu**

izrađen pod vodstvom mentora prof. dr. sc. Marijan Herceg

i sumentora

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis pristupnika:

Sadržaj

1. UVOD

Posljednjih je godina brzi napredak tehnologije značajno utjecao na način na koji ljudi komuniciraju s uređajima. Tako brzim porastom upotrebe nosivih tehnologija, poput pametnih satova, i sve većom integracijom interneta objekata (engl. *Internet of Things, IoT*) u svakodnevicu raste potražnja za intuitivnijim metodama interakcije. Tradicionalne metode unosa kao što su ekrani osjetljivi na dodir i tipkovnice, iako su učinkovite, sve se više nadopunjuju ili zamjenjuju prirodnijim oblicima komunikacije, kao što su glasovne naredbe i kontrole temeljene na gestama.

Jedan od načina primjene ovakvog oblika komunikacije je detekcija gesti korisnika, gdje se prepoznavanjem unaprijed definiranih pokreta ruku pokreću različite naredbe za kontrolu uređaja ili pojedinih aplikacija. Prepoznavanje pokreta nudi prirodnije korisničko sučelje, što je osobito korisno u situacijama kada su tradicionalni načini unosa nepraktični, kao što je tijekom vožnje, kuhanja ili za osobe s tjelesnim invaliditetom. Sposobnost upravljanja uređajima pomoću jednostavnih gestikulacija rukom može poboljšati korisničko iskustvo, pristupačnost i sigurnost.

S obzirom na porast potražnje za nenametljivim i učinkovitim kontrolnim mehanizmima, algoritmi za detekciju gesti ruku i prstiju su postali sve sofisticiraniji. Tradicionalne metode često se oslanjaju na vizualne podatke snimljene kamerama, koje, iako su učinkovite, dolaze s izazovima povezanim s privatnošću, uvjetima osvjetljenja i računalnim resursima u vidu vremena obrade i memorije. Kako bi se smanjila ta ograničenja, pojavile su se alternativne metode koje koriste nečujne audio signale. Za detekciju i klasifikaciju gesti, ove metode iskorištavaju pojavu Dopplerovog efekta gdje se frekvencija zvučnog vala mijenja u odnosu na promatrača uslijed kretanja objekta, u ovom slučaju ruke korisnika.

U ovom diplomskom radu bit će prikazan tijek izrade sustava za detekciju gesti ruku koja se temelji na Dopplerovom pomaku nečujnog audio signala. Rad algoritma zasnovan je na dubokom učenju i koristi se za upravljanje *Spotify Web* aplikacijom. Pritom algoritam može raspoznati jedanaest različitih gesti koje pokreću različite funkcije upravljanja aplikacijom u stvarnom vremenu.

Rad je podijeljen na sljedeći način. U drugom poglavlju je dan pregled postojećih studija na temu prepoznavanja gesti ruku i prstiju s fokusom na detekciju temeljenu na audio signalima i Dopplerovom efektu. U trećem poglavlju je detaljno opisan rad vlastitog sustava za detekciju gesti ruku. Navedene su geste kojima će se upravljati aplikacijom *Spotify Web* i opisan je način kreiranja potrebnih baza podataka za treniranje i evaluaciju rada algoritma. Opisan je način rada sustava sa

fizikalnog aspekta i postupak obrade signala. Zatim je definirana arhitektura neuronske mreže, opisan je tijek treniranja i rad neuronske mreže u realnim uvjetima. U četvrtom poglavlju je opisana izvedba *Spotify Web* aplikacije i njen rad u stvarnom vremenu. U petom poglavlju je napravljena evaluacija rada neuronske mreže te je na kraju rada dan zaključak.

2. PREGLED POSTOJEĆIH RJEŠENJA ZA DETEKCIJU GESTI PRSTIJU I RUKU

Sve veća se važnost pridaje razvoju sustava za detekciju gesti ruku i prstiju što bi olakšalo interakciju korisnika i računala i učinilo ju prirodnijom. Tijekom godina, razvijena su mnoga rješenja za detekciju gesti ruku i prstiju koja bi se mogla podijeliti u dvije osnovne kategorije: rješenja temeljena na nosivim tehnologijama i bez njih.

Što se tiče rješenja temeljenih na nosivim tehnologijama, oslanjaju se na senzore kao što su akcelerometri i žiroskopi ugrađeni u uređaje koje korisnici nose na sebi. Primjer takvih uređaja su pametni satovi, podatkovne rukavice (engl. *data glove*), senzori pričvršćeni na ruku ili dlan. Na primjer, podatkovne rukavice korištene u [1] raspoznaju 5 unaprijed definiranih gesti ruku, postižući pouzdane performanse u aplikacijama u virtualnoj stvarnosti. U radu [2] predstavljen je uređaj temeljen na akcelerometru koji je postigao gotovo savršenu točnost prepoznavanja pokreta ruke neovisno o korisniku koristeći neuronske mreže i podudaranje sličnosti (engl. *similarity matching*). Međutim, nosivi uređaji mogu biti nepraktični u određenim situacijama i mogu povećati troškove i složenost implementacije rješenja.

S druge strane, rješenja temeljena na tehnologijama koje korisnik ne treba nositi na sebi omogućuju prepoznavanje gesti putem interakcije ruke korisnika sa audio signalima i senzorima kao što su kamere. Ovakva rješenja su korisna za upotrebu na većim mobilnim uređajima jer omogućuju korisnicima unos gesti bez blokiranja zaslona, poboljšavajući korisničko iskustvo u situacijama kada korisnik nosi rukavice ili mu je uređaj u džepu. Rješenja ove kategorije se mogu dalje podijeliti na rješenja temeljena na računalnom vidu, radiofrekvencijske sustave i rješenja koja se temelje na ultrazvučnim valovima.

Sustavi temeljeni na računalnom vidu koriste kamere za snimanje gesti ruku i često rade segmentaciju područja ruke pomoću informacija o boji i/ili dubini, a značajke za dinamičke geste se koriste za treniranje klasifikatora kao što su skriveni Markovljevi modeli (engl. *Hidden Markov Model, HMM*) i stroj s potpornim vektorima (engl. *Support Vector Machine, SVM*). [3] Usprkos općenito visokoj preciznosti u kontroliranim uvjetima, ovakve metode loše će raditi u uvjetima slabog osvjetljenja, zahtijevaju puno računalnih resursa i da se korisnik nalazi u području vidljivosti kamere (engl. *Line of Sight, LOS*).

Radiofrekvencijski sustavi, kao što su Wi-Fi i RFID, stekli su popularnost zbog niske cijene i nenametljivosti. Wi-Vi [4] koristi radiofrekvencijske zrake i ISAR tehnike kako bi bilo moguće detektirati gestu čak i kroz zid. ISAR je radarska tehnika snimanja koja koristi kretanje

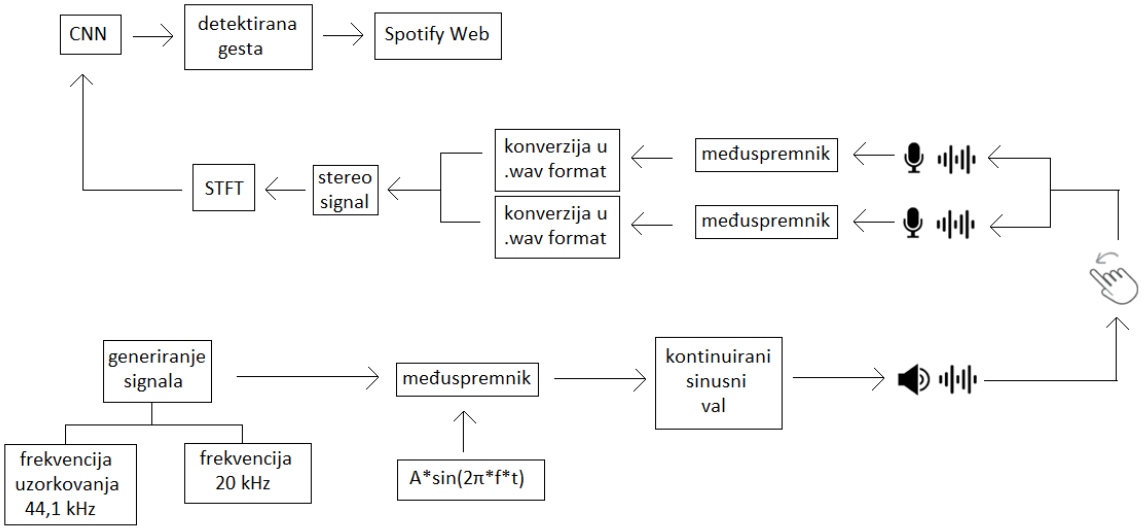
objekta za dobivanje radarske slike visoke rezolucije. [5] Sustav WiGest [6] iskorištava promjenu u jačini primljenog signala (engl. *Received Signal Strength Indicator, RSSI*) iz Wi-Fi mreže kako bi raspoznao različite geste ruke i utvrdio smjer i brzinu kretanja. WiFinger [7] raspoznaje pokrete prstiju s točnošću preko 93% korištenjem jednog Wi-Fi uređaja. Nedostatak ovog sustava je nepraktičnost za vanjsku uporabu zbog ovisnosti o bežičnim pristupnim točkama (engl. *Wireless Access Point, WAP*).

Ultrazvučni signali se sve češće koriste za detekciju gesti nudeći pritom visoku točnost detekcije. Na primjer, SonarGest [8] za detekciju pokreta ruku koristi tri ultrazvučna prijemnika i jedan odašiljač za detekciju 8 gesti ruku. Tehnika koja se pritom koristi je nadzirani model mješavine Gaussovih distribucija (engl. *Gaussian Mixture Model*) koji može zabilježiti distribuciju vektora značajki dobivenih iz signala gesti s Dopplerovim pomakom. Međutim, potrebno je prikupiti podatke za trening (što je potencijalno radno intenzivno i dugotrajno) i zahtijeva dodatni zvučni hardver. SoundWave [9] se također temelji na Dopplerovom efektu. Za razliku od SonarGest-a, ne koristi nikakvu dodatnu opremu; koristi ugrađene zvučnike i mikrofone u računalima i ne zahtjeva treniranje. SoundWave dizajnira tehniku dinamičkog praćenja vršnih vrijednosti temeljenu na pragu kako bi učinkovito zabilježio Dopplerov pomak, čime može razlikovati pet različitih gesti ruku i pri tome postiže točnost od 85%, međutim, ima poteškoća s mogućnošću praćenja ruke jer je prikladan samo za unaprijed definirane geste. FingerIO [10] i LLAP [11] koriste pametni telefon kao sonar, pomoću ugrađenih zvučnika i mikrofona. Kod sustava FingerIO, zvučnik emitira OFDM signale u rasponu od 18 do 20 kHz s cikličkim sufiksima, a mikrofon bilježi refleksije signala od ruke korisnika. Ciklički sufiksi se koriste za ispravljanje pogrešaka uzorkovanja kako bi se postiglo praćenje pokreta prstiju s milimetarskom točnošću. Reflektirani signali se analiziraju da bi se izračunala udaljenost između ruke i mikrofona i omogućilo 2D praćenje. FingerIO postiže 2D praćenje ruke s prosječnom pogreškom od 8 mm, a točnost detekcije ozbiljno opada kada se smetnje pojave unutar 50 cm od telefona. Ovaj se sustav još proširuje i na pametne satove gdje postiže točnost praćenja 1,2 cm za 2D praćenje. LLAP zvučnicima emitira kontinuirani signal frekvencije 17-23 kHz, a mikrofoni bilježe refleksije od ruke korisnika. Za mjerenje udaljenosti ruke koristi fazne promjene (Dopplerov pomak) u zvučnim signalima. Konkretno, koristi se I/Q demodulator kako bi se dobio kompleksni signal, a potom se on dijeli na statički i dinamički vektor. Statički vektor proizlazi iz statičnih objekata, a dinamički vektor od pokreta ruke. Nakon toga se informacija o fazi signala pretvara u informaciju o udaljenosti s obzirom na dinamički vektor kako bi se postiglo 1D mjerenje udaljenosti. Uz 1D, postiže se i 2D praćenje gesti ruku (npr. crtanje oblika ili riječi) kombiniranjem mjerenja faze i

kašnjenja. Kada se ruka pomakne za 10 cm na udaljenosti od 20 cm od telefona, srednja pogreška udaljenosti je 3,5 mm. Ovaj sustav može pouzdano mjeriti udaljenost brzinom od 4 do 25 cm/s i otporan je na okolni šum. Nadalje, postiže pogrešku praćenja od 4.57 mm, točnost prepoznavanja 26 latiničnih znakova od 92.3%, odnosno točnost prepoznavanja 11 riječi (npr. da, može) od 91,2%. Što se tiče odziva sustava, njegovo kašnjenje je manje od 15 ms. Međutim, LLAP može realizirati praćenje samo jednog objekta. Tako prste i šaku vidi kao jedan objekt i ne može raspoznati pokrete s više prstiju. AudioGest [4] koristi pametni telefon za detekciju gesti ruku. Ovaj sustav ne samo da može klasificirati geste ruku, već može i identificirati brzinu pokreta ruke, opseg pokreta ruke i vrijeme trajanja izvođenja geste, povećavajući tako mogućnosti interakcije. Strata [12] također koristi pametni telefon za praćenje gesti na temelju ultrazvučnog signala. Prvo koristi zvučnik za prijenos zvučnog signala frekvencije 18-22 kHz, a zatim mikrofonima bilježi reflektirani signal. Strata uzima u obzir višestazne efekte, prateći putanje ruku koristeći procjenu kanalnog impulsnog odziva (engl. *Channel Impulse Response, CIR*). Nakon procjene CIR-a iz zabilježenog signala, promjene relativne udaljenosti i informacije o apsolutnoj udaljenosti izračunate pomoću CIR-a kombiniraju se kako bi se mogao pratiti pokret ruke. Uz srednju pogrešku od 1,0 cm za 1D praćenje i 1,01 cm za 2D praćenje, pokazuje izvrsnu robusnost, čak i pod uvjetima s puno šuma (npr. glazba u pozadini). Strata također postiže srednju pogrešku prepoznavanja oblika od 0,57 cm i ima nisko kašnjenje od 12.5 ms, što omogućuje praćenje gotovo u stvarnom vremenu.

3. OPIS VLASTITOG RJEŠENJA ZA DETEKCIJU GESTI RUKU

Za izradu sustava za detekciju gesti ruku korisnika korišteni su komercijalni zvučnici (*White Shark 2.0 GSP-619 Beat*) i mikrofoni (*White Shark DSM-02 Nagara*). Zvučnici generiraju kontinuirani sinusni signal frekvencije 20 kHz, a dva mikrofona istovremeno snimaju signal reflektiran od ruke korisnika koji, zbog pojave Dopplerovog efekta, ima promjenu u frekvenciji i sada više nije čisti signal frekvencije 20 kHz, nego se nalazi u području od 19,6 do 20,4 kHz. Snimljeni audio signali se potom pretvaraju u jedan stereo signal na koji se primjenjuje kratkotrajna Fourierova transformacija (engl. *Short-Time Fourier Transform, STFT*) kojom se signal transformira iz vremenske u frekvencijsku domenu i dobiva se spektrogram signala. Dobiveni spektrogram predstavlja ulaz u konvolucijsku neuronsku mrežu (engl. *Convolutional Neural Network, CNN*) koja će vršiti detekciju gesti ruku korisnika. Detektirana gesta pokreće odgovarajuću funkciju za kontrolu *Spotify Web* aplikacije. Slika 3.1. prikazuje način na koji ovaj sustav funkcionira.



Slika 3.1. *Prikaz načina rada sustava za detekciju gesti ruku*

Razvoj sustava može biti podijeljen u tri glavna koraka: kreiranje baze podataka, obradu signala i treniranje neuronske mreže. Prvo je potrebno definirati geste koje će sustav moći raspoznavati. Da bi se geste dobro definirale, ključno je da budu jednostavne i dovoljno različite te smislene za definiranu primjenu: u ovom slučaju će to biti upravljanje *Spotify* glazbenim servisom.

Nakon što su definirane geste koje će sustav koristiti, potrebno je izraditi bazu podataka koja će se kasnije koristiti za trening neuronske mreže. Pri tome je važno da se u bazi podataka

nalazi dovoljan broj uzoraka za svaku gestu kako bi se osigurala visoka točnost i preciznost modela u stvarnim uvjetima. Također, bitno je da su podaci odgovarajućeg oblika za ulaz u neuronsku mrežu. CNN kao ulaz prima sliku, stoga je bitno prvo obraditi signale kako bi se iz sirovog audio zapisa dobila slika spektra signala.

Nakon što je definirana struktura neuronske mreže i kreirana baza podataka, slijedi trening neuronske mreže. Pri tome je korištena metoda nadziranog učenja (engl. *supervised learning*) gdje model radi s poznatim skupom ulaznih i izlaznih podataka i primijenjen je jedan od postupaka nadziranog učenja – klasifikacija. Ulazni podaci su u ovom slučaju uzorci iz baze podataka, a izlazni podaci su unaprijed definirane klase u koje mreža kasnije razvrstava podatke.

Na kraju je potrebno povezati istrenirani model za detekciju gesti ruku sa *Spotify* servisom da bi se testirao rad sustava u stvarnom vremenu. Ako se gesta detektira, pokreće se odgovarajuća funkcija za kontrolu *Spotify Web* aplikacije.

3.1. Odabir gesti

Odabrano je ukupno 11 gesti koje će sustav moći klasificirati: *gore, dolje, lijevo, desno, prema, beskonačno, krug u smjeru kazaljke na satu, krug u suprotnom smjeru od kazaljke na satu, shuffle, zatvorena šaka* i *ništa*. Način izvođenja gesti je prikazan u prilogu P.3.1. Pri odabiru gesti vodilo se računa da budu lako izvedive, međusobno dovoljno različite, smislene i intuitivne za odabranu primjenu – upravljanje *Spotify* glazbenim servisom.

Ukratko, gestama *gore, dolje, prema* i *zatvorena šaka* upravlja se reprodukcijom zvuka, gestama *lijevo* i *desno* se upravlja reprodukcijom pjesama u nizu, a gestama *beskonačno, krug u smjeru kazaljke na satu, krug u suprotnom smjeru od kazaljke na satu* i *shuffle* se upravlja trenutnim popisom pjesama.

Izvođenjem geste *gore*, korisnik pojačava glasnoću zvuka reprodukcije glazbe u *Spotify Web* aplikaciji za 10%. Gesta *dolje* joj je suprotna pa se njenim izvođenjem smanjuje glasnoća za 10%. Geste *lijevo* i *desno* služe za upravljanje redoslijedom reprodukcije, pa tako gesta *lijevo* reproducira prethodnu pjesmu na listi, a gesta *desno* sljedeću. Gesta *prema* predstavlja početak i kraj reprodukcije (ako je pjesma trenutno u aktivnoj fazi izvođenja, izvršavanjem ove geste reprodukcija se zaustavlja i obrnuto). Izvođenjem geste *beskonačno*, popis za reprodukciju pjesama se izvodi ispočetka jednom kada se dođe do kraja popisa, dok se izvođenjem geste *shuffle* pjesme izvode nasumičnim redoslijedom. Gesta *krug u suprotnom smjeru od kazaljke na satu*

reproducira trenutnu pjesmu ponovno od početka, a gesta *krug u smjeru kazaljke na satu* neprestano ponavlja reprodukciju iste pjesme. Gesta *zatvorena šaka* ima funkciju isključivanja i ponovnog uključivanja zvuka (engl. *mute/unmute*). Konačno, gesta *ništa* kontrolna je gesta. Kada model prepozna ovu gestu, on neće pokrenuti niti jednu od funkcija za upravljanje *Spotify* servisom, čime se smanjuje mogućnost lažnih detekcija kada se korisnik nalazi ispred sustava, a ne izvršava niti jednu od gesti namijenjenih za kontrolu *Spotify*-ja. U tablici 3.1. prikazane su funkcije za kontrolu *Spotify Web* aplikacije koje pojedine geste pokreću.

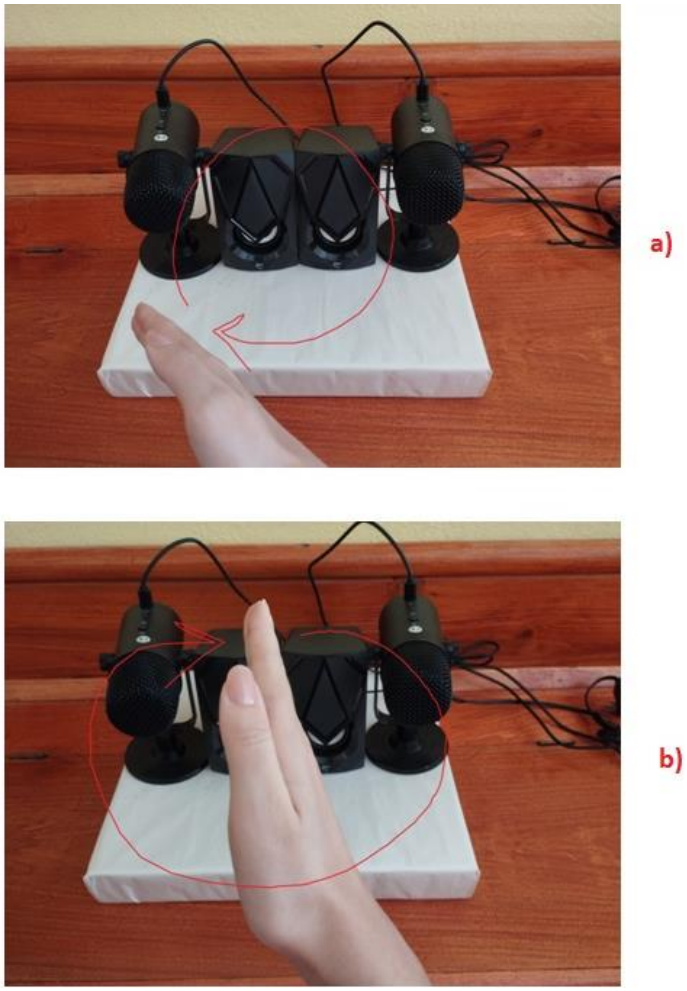
***Tablica 3.1**. Geste i funkcije koje pokreću*

gore	pojačava glasnoću zvuka reprodukcije glazbe za 10%
dolje	smanjuje glasnoću zvuka reprodukcije glazbe za 10%
zatvorena šaka	isključivanje/uključivanje zvuka
lijevo	reproducira prethodnu pjesmu na listi
desno	reproducira sljedeću pjesmu na listi
prema	početak/kraj reprodukcije
beskonačno	popis za reprodukciju pjesama se izvodi ispočetka
shuffle	pjesme se izvode nasumičnim redoslijedom
krug u suprotnom smjeru od kazaljke na satu	reproducira trenutnu pjesmu ponovno od početka
krug u smjeru kazaljke na satu	neprestano ponavlja reprodukciju iste pjesme
ništa	kontrolna gesta

3.2. Kreiranje baze podataka

S obzirom da ne postoje javno dostupne baze podataka, napravljena je vlastita. Za potrebe treniranja CNN mreže, napravljene su tri baze, od čega su dvije korištene za treniranje mreže, a jedna za konačno testiranje rada mreže u stvarnom vremenu. Baze korištene za treniranje obje sadrže uzorke za ukupno 11 gesti te su iste geste snimane s različitim načinom izvođenja kako bi sustav bio istreniran na što raznovrsnijim podacima i time bolje primjenjiv u stvarnim uvjetima. Na primjer, određeni broj gesti je izveden lijevom rukom, a ostatak desnom. Zatim, za geste *krug u smjeru kazaljke na satu*, *krug u suprotnom smjeru od kazaljke na satu* i *beskonačno* početna točka izvođenja geste je različita pa je npr. u jednom uzorku gesta započeta iz donjeg lijevog kuta,

a u drugom uzorku iz centra itd. Sve geste su snimane na udaljenosti između 5 i 20 cm od mikrofona te su izvođene različitom brzinom i širinom. Na slici 3.2. se nalazi primjer izvođenja geste *krug u smjeru kazaljke na satu* sa dvije različite početne točke.



Slika 3.2. *Prikaz izvođenja geste krug u smjeru kazaljke na satu: a) početna točka je donji lijevi kut, b) početna točka je centar*

Prva baza podataka, korištena za trening mreže, sastoji se od ukupno 1100 uzoraka, po 100 uzoraka za svaku gestu te ih je snimila ista osoba.

Druga baza podataka je služila za prvo testiranje rada mreže pri ponovnom treningu. Uzorke je snimilo 10 različitih osoba. Svaka osoba je izvodila geste 10 puta, a kontrolna gesta *ništa* se nije ponovno izvodila, nego je preuzeta iz prve baze podataka što ukupno čini 1100 uzoraka. Zatim su obje baze spojene u jednu, što ukupno čini 2100 uzoraka za drugo testiranje mreže pri ponovnom treniranju.

Na kraju je napravljena posebna baza podataka za konačno testiranje i evaluaciju rada mreže koju je snimilo 5 novih osoba. Svaka osoba je snimila 10 uzoraka po gesti, što ukupno čini 550 testnih uzoraka.

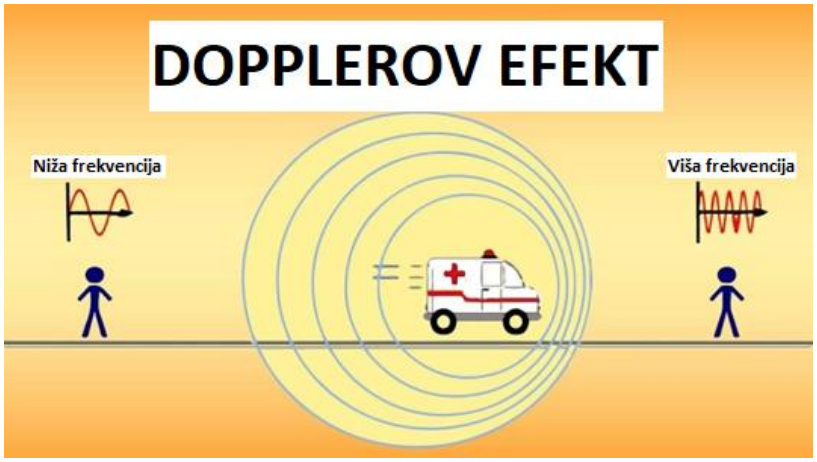
3.3. Utjecaj okoline

3.3.1. Dopplerov efekt

Sustav za detekciju gesti ruku baziran je na pojavi Dopplerovog efekta. Ovaj efekt se očituje promjenom frekvencije izvornog zvučnog signala uslijed gibanja izvora i točke promatranja, ili, u slučaju statičnog izvora i točke promatranja, zbog refleksije od površine objekta u pokretu. Nova frekvencija će biti viša ili niža od izvorne ovisno o brzini i smjeru gibanja ruke od mikrofona. Razlika između dviju frekvencija se naziva Dopplerov pomak. U uvjetima kada nema refleksije zvučnog vala od neke površine, nego se izvor zvuka ili točka promatranja gibaju relativno jedno u odnosu na drugo, Dopplerov pomak se računa na sljedeći način:

$$f_0 = f_s \frac{v + v_s}{v - v_0} \qquad (3-1)$$

gdje su f_0 i f_s frekvencije točke promatranja i izvora, a v, v_s i v_0 su brzine zvuka, izvora i točke promatranja [13], [14]. Klasičan primjer je sirena kola hitne pomoći koja se giba relativno u odnosu na osobu koja je statična. Osoba čuje Dopplerov efekt kao promjenu u visini tona sirene u vidu povišenja tona kada joj se kola približavaju (što predstavlja povećanje frekvencije), a potom sniženju kada se kola udaljavaju od nje (frekvencija signala se smanjuje). Ovo je prikazano na slici 3.3.

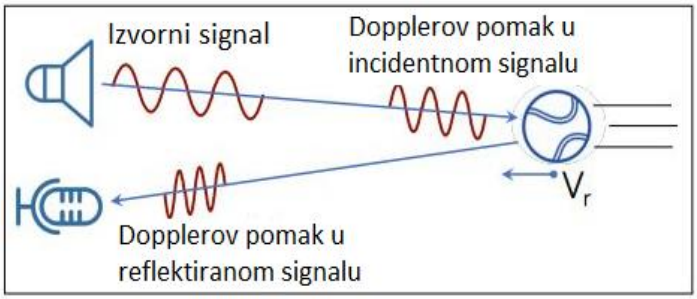


Slika 3.3. *Dopplerov efekt u slučaju statične točke promatranja i izvora zvuka koji se giba [15]*

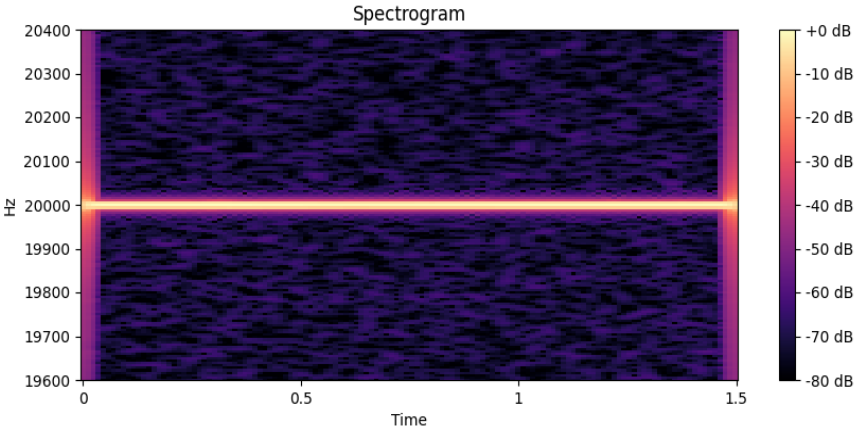
S druge strane, kada su i izvor (zvučnik) i točka promatranja (mikrofon) statični, mikrofon može opaziti Dopplerov pomak ukoliko se u blizini kreće objekt od kojeg će se zvučni val sa izvora reflektirati. Taj objekt se onda ponaša kao izvor signala pa dvaput dolazi do pomaka u frekvenciji: jednom dok signal putuje od izvora do reflektora, a zatim dok putuje od reflektora do mikrofona. U ovom slučaju će se Dopplerov pomak onda odrediti kao:

$$\Delta f = f_s \frac{2v_r}{v - v_r} \approx f_s \frac{2v_r}{v}, \qquad v_r \ll v \tag{3-2}$$

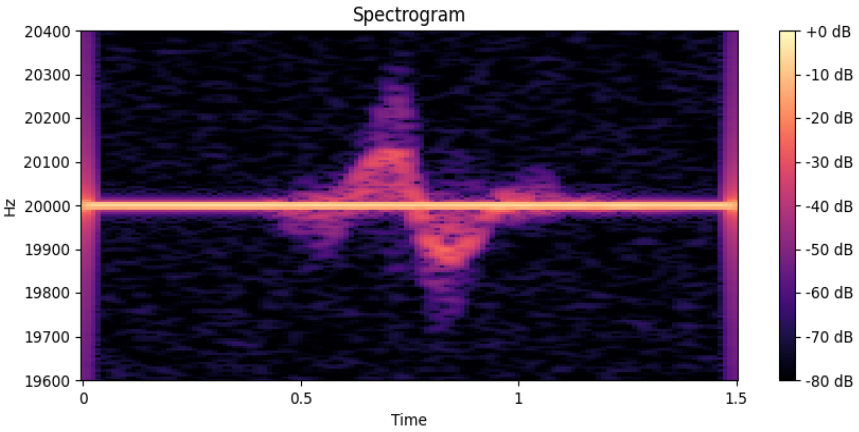
pri čemu je v_r brzina reflektora [13]. Ovaj slučaj je prikazan na slici 3.4. i upravo je na takvoj vrsti Dopplerovog efekta baziran sustav za detekciju gesti ruku: reflektor predstavlja ruku korisnika, izvor zvuka je zvučnik, a točka promatranja je mikrofon. Slike 3.5. i 3.6. prikazuju spektrograme izvornog signala i reflektiranog signala gdje je jasno vidljiv pomak u frekvenciji u vremenu u odnosu na izvorni signal.



Slika 3.4. *Dopplerov pomak frekvencije izvornog signala uslijed gibanja reflektora, pri čemu su zvučnik i mikrofon statični [13]*



Slika 3.5. *Spektrogram izvornog signala*

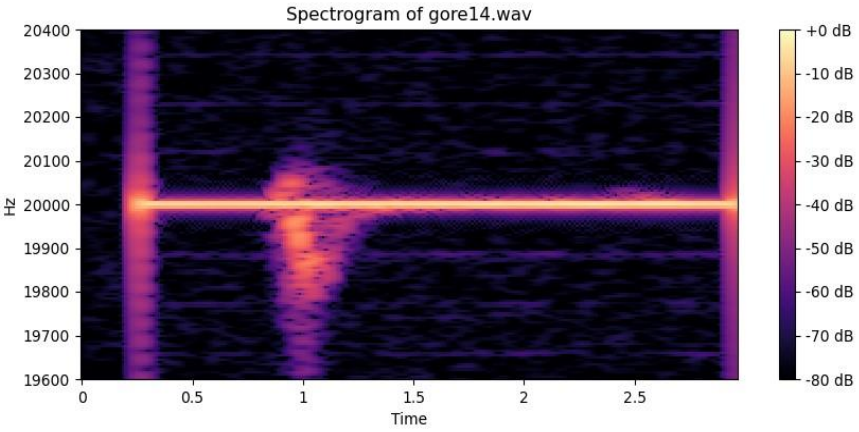


Slika 3.6. *Spektrogram reflektiranog signala, vidljiv je Dopplerov pomak*

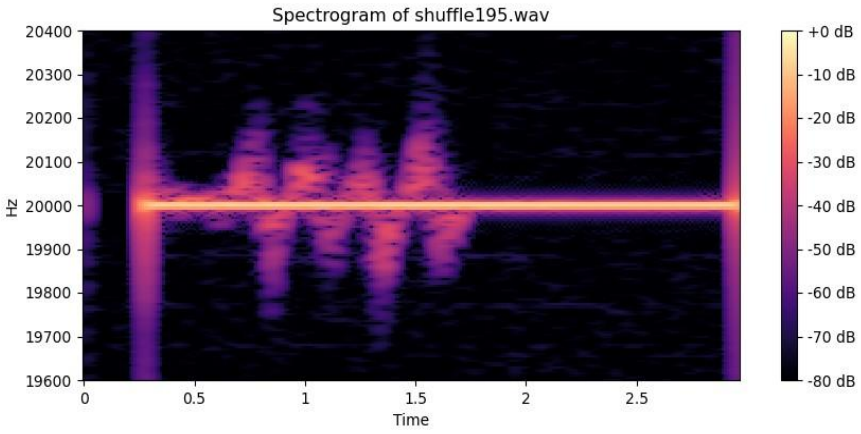
3.3.2. Utjecaj okoline na kvalitetu snimljenog signala

Čujno područje ljudskog uha je u rasponu frekvencija od 16 Hz do 20 kHz. Pri tome je zvuk frekvencije niže od 16 Hz u infrazvučnom području, a zvuk od 20 kHz pa na više ulazi u ultrazvučno područje [16]. Budući da se detekcija gesti ruku vrši na temelju zvuka, on mora biti nečujan kako ne bi ometao aktivnosti korisnika. S obzirom da je gornja granica čujnosti u prosječne odrasle osobe oko 15 do 17 kHz [17], za detekciju se stoga koristi ultrazvučni val koji je modeliran kao čisti sinusni signal frekvencije 20 kHz generiran na izvoru.

Kad korisnik izvede neku od gesti, generirani zvučni val se odbija od ruke korisnika i zbog Dopplerovog efekta dolazi do promjene frekvencije koja je sada u rasponu od 19,6 do 20,4 kHz. Svaka gesta koju korisnik napravi će uzrokovati drugačiju refleksiju te ima jedinstveni frekvencijski potpis na temelju kojeg će ih mreža moći razlikovati. Na slikama 3.7. i 3.8. je prikazan primjer potpisa dvije različite geste.

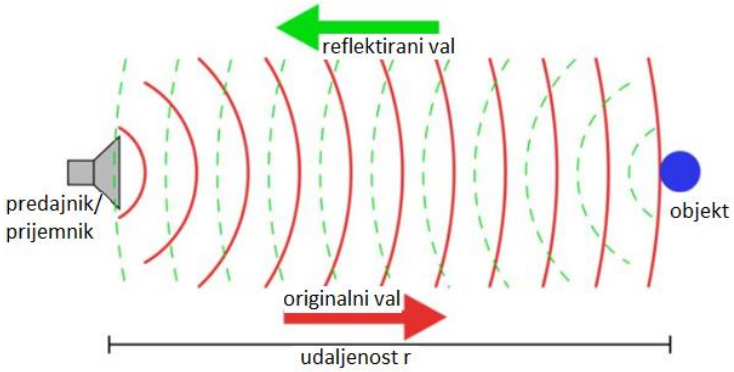


Slika 3.7. *Spektrogram geste gore*



Slika 3.8. *Spektrogram geste shuffle*

Reflektirani val bilježe dva mikrofona, Kada se zvučni val odbije od površinu ruke, reflektirani val putuje u suprotnom smjeru od smjera širenja zvučnog vala s izvora (prikazano na slici 3.9.).



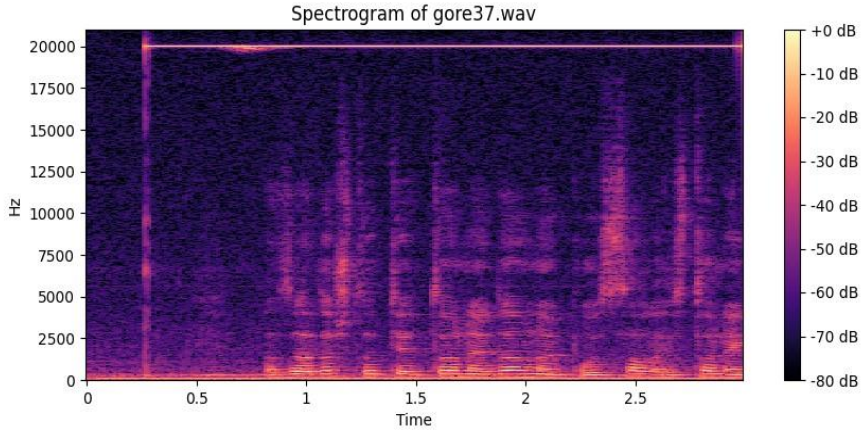
Slika 3.9. *Prikaz smjera širenja vala s izvora i reflektiranog vala [18]*

Stoga, da bi mikrofoni mogli zabilježiti refleksiju, moraju biti usmjereni u smjeru širenja vala s izvora, tj. trebaju biti okrenuti na istu stranu gdje su okrenuti zvučnici. Do oba mikrofona dolazi malo drugačija refleksija s obzirom da se nalaze na različitim mjestima u prostoru i usmjereni su pod različitim kutom u odnosu na izvor refleksije (ruke korisnika). Upravo zato što se signal različito reflektira po prostoru, jako je bitno da oprema bude fiksirana tako da su mikrofoni uvijek u istom položaju jedan u odnosu na drugi kao i u odnosu na zvučnike da bi se izbjegla pogreška pri detekciji. Na primjer, ako korisnik dvaput zaredom izvede gestu *lijevo,* ali su pri prvom izvođenju mikrofoni postavljeni pod jednim kutom, a pri drugom izvođenju pod malo drugačijim kutom, premda će mikrofoni u oba slučaja zabilježiti refleksiju za istu gestu, refleksije se mogu razlikovati pa će imati drugačiju sliku spektra. To u konačnici može navesti mrežu da istu gestu

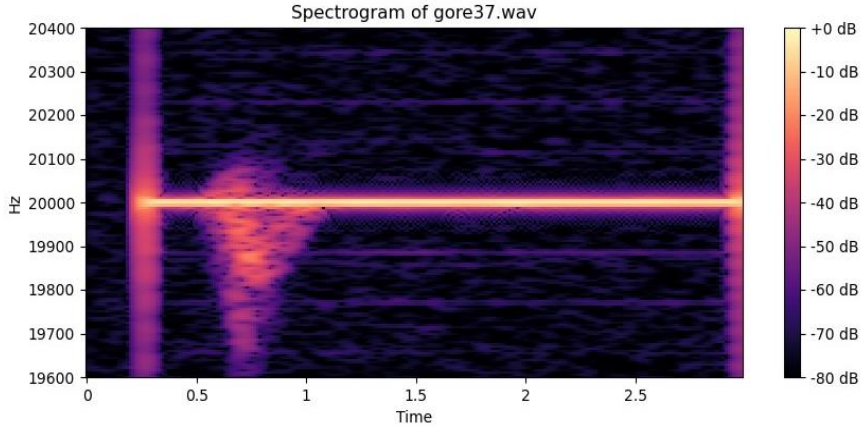
pogrešno klasificira kao dvije različite. Upravo iz tog razloga nema javno dostupnih baza podataka jer snimljene refleksije jako ovise o položaju postavljene opreme.

Pored konzistentnosti u podacima prilikom snimanja reflektiranih signala, bitno je postaviti opremu na način da se što više ograniče neželjeni vanjski utjecaji kao što su refleksije od okolnih objekata i šum.

Šum mogu biti razni uređaji koji se nalaze u blizini sustava za detekciju gesti kao što je na primjer rad računala ili klima uređaja u uredskom prostoru, kao i razgovor i zvuk prometa ili prirode u blizini. Takav šum se u većoj mjeri eliminirao ograničavanjem spektrograma na raspon frekvencija od 19,6 do 20,4 kHz. Na slici 3.10. je prikazan spektrogram zvučnog signala na cijelom frekvencijskom području gdje se jasno vidi šum. Šum na nižim frekvencijama se dogodio zbog rada računala i razgovora, a šum koji se javlja i na višim frekvencijama je samo rad računala. Slika 3.11. prikazuje spektrogram s ograničenim rasponom frekvencija.

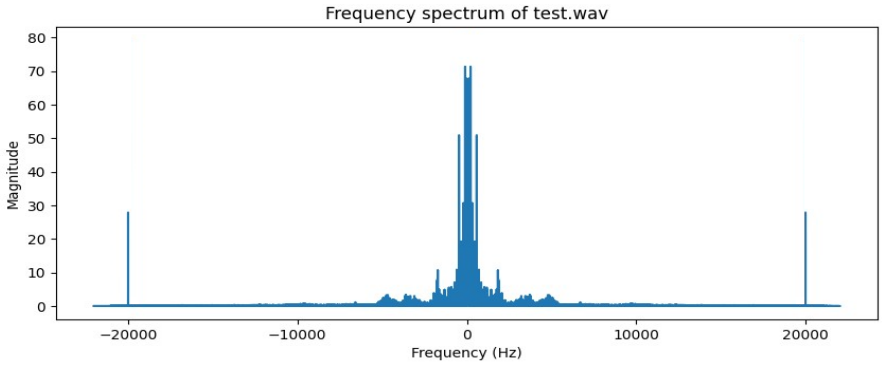


Slika 3.10. *Spektrogram zvučnog signala prikazuje cijelo frekvencijsko područje – jasno je vidljiv šum pri nižim frekvencijama*

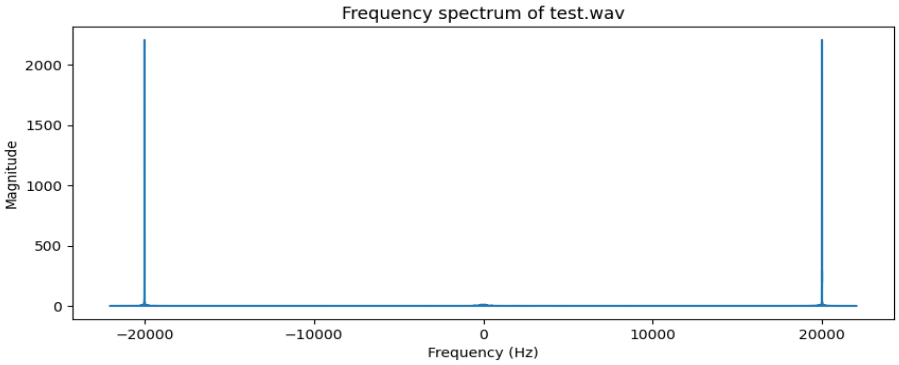


Slika 3.11. *Spektrogram zvučnog signala ograničen na frekvencije između 19,6 i 20,4 kHz*

Ukoliko se sustav postavi blizu većih prepreka, kao što su zid ili kućište računala, dolazi do neželjenih refleksija signala od površine prepreke. To uzrokuje propade u magnitudi reflektiranog signala jer su neželjene refleksije iz okoline prejake pa magnituda željenog signala (20 kHz) postaje premala u odnosu na šum. Osim što je jakost signala dosta smanjena (vidljivo na slici 3.12.), sustav postaje jako nestabilan i nepouzdan ukoliko mu se magnituda neprestano mijenja i nije relativno konstantna. Također, sam položaj mikrofona u odnosu na zvučnike jako utječe na magnitudu reflektiranog signala. Putem metode pokušaja i pogrešaka došlo se do položaja mikrofona u odnosu na zvučnike u kojem ne dolazi do propada magnitude u vremenu, nego je magnituda reflektiranog signala zadovoljavajuće jakosti i relativno konstantna u vremenu. Na slici 3.13. prikazan je frekvencijski spektar sa zadovoljavajućim magnitudama signala: magnituda 20 kHz je dovoljno jaka u odnosu na okolni šum. Na slici 3.14. je prikazan konačni, fiksirani položaj opreme. Oprema je fiksirana na takvu podlogu da zvučnici i mikrofoni uvijek budu na međusobno istom položaju, a da je sustav ujedno lako prenosiv.



Slika 3.12. *Prikaz frekvencijskog spektra signala nezadovoljavajuće magnitude*



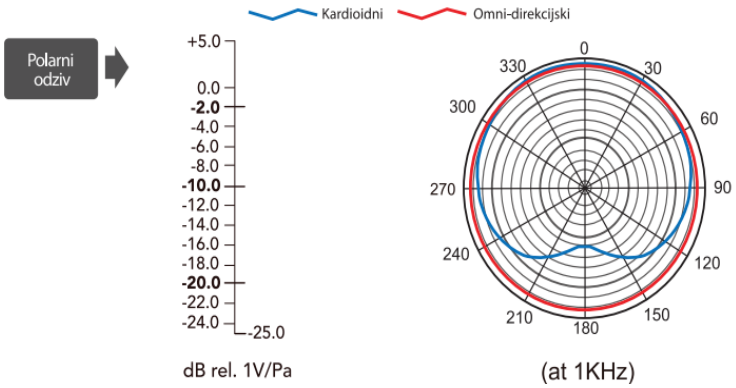
Slika 3.13. *Prikaz frekvencijskog spektra signala zadovoljavajuće magnitude*

Slika 3.14. *Prikaz hardverskog dijela sustava za detekciju gesti na temelju zvuka*

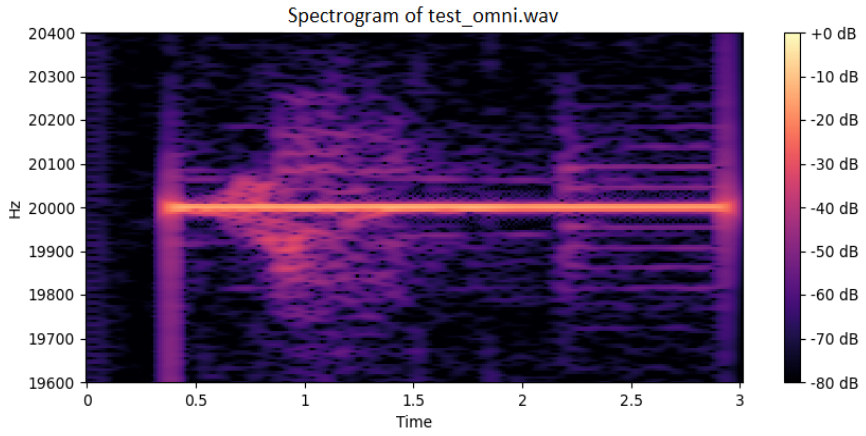
Polarizacija mikrofona također jako utječe na snagu reflektiranih signala kao i na razinu šuma. Korišteni mikrofoni imaju dva načina rada: kardioidni i omni-direkcijski.

Kod omni-direkcijske polarizacije mikrofoni bilježe zvučne valove iz svih smjerova jednako što može biti korisno kada ne znamo iz kojeg će točno smjera doći zvuk od interesa.

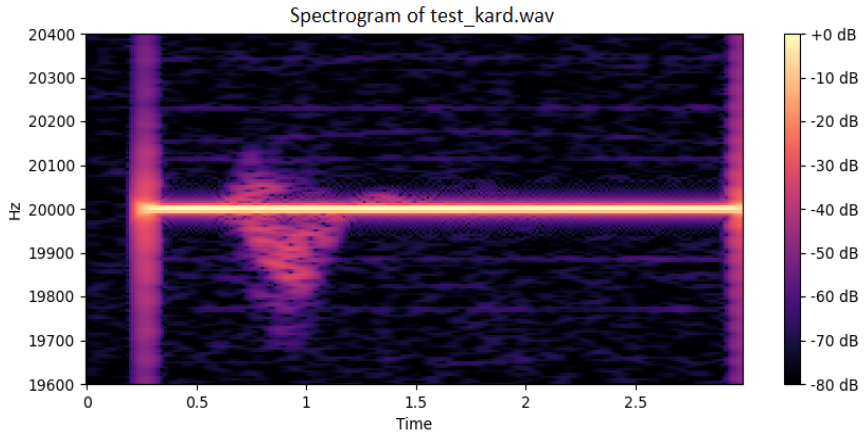
Kod kardioidne polarizacije mikrofon je osjetljiviji u određenim pravcima. Najveća osjetljivost mikrofona je točno ispred kapsule mikrofona, odnosno na 0°, gotovo uopće nije osjetljiv na zvuk koji dolazi točno iza mikrofona, na 180°, i smanjene je osjetljivosti na zvuk koji dolazi sa strane, odnosno iz smjerova 90° i 270°. Zbog toga, mikrofon u ovom načinu rada puno lakše i bolje uklanja pozadinski šum i refleksiju iz prostorije u odnosu na omni-direkcijski način rada, a usmjeravanjem mikrofona u smjeru izvora reflektiranog zvuka, pojačava mu se snaga i time se poboljšava kvaliteta snimljenog reflektiranog signala [19]. Zato je pri snimanju reflektiranih signala korištena kardioidna polarizacija na oba mikrofona. Slika 3.15. prikazuje polarizacije korištenih mikrofona, a slike 3.16. i 3.17. spektrograme signala snimljenih sa dvije različite usmjerenosti mikrofona gdje se jasno vidi razlika u kvaliteti snimljenih reflektiranih signala.



Slika 3.15. *Polarizacija mikrofona [20]*

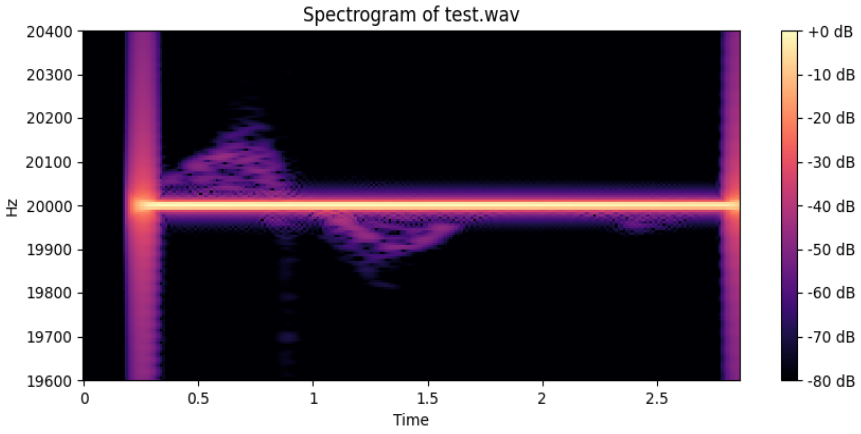


Slika 3.16. *Spektrogram reflektiranog signala u omni-direkcijskom načinu rada mikrofona*

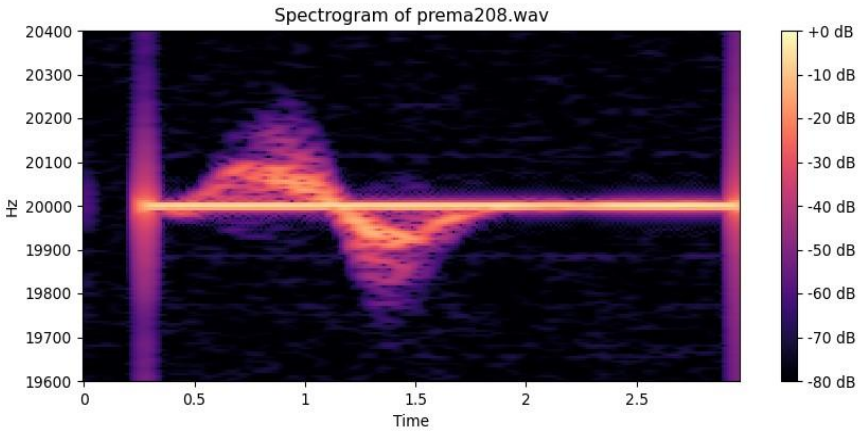


Slika 3.17. *Spektrogram reflektiranog signala u kardioidnom načinu rada mikrofona*

Međutim, unatoč fiksiranju opreme u zadovoljavajući položaj, postavljanju sustava u relativno slobodni prostor da se odmakne od većih fiksnih prepreka u blizini i usmjeravanju mikrofona, ponekad šum i neželjene refleksije nije moguće izbjeći jer su mikrofoni jako osjetljivi i zabilježit će sve zvučne signale koji su mu u blizini. Primjer takve situacije je kada korisnik sam uzrokuje neželjene refleksije na način da ispred mikrofona izvede radnju koja nije jedna od definiranih gesti, kao što je npr. posezanje za objektom u blizini. Zbog toga je bitno da kada sustav aktivno snima zvuk radi detekcije, korisnik ne radi previše nepotrebnih pokreta ispred sustava jer te pokrete mreža može interpretirati kao jednu od korisnih gesti i pokrenuti odgovarajuću funkciju za upravljanjem *Spotify*-jem, što je najveći nedostatak ovakvog sustava. Slika 3.18. prikazuje neželjenu refleksiju: korisnik poseže za olovkom koja se nalazi ispred mikrofona tijekom aktivnog snimanja zvuka, što bi sustav mogao interpretirati kao gestu *prema* s obzirom na jako veliku sličnost te dvije slike signala. Slika signala geste *prema* prikazana je na slici 3.19.



Slika 3.18. *Spektrogram neželjenog reflektiranog signala*



Slika 3.19. *Spektrogram geste prema*

3.4. Obrada signala

Neuronska mreža koju koristi sustav za detekciju gesti ruku kao ulaz prima sliku. S obzirom da zvuk u svom sirovom obliku nije slika, treba izvršiti nekoliko računalnih operacija da uzorci signala budu odgovarajućeg oblika za ulaz u mrežu.

Prvo, svaki mikrofon istovremeno snima monokanalni zvučni zapis koji se zatim kombinira u jedan stereo zapis čime sustav dobiva percepciju dubine prostora. To omogućuje lakše razlikovanje komplementarnih gesti kao što su geste *lijevo/desno, gore/dolje, krug u smjeru/suprotnom smjeru od kazaljke na satu.* Na primjer, za gestu *lijevo* se zna iz kojeg smjera dolazi jer će se refleksija prvo pojaviti na desnom pa onda lijevom mikrofonu, a za gestu *desno* će situacija biti obrnuta. Da bi ovo bilo moguće, mikrofoni su blago usmjereni jedan prema drugome.

Tijekom snimanja zvučnog signala, mikrofon bilježi vremenski kontinuirani signal, međutim računalo ne može obraditi takav signal pa ga je potrebno diskretizirati prije pohranjivanja u memoriju. Prilikom digitalizacije ovakvog signala, vremenski kontinuirani analogni signal se dijeli na konačan broj uzoraka u procesu analogno-digitalne (A/D) pretvorbe koju provodi zvučna kartica računala. Pritom, zvučna kartica definira frekvenciju uzorkovanja signala koja određuje vremensku rezoluciju snimljenog signala i predstavlja broj diskretnih uzoraka po sekundi trajanja signala. Također, definira dubinu bita (engl. *bit depth*) koja utječe na dinamički raspon snimljenog signala kao i vrstu modulacije kojom se kodiraju i komprimiraju podaci prilikom spremanja signala u memoriju.

Da bi digitalizacija signala bila uspješno provedena, važno je da je frekvencija uzorkovanja barem dvostruko veća od najviše frekvencijske komponente izvornog analognog signala. Drugim riječima, samo signali čije su najviše frekvencije manje od polovice frekvencije uzorkovanja mogu se točno digitalizirati. Ova maksimalna frekvencija signala se zove Niquistova frekvencija, a pravilo je Niquistov teorem. [21] Signali iznad Nyquistove frekvencije nisu ispravno snimljeni i njihove frekvencije se preslikavaju nazad preko Niquistove i tako uvode umjetne frekvencije u procesu koji se naziva preslikavanje frekvencija (engl. *aliasing*). Da bi se preslikavanje izbjeglo, A/D pretvaračima često prethode niskopropusni filtri kojima se guše frekvencije iznad Niquistove prije nego što zvuk stigne do pretvarača.

Dubina bita određuje koliko amplitudnih vrijednosti sadrži svaki vremenski diskretni uzorak definiran frekvencijom uzorkovanja. Dubina bita i frekvencija uzorkovanja onda zajedno određuju rezoluciju zvuka. Što je dubina veća, to se bilježi veći broj amplitudnih vrijednosti po uzorku što se zatim koristi za rekonstrukciju izvornog audio signala tijekom A/D pretvorbe. Veća dubina bita znači bolju rezoluciju i veći broj mogućih amplitudnih vrijednosti, stoga je amplituda kontinuiranog analognog signala bliža amplitudama korištenim prilikom digitalizacije signala čime digitalna aproksimacija amplitude postaje bliža izvornom kontinuiranom signalu. S druge strane, ako je dubina premala doći će do gubitka informacija iz izvornog audio signala.

Korištena frekvencija uzorkovanja je 44.1 kHz, dubina je 16 bita, a modulacija je pulsno-kodna (engl. *Pulse Code Modulation, PCM*).

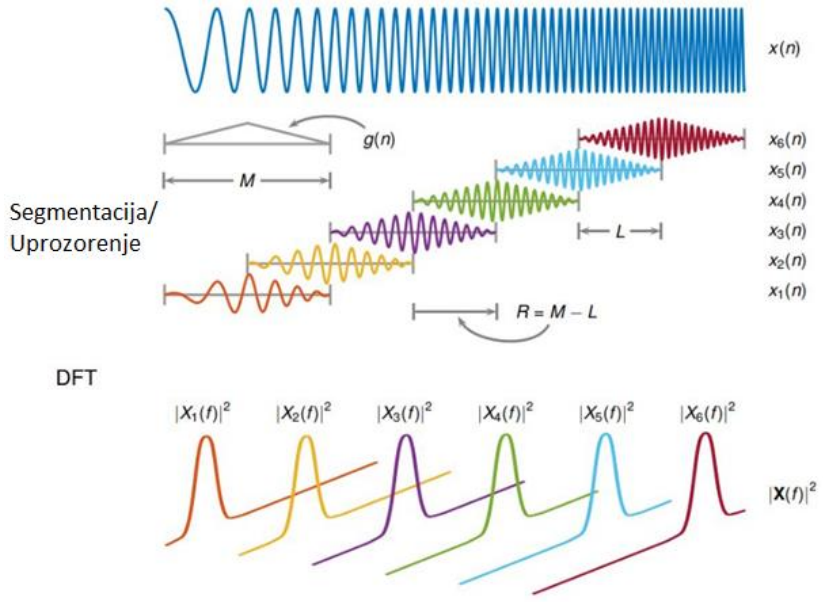
3.4.1. Kratkotrajna Fourierova transformacija

Nakon pretvorbe u stereo signal, primjenjuje se kratkotrajna Fourierova transformacija kojom se signal transformira iz vremenske u frekvencijsku domenu i dobiva se slika frekvencijskog spektra signala – spektrogram.

STFT signala se računa pomicanjem prozora za analizu (engl. *analysis window*) $g(n)$ veličine M preko izvornog signala i izračunavanjem diskretne Fourierove transformacije (engl. *Discrete Fourier transform, DFT*) za svaki segment uprozorenih podataka (engl. *windowed data*). Pritom je n u izrazu $g(n)$ indeks vremenski diskretnog uzorka signala koji dolazi s mikrofona, a M označava koliko se ovih uzoraka nalazi unutar jednog prozora za analizu, odnosno u jednom koraku STFT transformacije. Prozor za analizu se pomiče po izvornom signalu u intervalima od R uzoraka, što je ekvivalentno $L = M - R$ uzoraka preklapanja između susjednih segmenata. DFT svakog uprozorenog segmenta se dodaje u matricu koja sadrži magnitudu i fazu svake točke u vremenu i frekvenciji i ima sljedeći oblik:

$$k = \left\lfloor \frac{N_x - L}{M - L} \right\rfloor. \qquad (3-3)$$

N_x je duljina signala $x(n)$, M je veličina prozora, a L je broj preklapajućih uzoraka [22]. Slika 3.20. prikazuje ovaj postupak.



Slika 3.20. *prikaz izračuna STFT za signal $x(n)$ [22]*

Zbog ovakvog načina rada, STFT se još naziva i uprozorena Fourierova transformacija [23].

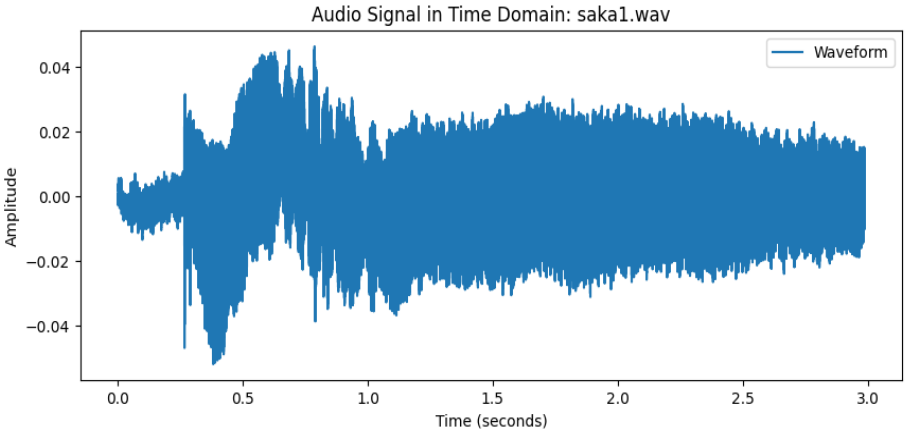
Za izračun STFT sirovog signala, korišteni su sljedeći parametri: veličina prozora, funkcija prozora, frekvencija uzorkovanja i veličina koraka.

Veličina prozora se odnosi na broj uzoraka unutar svakog segmenta signala kada se izvodi Fourierova transformacija. Prilikom izračunavanja STFT, signal se u vremenskoj domeni dijeli na međusobno preklapajuće segmente od kojih se svaki množi sa funkcijom prozora i zatim transformira u frekvencijsku domenu. Veličina prozora utječe na vremensku i frekvencijsku rezoluciju pa tako što je prozor veći, to je frekvencijska rezolucija signala bolja, a vremenska rezolucija lošija. S druge strane, manji prozor poboljša vremensku rezoluciju, a pogoršava frekvencijsku rezoluciju signala.

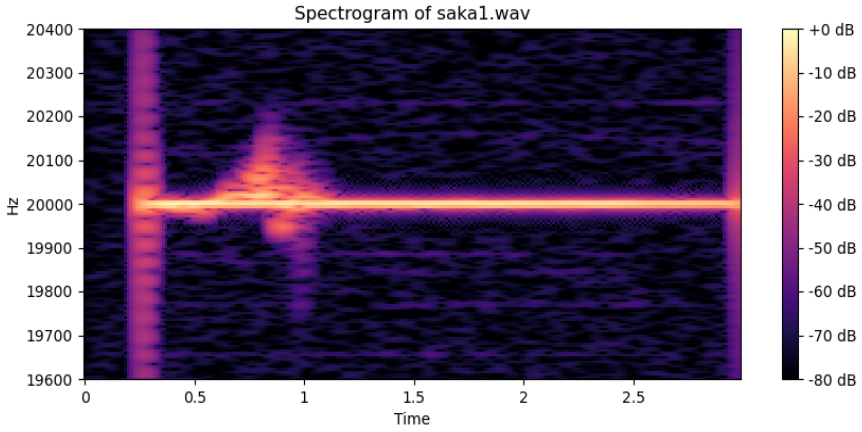
Veličina koraka (engl. *hop length*) se odnosi na broj uzoraka za koje se prozor pomiče u vremenu pri izračunu STFT za idući segment u nizu. Nakon primjene funkcije prozora i izvođenja Fourierove transformacije na jednom segmentu signala, prozor se pomiče na sljedeći segment za definiranu veličinu koraka, a proces se ponavlja. Veličina koraka kontrolira količinu preklapanja između uzastopnih prozora. Tako će manji korak rezultirati većim preklapanjem između susjednih prozora čime se osigurava glađi prikaz odnosa vremena i frekvencije signala, ali se povećava vrijeme obrade signala. S druge strane, veći korak smanjuje preklapanje prozora, što omogućuje bržu obradu signala, ali smanjuje glatkoću spektrograma.

Prilikom izračuna STFT korištena je veličina prozora 8192 da se dobije dobra frekvencijska rezolucija signala. Frekvencija uzorkovanja originalnog signala je 44100 Hz, a korištena veličina koraka je 512 što znači da se prozor pomiče za 512 uzoraka u vremenu za svaki sljedeći segment. Funkcija prozora je Hanning.

Nakon izračuna STFT, spektrogram je ograničen na područje 20000 ± 400 Hz kako bi se smanjila količina podataka s kojom mreža mora raditi i što bolje uklonio pozadinski šum. Sirovi signal prikazan na slici 3.21. sadrži samo vrijednosti amplitude signala u vremenu, dok spektrogram prikazan na slici 3.22. prikazuje frekvencijski spektar signala kao i magnitudu signala u vremenu. Frekvencija i amplituda su prikazane na ordinati, a vrijeme na apscisi. Kod spektrograma je još bojom prikazana magnituda segmenata signala po vremenu na način da što je boja bliža bijeloj, signal je u tom trenutku veće snage.



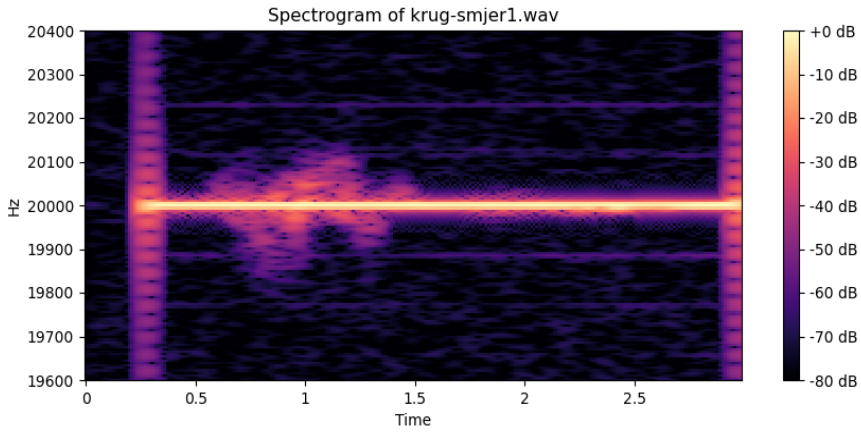
Slika 3.21. *Signal u vremenskoj domeni, prije primjene STFT*



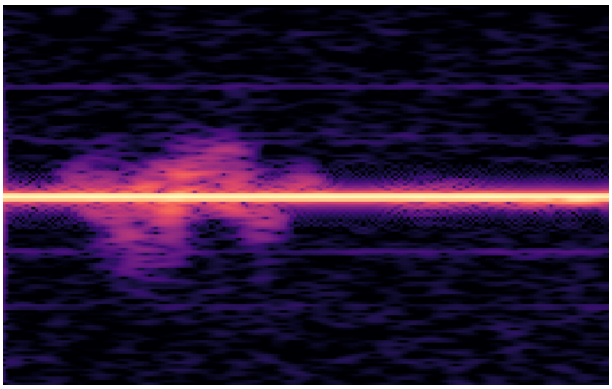
Slika 3.22. *Signal u frekvencijskoj domeni, nakon STFT*

3.4.2. Područje od interesa i sustav boja

S obzirom da se na početku i na kraju spektrograma ponekad pojavi šum zbog ograničenja opreme koji može negativno utjecati na detekciju i tako narušiti točnost modela prilikom treniranja i potom primjene na stvarnim podacima, i s obzirom da se na spektrogramu nalazi vremenska i frekvencijska skala te oznake koje nisu potrebne mreži za učenje i potom detekciju jer su ti podaci uvijek isti, na slici se označava područje od interesa (engl. *Region of Interest, ROI*). Sada, umjesto da se mreži kao ulaz preda cijeli spektrogram sa svim oznakama, mreži će se predati samo manji dio slike na kojem se pojavljuje signal kojeg je potrebno klasificirati. Tako mreža radi sa slikom manjih dimenzija što odmah znači i manji broj parametara s kojima mreža mora raditi čime se optimizira rad sustava za detekciju gesti ruku. Na slici 3.23. je prikazan primjer spektrograma signala sa svim oznakama, a na slici 3.24. definirani ROI istog tog spektrograma.



Slika 3.23. *Potpuni spektrogram signala*

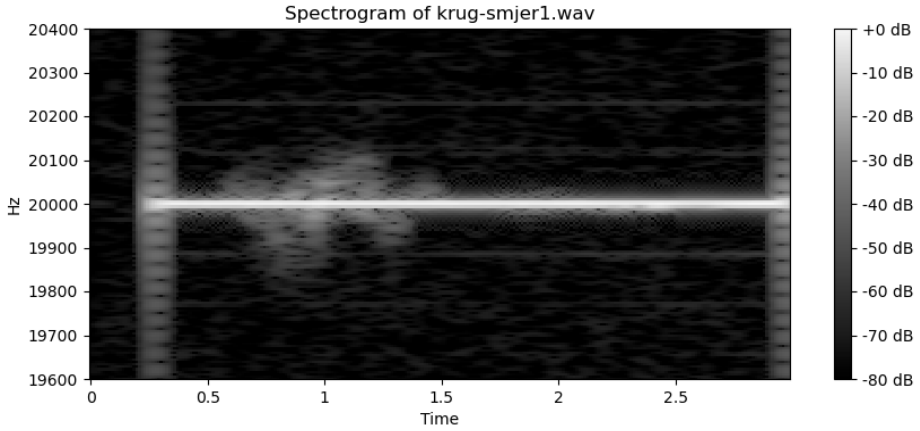


Slika 3.24. *ROI prethodnog spektrograma*

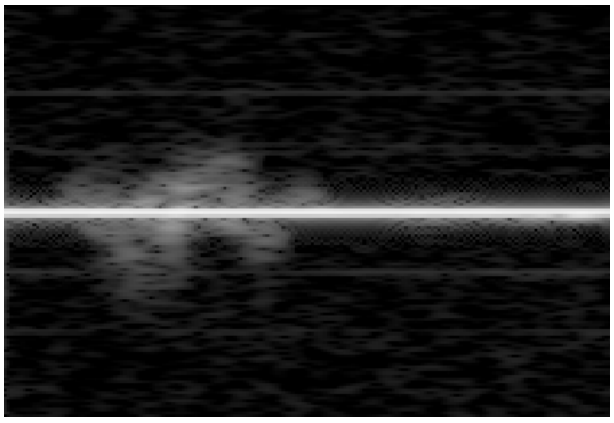
Spektrogram se po zadanim postavkama prikazuje u RGB sustavu boja gdje je slika prikazana pomoću tri 2D matrice s vrijednostima od 0 do 255. Ove vrijednosti označavaju intenzitet crvene, zelene i plave boje za svaki piksel slike [24]. Tako će na primjer za sliku dimenzija 200x200 piksela ukupna količina podataka s kojom mreža radi biti 120000 parametara po slici s obzirom da se množe tri matrice dimenzija 200x200 piksela. To uvelike utječe na brzinu obrade pojedinih slika jer zahtjeva više računalnih resursa u vidu vremena procesiranja i memorije.

S druge strane, ukoliko je slika u tonovima sive boje (engl. *grayscale*), sliku će činiti samo jedna 2D matrica s vrijednostima od 0 do 255 te će ukupan broj parametara s kojima mreža radi biti trostruko manji u odnosu na RGB sliku istih dimenzija. Za potrebe detekcije zvuka na temelju spektrograma, mreži nije potrebna informacija o boji, nego samo o položaju pojedinih značajki prikazanog zvuka. Stoga je na kraju dobiveni spektrogram potrebno konvertirati iz RGB sustava boja u tonove sive boje, čime se smanjuje opterećenje memorije i ubrzava procesiranje svake slike koja mreži dolazi na ulaz, što je ključno u vremenski kritičnim aplikacijama. Slika 3.25. prikazuje

spektrogram signala nakon konverzije u tonove sive boje, a slika 3.26. konačni ulaz u mrežu nakon obrade signala: ROI spektrograma geste u tonovima sive boje. Također je na kraju potrebno prilagoditi dimenzije ROI slike da budu u skladu s definiranim dimenzijama slike koje mreža prima kao ulaz da bi se mreža uspješno mogla istrenirati i kasnije primijeniti na stvarne podatke koji kontinuirano dolaze s ulaza (mikrofona).



Slika 3.25. *Spektrogram signala nakon konverzije u tonove sive boje*



Slika 3.26. *Prikaz ROI spektrograma u tonovima sive boje*

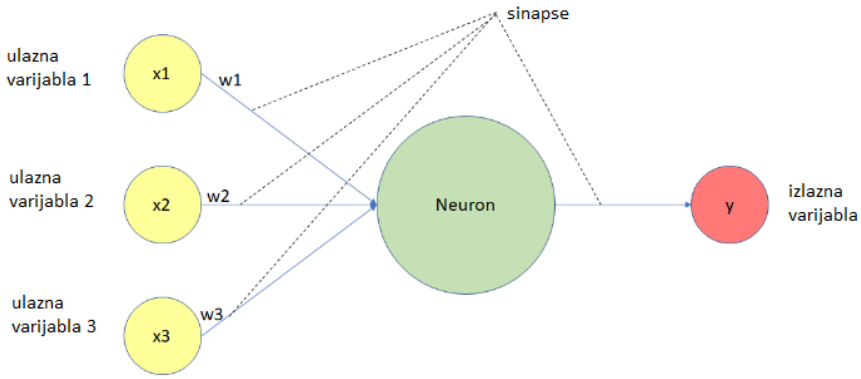
U prilozima P.3.2. do P.3.5. se nalaze kodovi korišteni za generiranje i obradu signala.

3.4. Konvolucijska neuronska mreža

Konvolucijska neuronska mreža vrsta je duboke umjetne neuronske mreže (engl. *Artificial Neural Networks, ANN*) za klasifikaciju i izdvajanje značajki iz skupa podataka koji imaju matrični oblik rešetke, kao što je slika. Svaka ANN se sastoji od skupa povezanih neurona i tri osnovna

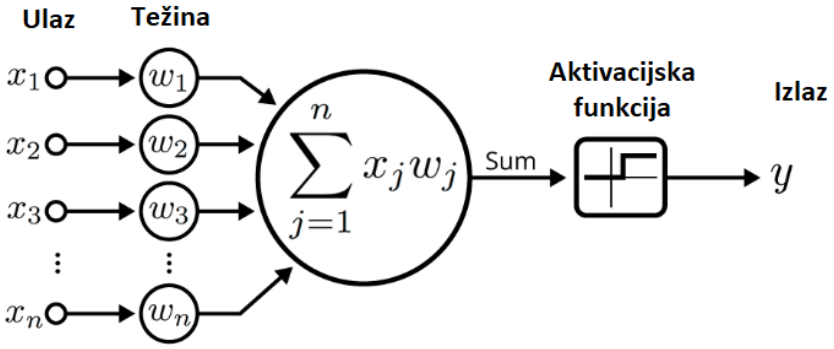
sloja: ulaznog, skrivenog i izlaznog sloja. Ukoliko mreža ima više skrivenih slojeva, riječ je o dubokoj neuronskoj mreži.

Neuron je matematički model biološkog neurona i osnovna je gradivna jedinica neuronske mreže, a predstavlja čvor kroz koji teku podaci i proračuni za svaki podatak. Neuroni su međusobno povezani sinapsama. Svaki neuron prima jedan ili više ulaznih podataka x_m koji mogu biti izlazi iz jednog ili više neurona u prethodnom sloju mreže ili dio iz skupa neobrađenih podataka ako se radi o ulaznom sloju mreže. Svaka sinapsa ima pridruženu težinu w_m koja utječe na važnost prethodnog neurona općenito u neuronskoj mreži. Svaki ulazni podatak množi se s odgovarajućom težinom i računa se težinska suma ulaznih vrijednosti koja predstavlja ukupnu snagu ulaznih signala, zatim se primjenom aktivacijske funkcije na ovu sumu određuje izlaz koji se potom prenosi drugim neuronima putem sinapsi. [25], [26] Osnovni način rada neurona je prikazan na slici 3.27.



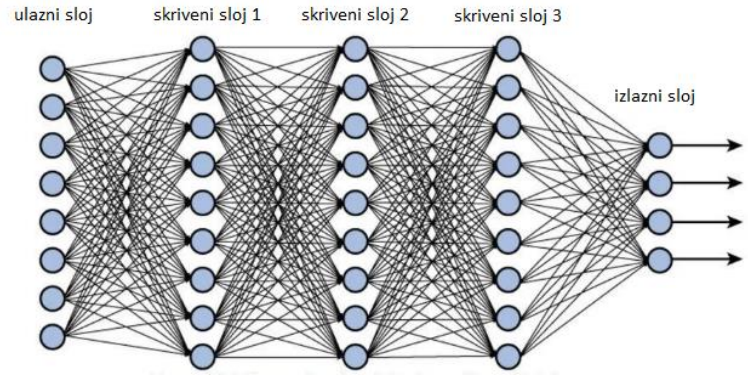
Slika 3.27. *Dijagram rada neurona u dubokoj neuronskoj mreži [26]*

Slika 3.28. prikazuje matematički postupak pretvaranja ulaza u izlaz za svaki pojedini neuron, gdje neuron zbraja svaki ulazni podatak, koji dobije od prethodnog neurona, pomnožen s odgovarajućom težinom i prosljeđuje ih aktivacijskoj funkciji koja procjenjuje izlaz. [26]



Slika 3.28. *Matematički postupak pretvaranja ulaza u izlaze [27]*

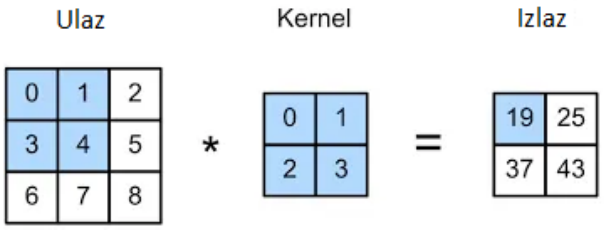
Na ulazni sloj dolazi ulazni podatak koji će mreža obraditi. Broj neurona u ovom sloju jednak je ukupnom broju značajki u ulaznim podacima, npr. broju piksela u slici, gdje svaki neuron predstavlja jednu značajku slike. Podatak iz ulaznog sloja potom na daljnju obradu ulazi u skriveni sloj, kojih može biti nekoliko. Broj skrivenih slojeva ovisi o ANN modelu i veličini podataka s kojima mreža radi. Svaki skriveni sloj može imati različit broj neurona koji je općenito veći od broja značajki. Ovaj sloj je odgovoran za izdvajanje potrebnih značajki iz ulaznih podataka dobivenih prethodnim slojem. Izlaz iz svakog skrivenog sloja računa se matričnim množenjem izlaza prethodnog sloja s odgovarajućim težinama tog sloja i dodavanjem pristranosti (engl. *bias*) nakon čega slijedi aktivacijska funkcija koja mrežu čini nelinearnom. Izlaz iz skrivenog sloja zatim ulazi u izlazni sloj gdje aktivacijska funkcija vrši ocjenu vjerojatnosti da podatak pripada svakoj od definiranih klasa, odnosno vrši se klasifikacija podataka. U ovom sloju broj neurona odgovara broju izlaznih klasa. [27], [28], [29] Slika 3.29. prikazuje primjer arhitekture duboke neuronske mreže s tri skrivena sloja, ulaznim i izlaznim slojem.



Slika 3.29. *Prikaz arhitekture duboke neuronske mreže [30]*

CNN se sastoji od konvolucijskih slojeva, slojeva sažimanja, sloja poravnanja i potpuno povezanog sloja. Konvolucijski slojevi su osnovne gradivne jedinice CNN modela i služe za izdvajanje značajki iz ulaznih podataka. Oni provode konvoluciju, matematičku operaciju koja omogućuje spajanje dva skupa informacija, pri čemu prostorna struktura ulaznih podataka ostaje sačuvana. Konvolucija se primjenjuje na ulazne podatke radi filtriranja informacija i stvaranja mape značajki. Filtar koji se pritom koristi naziva se kernel i ima oblik matrice koja može biti različitih dimenzija, npr. 2x2, 3x3, 5x5, 5x7 itd. Kernel klizi preko ulazne slike od početne pozicije (0,0) prema krajnjoj poziciji u slici (m,n) i pritom se na svakom dijelu slike kojeg kernel pokriva računa skalarni produkt dijela slike i kernela. To se vrši na način da se pomnože vrijednosti kernela s vrijednostima piksela dijela slike kojeg kernel prekriva i zatim se zbroje rezultati množenja za cijelo područje slike koju pokriva kernel. Primjenom istog kernela na različitim pozicijama u slici

kao izlaz iz sloja dobiva se aktivacijska mapa ili mapa značajki – 2D matrica s odzivima na pojedinom dijelu slike. Slika 3.30. prikazuje primjer konvolucije primjenom 2x2 kernela na sliku dimenzija 3x3. Brojevi u aktivacijskoj mapi se dobiju na sljedeći način:(0×0)+(1×1)+(3x29)+(4×3)=19; (1×0)+(2×1)+(4×2)+(5×3)=25; (3×0)+(4×1)+(6×2)+(7×3) =37; (4×0)+(5×1)+(7×2)+(8×3) = 43. [28], [29], [31], [32]

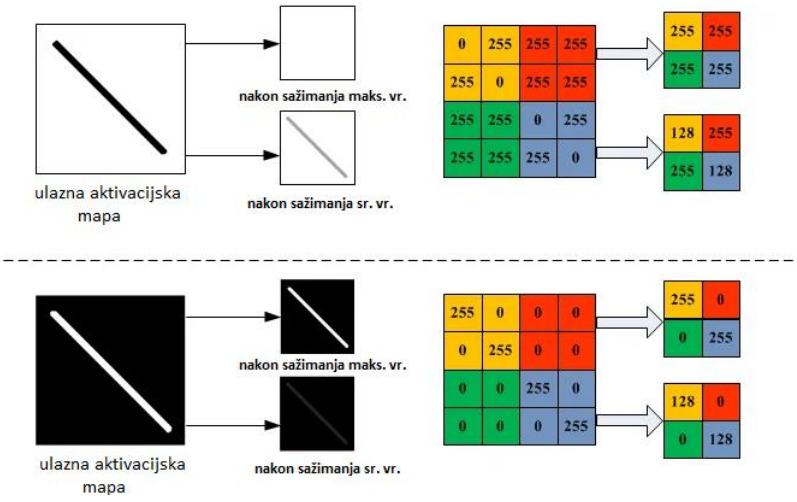


Slika 3.30.*Matematička operacija konvolucije na slici dimenzija 3x3 uz 2x2 kernel [29]*

Tri bitna hiperparametra koji utječu na veličinu izlaza koje je potrebno postaviti prije treniranja mreže su: dubina sloja, korak (engl. *stride*) i nadopuna ruba slike (engl. *padding*). Dubina se odnosi na broj kernela koje konvolucijski sloj sadrži. Svaki kernel stvara zasebnu aktivacijsku mapu, a skup ovih mapa čini konačni izlaz iz sloja. Npr. četiri različita kernela će dati četiri različite mape, čineći tako dubinu sloja jednaku vrijednosti četiri. Korak se odnosi na broj piksela za koji se kernel pomiče preko ulazne slike. Veći korak će rezultirati manjim dimenzijama izlaza jer će kernel brže pokriti cijelu sliku što može biti korisno za izdvajanje općih značajki i kontroliranje pojave pretreniranosti mreže te čini mrežu računalno učinkovitijom. S druge strane, manji korak rezultira većim dimenzijama izlaza i omogućuje izdvajanje finijih značajki i više detalja. Nadopuna ruba slike se obično koristi kada veličina kernela ne odgovara dimenzijama ulazne slike. Svrha korištenja ovog hiperparametra je prilagodba prostornih dimenzija izlaza i očuvanje prostornih informacija na rubovima slike. Postoje tri vrste nadopune: *valid* (zadnja operacija konvolucije se odbacuje ako dimenzije slike i kernela nisu usklađene), *same* (osigurava da izlaz ima istu veličinu kao ulaz dodavanjem nula oko ruba slike) i *full padding* (povećava veličinu izlaza dodavanjem nula na rub izlaza). [29], [31], [33]

Sloj sažimanja kao osnovnu funkciju ima smanjenje prostorne dimenzionalnosti izlaza što povećava računalnu učinkovitost mreže i sprječava pojavu pretreniranja. Pritom se koristi filtar preko cijele slike kao i za konvoluciju, ali je razlika što sada kernel kao izlaznu vrijednost u aktivacijskoj mapi uzima maksimalnu ili prosječnu vrijednost elemenata slike koje obuhvaća. Tako postoje dvije glavne vrste sažimanja: sažimanje maksimalnom vrijednošću (engl. *Max*

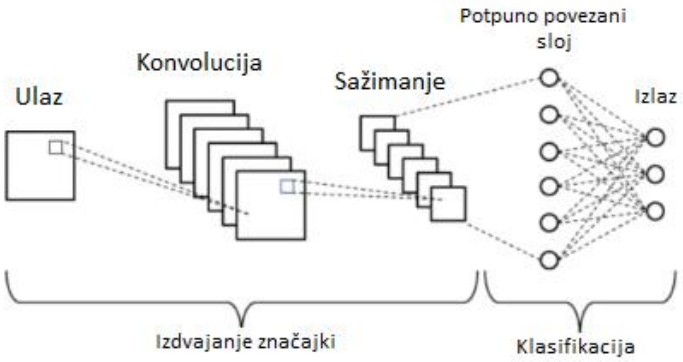
Pooling) i srednjom vrijednošću (engl. *Average Pooling*). Slika 3.31. prikazuje razliku u aktivacijskim mapama koje se dobiju korištenjem ove dvije vrste sažimanja.



Slika 3.31. *Rezultat primjene dvije vrste sažimanja na binarnu sliku [34]*

Nakon što su konvolucijski i slojevi sažimanja izdvojili relevantne značajke iz ulazne slike, izlaz je višedimenzionalnog oblika. Kako bi se ulaz prilagodio za zadnji, potpuno povezani sloj, potrebno ga je pretvoriti u jednodimenzionalni vektor da bi se mogla izvršiti klasifikacija podataka. Tu nastupa sloj poravnanja (engl. *Flatten*), koji mijenja oblik podataka, ali pritom ne mijenja stvarne informacije.

Potpuno povezani sloj (engl. *Dense*) ima broj neurona jednak broju unaprijed definiranih klasa čime se izdvojene značajke povezuju s odgovarajućim oznakama i vrši se kategorizacija ulaznih podataka u klase. Pritom se obično koristi aktivacijska funkcija Softmax koja vjerojatnosti da ulazni podatak pripada određenoj klasi skalira na vrijednosti između 0 i 1. Slika 3.32. prikazuje osnovnu arhitekturu CNN modela.



Slika 3.32. *Arhitektura CNN modela [32]*

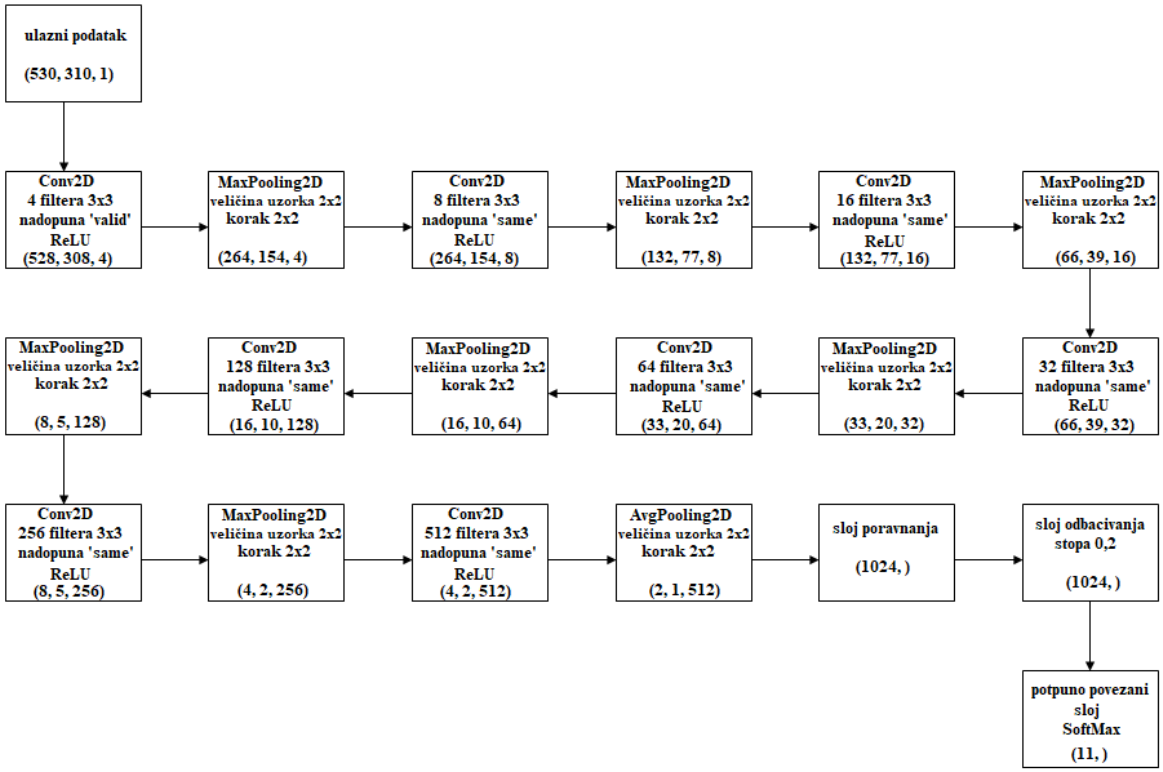
3.5. Arhitektura neuronske mreže

Razvijena CNN mreža se sastoji od ukupno 19 slojeva. Od toga ih je 8 konvolucijskih, 7 slojeva sažimanja maksimalnom vrijednošću i po jedan sloj sažimanja srednjom vrijednošću, sloj poravnanja, sloj odbacivanja (engl. *Dropout*) i potpuno povezani sloj.

Svaki konvolucijski sloj koristi aktivacijsku funkciju ReLU i kernel dimenzija 3x3 piksela. Svi osim prvog sloja imaju i nadopunu ruba slike koja je postavljena na *same* čime se oko slike dodaje sloj piksela kako bi pri prolasku kernela dimenzije slike ostale sačuvane i ne bi se prebrzo smanjile. To omogućuje mreži da održi dovoljnu veličinu slike za izvlačenje značajki prije nego što joj slojevi sažimanja dodatno smanje dimenzije. Slojevi sažimanja maksimalnom vrijednošću i prosječnom vrijednošću koriste veličinu uzorka (engl. *pool size*) 2x2 piksela i korak 2x2 piksela. Sloj poravnanja služi za konverziju izlaza iz prethodnog sloja sažimanja prosječnom vrijednošću u jednodimenzionalni vektor, koji se zatim dalje koristi kao ulaz u zadnji, potpuno povezani sloj. Potpuno povezani sloj koristi aktivacijsku funkciju Softmax i ima 11 mogućih izlaza s obzirom da mreža klasificira 11 gesti. Između sloja poravnanja i potpuno povezanog sloja se ubacuje sloj odbacivanja da bi se smanjila mogućnost pretreniranosti mreže nasumičnim odbacivanjem neurona u procesu treniranja mreže. Pri tome se koristi stopa odbacivanja 0,2 što znači da će se tijekom treninga odbaciti 20% nasumičnih neurona čime se osigurava da model ne ovisi previše o nekom specifičnom skupu neurona, nego mreža uči općenitije značajke i time postaje robusnija i bolje upotrebljiva na stvarnim podacima.

Mreža kao ulaz prima sliku u tonovima sive boje dimenzija 530x310 piksela koja je mreži predstavljena kao *numpy* polje dimenzija (530, 310, 1). Slika ulazi u prvi konvolucijski sloj gdje se na nju primjenjuju četiri 3x3 kernela. Po izlasku iz ovog sloja, slika će biti dimenzija (528, 308, 4) jer nadopuna ruba slike nije eksplicitno definirana pa poprima zadanu vrijednost *valid*. Nakon toga, slika novih dimenzija se prosljeđuje prvom sloju sažimanja maksimalnom vrijednošću sa veličinom uzorka 2x2 piksela i pomakom za 2x2 piksela. Na izlazu iz sloja slika ima dimenzije (264, 154, 4). Zatim se na nju primjenjuje novih osam 3x3 kernela u drugom konvolucijskom sloju i sada je dimenzija slike (264, 154, 8) jer je nadopuna ruba slike postavljena na *same*. Potom se još pet puta ponavljaju parovi konvolucijskih i slojeva sažimanja maksimalnom vrijednošću. Pri tome svi slojevi sažimanja maksimalnom vrijednošću imaju veličinu uzorka 2x2 piksela i pomak za 2x2 piksela, a konvolucijski slojevi svi imaju kernel veličine 3x3 piksela, ali im se broj kernela povećava na 16, 32, 64, 128 i 256 sa svakim idućim slojem. Dolaskom na zadnji konvolucijski sloj u mreži, dimenzija slike je (4, 2, 256). Ovaj sloj ima 512 kernela dimenzije 3x3 piksela i prosljeđuje sliku novih dimenzija (4, 2, 512) dalje na sloj sažimanja prosječnom vrijednošću koji

ima veličinu uzorka, kao i pomak, 2x2 piksela. Izlaskom slike iz ovog sloja, dimenzije su joj se smanjile na (2, 1, 512) te potom slika ulazi u sloj poravnanja nakon kojega slika ima dimenzije (1024,). Slika zatim ulazi u sloj odbacivanja sa stopom odbacivanja 0,2 i na kraju u potpuno povezani sloj koji ima 11 izlaza. Na slici 3.33. je prikazana arhitektura mreže kao i veličine izlaza slike iz svakog sloja. U prilogu P.3.6. prikazan je kod kojim je definirana struktura neuronske mreže.



Slika 3.33. *Arhitektura CNN mreže*

3.6. Trening neuronske mreže

Nakon što se signal obradi i pripremi za ulaz u mrežu kako je opisano u potpoglavljima 3.3.2.-3.3.4., potrebno je mrežu istrenirati prije nego što ju je moguće primijeniti za detekciju gesti u stvarnim uvjetima kako bi naučila značajke svake pojedine geste i mogla ih razlikovati. S obzirom da je odabrana CNN mreža, potreban je veliki broj podataka za njen trening kako bi što bolje radila u stvarnim uvjetima. Za to su korištene dvije baze opisane u poglavlju 3.2.

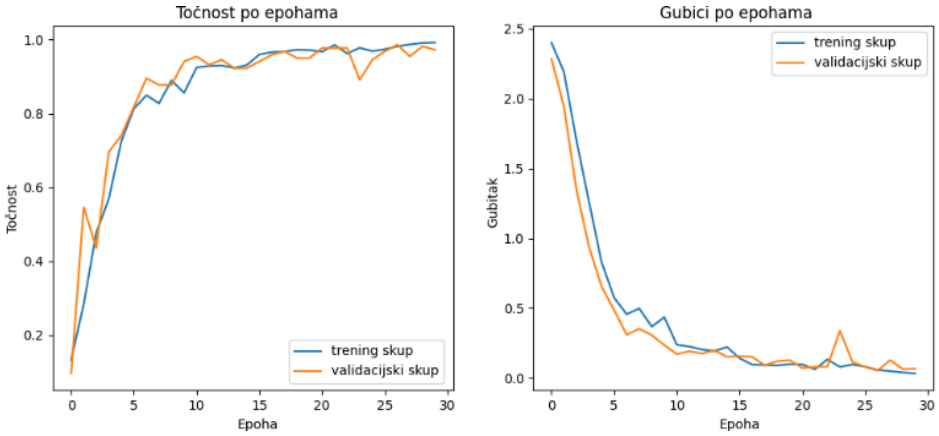
Pri treningu su korišteni sljedeći parametri: veličina grupe uzoraka (engl. *batch size*), broj epoha, Adam optimizator i stopa učenja (engl. *learning rate*). Veličina grupe odnosi se na broj uzoraka za trening mreže koji se obrađuju prije nego što se ažuriraju težine modela. Na kraju svake grupe uzoraka predikcije se uspoređuju s očekivanim izlaznim varijablama mreže i računa se

pogreška koju algoritam ažuriranja potom koristi za poboljšanje modela. Manja grupa povećava točnost modela zbog bolje generalizacije, ali povećava i vrijeme treninga. Veća grupa ubrzava proces treniranja mreže, ali smanjuje točnost jer postoji mogućnost da se mreža pretrenira ili da generalizacija podataka ne bude dobra. Broj epoha predstavlja broj prolaza kroz cijeli trening skup podataka, a jedna epoha znači da je svaki uzorak u trening skupu podataka proveo ažuriranje težina modela. Epoha se dakle sastoji od jedne ili više grupe uzoraka. Na primjer, ako se skup podataka sastoji od 200 uzoraka, a za veličinu grupe se uzme vrijednost 5, to znači da će skup podataka biti podijeljen na 40 grupa unutar kojih se nalazi po 5 uzoraka. Također, to znači da će se jedna epoha sastojati od 40 grupa uzoraka i provest će se 40 ažuriranja težina modela tijekom jedne epohe. Veći broj epoha znači da će model više puta proći kroz trening skup podataka pa može bolje učiti iz njega što u konačnici može poboljšati točnost modela. Međutim, nakon određenog broja epoha model se počinje previše prilagođavati podacima na kojima uči što može dovesti do pretreniranja modela. Pretreniranje znači da je model previše prilagođen podacima na kojima je učio i postizat će visoku točnost nad trening podacima, ali će davati loše rezultate na podacima koji su mu novi – test i validacijskom skupu podataka. S druge strane, ukoliko je broj epoha tijekom treninga premalen, doći će do nedovoljne treniranosti mreže (engl. *underfitting*) što znači da model nije dovoljno naučio iz trening podataka pa ima nisku točnost. Stopa učenja kontrolira brzinu kojom model uči i regulira vrijednost pogreške kojom se težine modela ažuriraju na kraju svake grupe uzoraka. Adam optimizator koristi adaptivnu stopu učenja čime se održava stabilnost treninga što na kraju može povećati točnost modela.

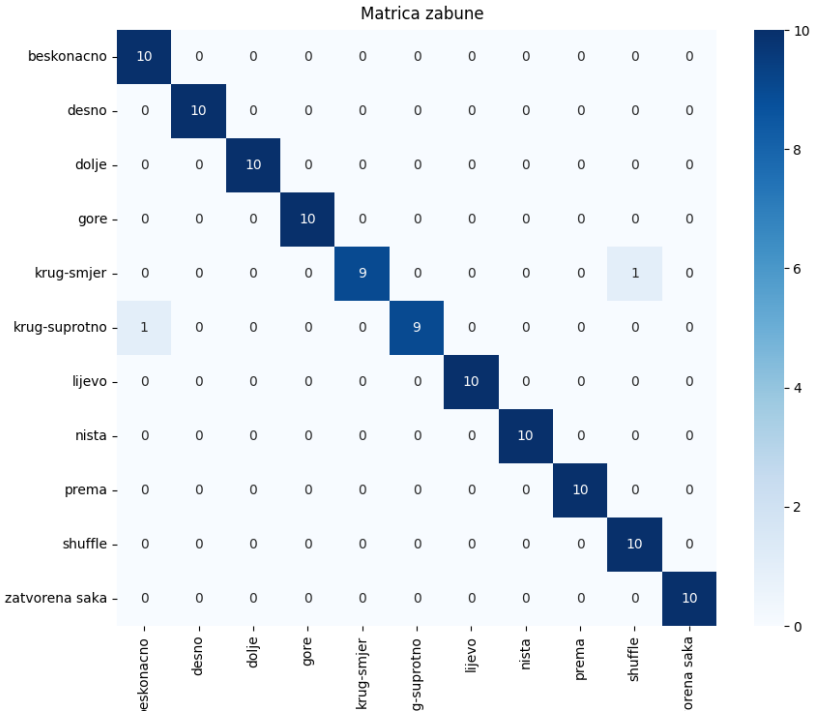
Tijekom treninga su se spremale informacije o tijeku treniranja uključujući metrike za trening i validaciju za svaku epohu. Te metrike računaju gubitke i točnost za validacijski i trening skup podataka, a grafičkim prikazom vrijednosti za svaku epohu omogućuje se praćenje promjene performansi modela u vremenu tijekom treninga, što je osobito korisno za dijagnosticiranje problema kao što su pretreniranost odnosno nedovoljna treniranost modela.

Za trening je prvo korištena baza sa podacima koje je snimila samo jedna osoba. Podaci iz baze su podijeljeni na skup podataka za trening, validaciju i test u omjeru 70:20:10, pa tako s obzirom da svaka gesta u ovoj bazi ima po 100 uzoraka, to će biti ukupno 770 podataka za trening (70 po svakoj gesti), 220 za validaciju (20 po gesti) i 110 za test mreže pri treningu (10 za svaku gestu). Pri prvom treningu je korištena veličina grupe 32, 15 epoha i Adam optimizator uz stopu učenja 0,0001. Po završetku treninga su nacrtani graf točnosti i graf gubitaka po epohama nad podacima za validaciju i test, što je prikazano na slici 3.34. Zatim je nacrtana matrica zabune kojom se prikazuje koliki je broj gesti točno klasificiran, a koje geste je mreža pogrešno

klasificirala kao neke druge, prikazano na slici 3.35. Ova metrika omogućuje dijagnosticiranje problema i optimizaciju rada mreže jer se točno vidi koje geste mreža miješa, a koje detektira točno pa se u idućoj iteraciji treninga mogu bolje uskladiti parametri da mreža što točnije klasificira geste. Trening je ponovljen nekoliko puta dok se nije dobila zadovoljavajuća točnost modela koja iznosi 98,18%.



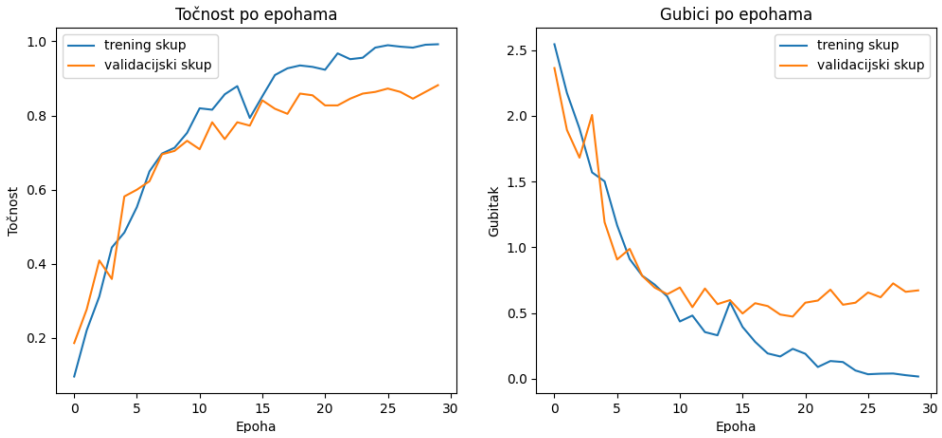
Slika 3.34. *Graf točnosti i graf gubitaka po epohama pri prvom treningu mreže*



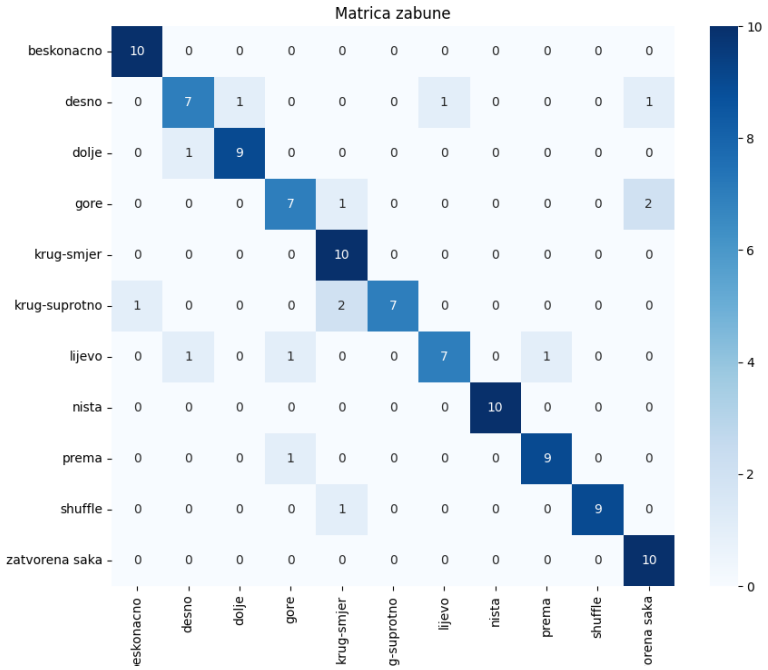
Slika 3.35. *Matrica zabune pri prvom treningu mreže*

Nakon toga je mreža ponovno trenirana nad podacima koje su snimile druge osobe kako bi se mreži uveli novi, raznovrsniji uzorci za svaku gestu koju je već naučila da bi bila robusnija i primjenjivija u stvarnim uvjetima. Ova baza kao i prethodna sadrži 1100 uzoraka gesti te je podjela

na skup podataka za trening, validaciju i test u istom omjeru - 770:220:110. Pritom je dobivena točnost 86,36%, a korištena veličina grupe je 32, broj epoha je 30 i korišten je Adam optimizator bez eksplicitno navedene stope učenja. Točnost u odnosu na model treniran nad podacima koje je snimila samo jedna osoba je smanjena jer se u ovoj bazi podataka nalaze raznovrsniji podaci, dok su u prvoj bazi podaci dosta homogeni. Na slici 3.36. prikazani su grafovi točnosti i gubitaka po epohama tijekom validacije i testa, a na slici 3.37. prikazana je matrica zabune.



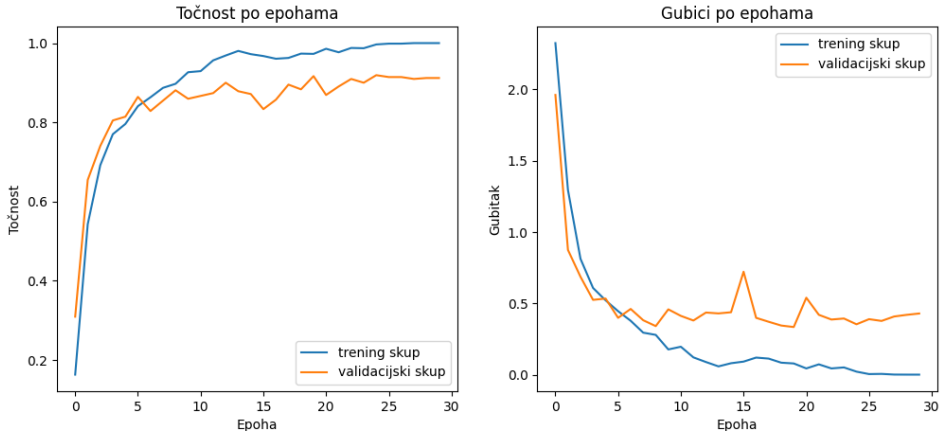
Slika 3.37. *Graf točnosti i graf gubitaka po epohama pri drugom treningu mreže*



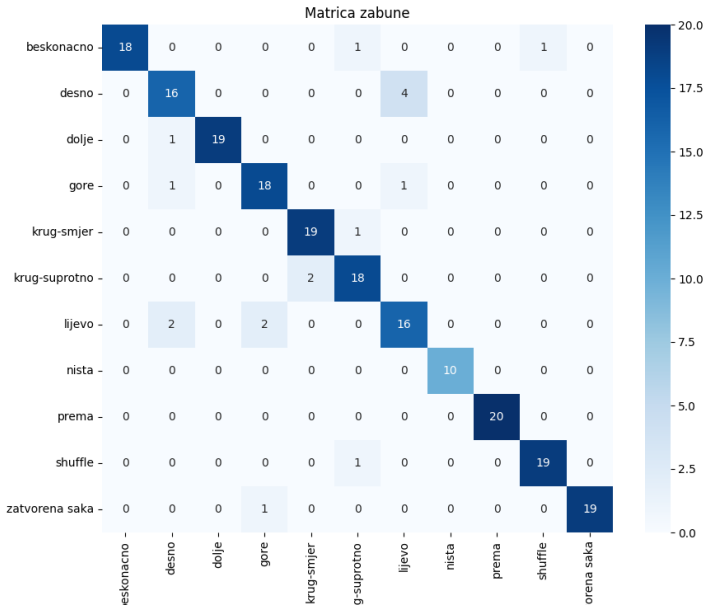
Slika 3.37. *Matrica zabune pri drugom treningu mreže*

Na kraju je trening proveden nad svim podacima da se poboljša točnost detekcije i popravi rad razvijenog algoritma s obzirom da su u prvoj bazi podataka uzorci gesti jako čisti i homogeni,

a u drugoj bazi je jedan dio uzoraka lošije kvalitete što je i bio cilj kako bi se simulirao način izvođenja gesti krajnjeg korisnika u različitim uvjetima da bi mreža onda i u takvim uvjetima mogla uspješno izvršiti detekciju. Ova baza podataka sadrži 2100 uzoraka i podjelom na trening, test i validacijski skup u omjeru 70:20:10, svaki skup ukupno čine 1470:420:210 uzoraka. Sada je dobivena točnost modela 91,43%, a pri tome su korišteni veličina grupe 32, broj epoha 30 i Adam optimizator bez navedene stope učenja. Na slici 3.38. prikazani su grafovi točnosti i gubitaka po epohama tijekom validacije i testa, a na slici 3.39. prikazana je matrica zabune.



Slika 3.38. *Graf točnosti i graf gubitaka po epohama pri trećem treningu mreže*



Slika 3.39. *Matrica zabune pri trećem treningu mreže*

Mreža se sastoji od ukupno 1585099 parametara koje koristi za trening. U prilogu P.3.7. prikazan je dio koda za trening neuronske mreže.

3.7. Rad razvijenog sustava za detekciju gesti ruku u stvarnom vremenu

U ovom poglavlju, opisan je princip rada sustava u stvarnom vremenu. Mikrofoni neprestano snimaju zvučne valove koji mu dolaze iz okoline. Snimljeni signali se istovremeno dijele na segmente trajanja 2 sekunde redom kojim dolaze s ulaza. Potom se segmenti obrađuju da budu odgovarajuće vrste i dimenzija za ulaz u istrenirani model kako je opisano u potpoglavljima 3.3.2.-3.3.4. Model se poziva nad svakim nadolazećim segmentom i vrši detekciju gesti. Primjer rezultata detekcije je prikazan na slici 3.40.

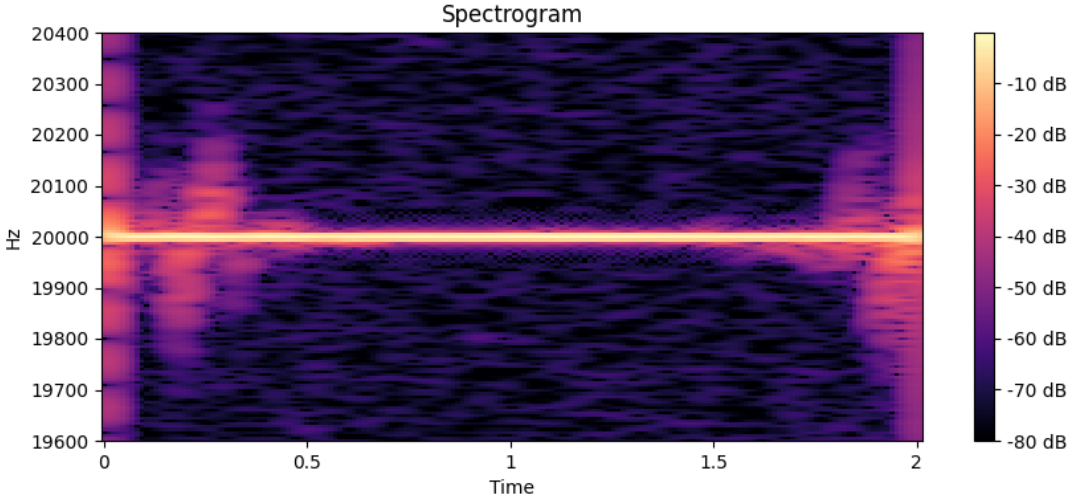
```
Starting recording and prediction cycle...
Real-time audio stream started. Press Ctrl+C to stop.
Recording completed.
Conversion to stereo completed
C:\Users\Dell OptiPlex 7040\Desktop\kodovi konačno\zad
  plt.figure(figsize=(10, 4))
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 155ms/step
Detected gesture: zatvorena saka
Starting recording and prediction cycle...
Recording completed.
Conversion to stereo completed
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 20ms/step
Detected gesture: nista
Starting recording and prediction cycle...
Recording completed.
Conversion to stereo completed
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 25ms/step
Detected gesture: nista
Starting recording and prediction cycle...
Recording completed.
Conversion to stereo completed
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 23ms/step
Detected gesture: prema
Starting recording and prediction cycle...
Recording completed.
Conversion to stereo completed
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 25ms/step
Detected gesture: dolje
Starting recording and prediction cycle...
Recording completed.
Conversion to stereo completed
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 30ms/step
Detected gesture: krug-suprotno
Starting recording and prediction cycle...
Recording completed.
Conversion to stereo completed
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 25ms/step
Detected gesture: krug-smjer
Starting recording and prediction cycle...

Interrupted by user.
Recording completed.
Conversion to stereo completed
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 35ms/step
Detected gesture: nista
Traceback (most recent call last):
  File "C:\Users\Dell OptiPlex 7040\Desktop\kodovi kor
    time.sleep(1)
KeyboardInterrupt
```

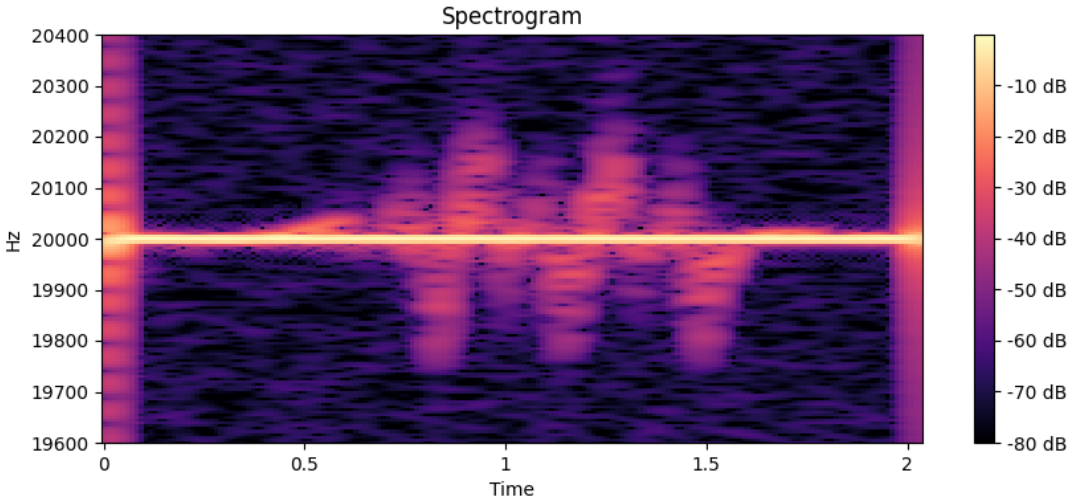
Slika 3.40. *Prikaz rezultata detekcije u terminalu*

Međutim, da bi detekcija bila uspješna potrebno je napraviti gestu točno unutar 2 sekunde trajanja jednog segmenta što nije primjenjivo u stvarnim uvjetima jer je teško pogoditi točno

vrijeme bez znanja kada se signal snima. Na slikama 3.41. i 3.42. su prikazani primjeri slučaja kada se gesta izvede unutar 2 sekunde i kada se gesta izvede na granici između dva segmenta.

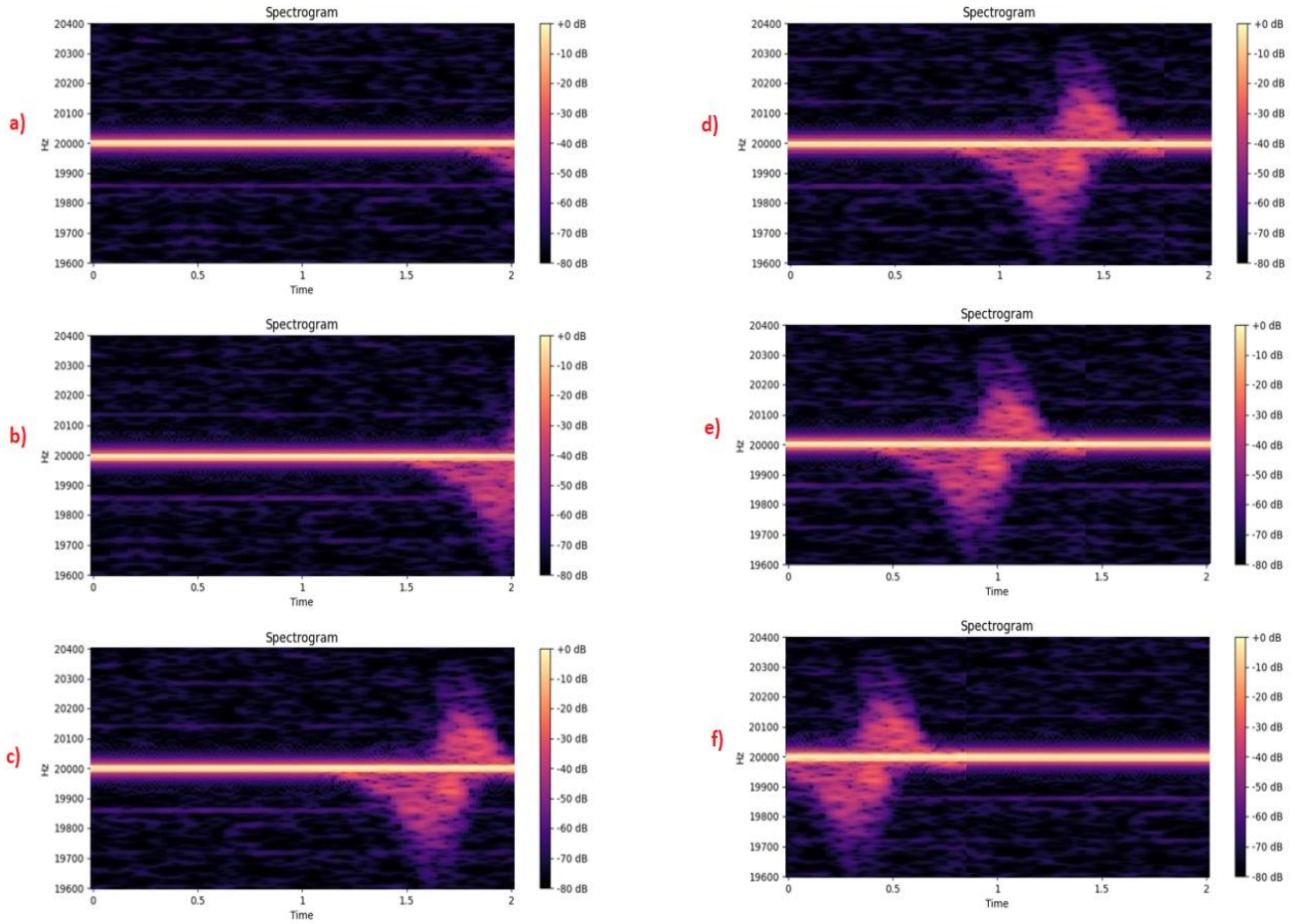


Slika 3.41. *Spektrogram geste shuffle izvedene u krivo vrijeme*



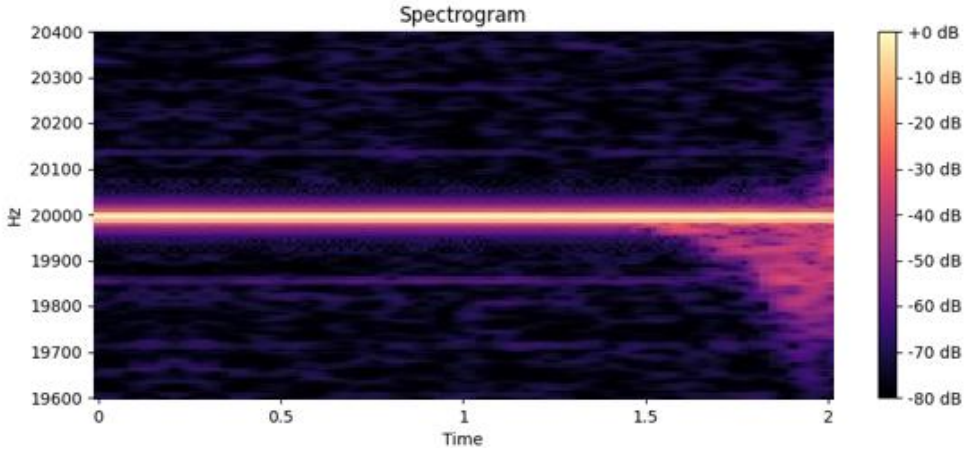
Slika 3.42. *Spektrogram geste shuffle izvedene u dobro vrijeme*

Stoga, napravljeno je rješenje koje je primjenjivo u stvarnim uvjetima upotrebom kliznih spektrograma i postavljanjem granice detekcije. Spektrogram se prvo generira za početne 2 sekunde trajanja signala koji dolazi s ulaza i nakon toga se snimaju segmenti trajanja 0,2 sekunde. Svaki novi nadolazeći segment spektrograma se postavlja na kraj prethodnog, a početne 0,2 sekunde starog spektrograma se brišu kako bi spektrogram nad kojim se poziva model za detekciju gesti uvijek bio jednake duljine: 2 sekunde. Način rada kliznih spektrograma je prikazan na slici 3.43.

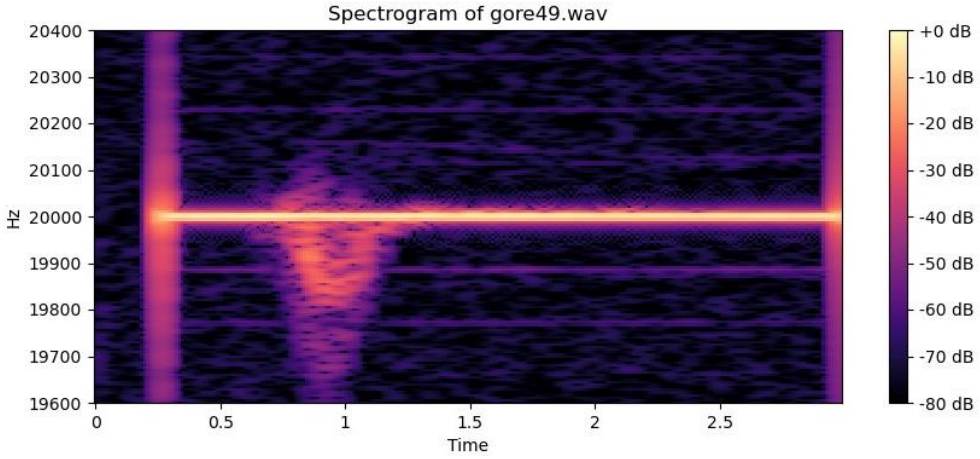


Slika 3.43. *Klizni spektrogram signala u stvarnom vremenu: a) trenutak 0-2 s, b) trenutak 0.2-2.2 s, c) trenutak 0.4-2.4s, d) trenutak 0.6-2.6 s, e) trenutak 0.8-2.8 s, f) trenutak 1,0-3,0s*

Budući da se u konceptu kliznog spektrograma ista gesta pojavljuje na nekoliko uzastopnih spektrograma, a kako se ista gesta ne bi detektirala više puta zaredom, uvedena je granica na koliko spektrograma zaredom gesta mora biti detektirana da bi bila klasificirana kao prava detekcija (stvarna gesta koja je napravljena). Osim problema uzastopne detekcije iste geste, postavljanjem granice se rješava i problem lažne detekcije s obzirom da se cjeloviti spektrogram ponekad neće pojaviti odmah, nego tek nakon nekoliko segmenata, a ponekad početak spektrograma jedne geste može izgledati kao cjeloviti spektrogram neke druge geste što je prikazano na slikama 3.44. i 3.45. Ukoliko se ne postavi granica, sustav će detektirati gestu *gore* i pokrenuti funkciju u aplikaciji *Spotify Web* za povećavanje glasnoće zvuka, međutim korisnik je zapravo napravio gestu *lijevo* u želji da prebaci pjesmu. Ovakva situacija jako narušava kvalitetu usluge i zadovoljstvo korisnika. Granica je postavljena na 3 uzastopna spektrograma.



Slika 3.44. *Prikaz početnog dijela spektrograma geste lijevo*



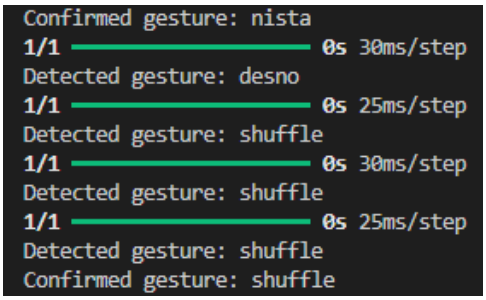
Slika 3.45. *Prikaz cijelog spektrograma geste gore*

Na slici 3.46. prikazan je primjer rezultata detekcije sa aktivnom granicom i primjenom kliznog spektrograma, a prilog P.3.8. prikazuje dio koda kojim se vrši detekcija gesti u stvarnom vremenu.

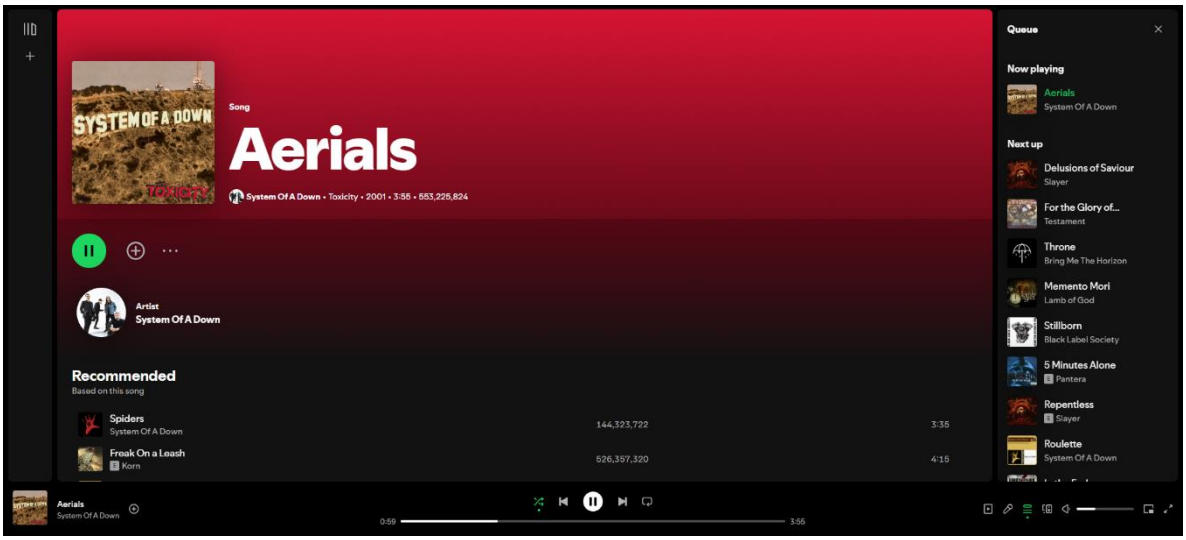
Slika 3.46. *Prikaz rezultata detekcije u terminalu sa aktivnom granicom detekcije*

4. PRIMJENA RAZVIJENOG SUSTAVA NA APLIKACIJU SPOTIFY WEB

Nakon što se omogućila uspješna detekcija gesti ruku u stvarnom vremenu, kako je opisano u trećem poglavlju, sustav za detekciju gesti ruku je primijenjen na aplikaciju *Spotify Web*. Kada sustav detektira jednu od ranije definiranih gesti, pokreće se odgovarajuća funkcija za kontrolu *Spotify Web* aplikacije. Za realizaciju ovog rješenja korišten je *Spotify Web* API za čiju se upotrebu prvo trebala provesti autorizacija korisnika i odrediti područje funkcija kojima će sustav za detekciju moći upravljati. Prilog P.4.1. prikazuje dio koda kojim su definirane funkcije za kontrolu *Spotify Web* aplikacije kada algoritam detektira odgovarajuću gestu, a prilog P.4.2. prikazuje način autentikacije korisnika i određivanje područja korištenih funkcija. Na slici 4.1. je prikazan primjer rezultata detekcije, a funkcija koju potom sustav pokreće na temelju detektirane geste je prikazano na slici 4.2. Detektirana je gesta *shuffle* koja pokreće funkciju za izvođenje pjesama nasumičnim redoslijedom.



Slika 4.1. *Prikaz rezultata detekcije u terminalu*



Slika 4.2. *Prikaz sučelja aplikacije Spotify Web: zelene strelice koje se križaju u donjem srednjem dijelu ekrana označavaju aktivnu funkciju za nasumično izvođenje pjesama*

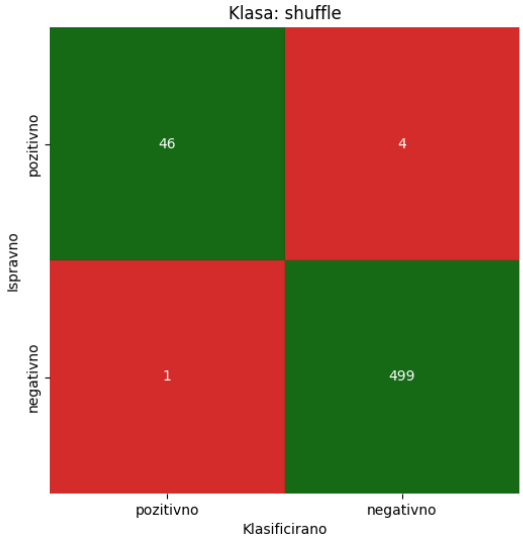
5. EVALUACIJA RAZVIJENOG SUSTAVA ZA DETEKCIJU GESTI RUKU

Nakon što se provede trening CNN modela za detekciju gesti ruku, potrebno je provesti evaluaciju njegova rada kako bi se utvrdilo koliko dobro model vrši detekciju i treba li uvesti daljnja poboljšanja. Evaluacija se vrši nad skupom neviđenih podataka. U tu svrhu se koristi baza podataka koja sadrži 550 uzoraka opisana u poglavlju 3.2. Tijekom procesa evaluacije, korištene su sljedeće metrike za procjenu rada modela: točnost, preciznost, odziv (engl. *recall*), matrica zabune, makro prosjek i težinski prosjek (engl. *weighted average*).

Matrica zabune je metrika koja služi za vizualizaciju klasifikacije uzoraka u pojedine klase. Matricom zabune se jasno vidi koliko uzoraka model točno klasificira, a koliko ih klasificira pogrešno na način da se točno klasificirani uzorci nalaze na dijagonali matrice, a svi uzorci koji su izvan nje su pogrešno klasificirani. Matrica je dimenzija $n\ x\ m$, gdje je n redak matrice i označava stvarnu klasu uzorka, a m predstavlja redak matrice i označava klasu u koju je model klasificirao uzorke. Matrica se sastoji od četiri zasebne metrike koje predstavljaju broj točnih i netočnih klasifikacija uzoraka [35]:

- Pravi pozitivi (engl. *True Positive, TP*) – broj točno klasificiranih uzoraka kao pozitivnih
- Lažni negativi (engl. *False Negative, FN*) – broj netočno klasificiranih pozitivnih uzoraka kao negativnih
- Lažni pozitivi (engl. *False Positive, FP*) – broj netočno klasificiranih negativnih uzoraka kao pozitivnih
- Pravi negativi (engl. *True Negative, TN*) – broj točno klasificiranih negativnih uzoraka kao negativnih

Cilj je maksimizirati prave pozitive i negative, a minimizirati lažne. Slika 5.1. prikazuje rezultat četiri metrike matrice zabune za gestu *shuffle*. U gornjem lijevom kutu prikazan je broj pravih pozitiva (46), u gornjem desnom kutu broj lažnih negativa (4), u donjem lijevom kutu broj lažnih pozitiva (1), a u donjem desnom kutu broj pravih negativa (499).



Slika 5.1. *matrica zabune za gestu shuffle*

Točnost modela predstavlja omjer točno klasificiranih uzoraka u odnosu na ukupan broj izvršenih klasifikacija [31]:

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \qquad 5-1$$

Iako se točnost koristi za općenitu procjenu izvedbe modela po svim klasama i dobro ju je koristiti pri treningu modela za podešavanje hiperparametara, ovu metriku nije dobro koristiti samostalno pri evaluaciji modela jer rezultat može biti varljiv. Na primjer, ukoliko postoji neuravnotežen skup podataka klasificiran u dvije klase gdje se u jednoj klasi nalazi 550 uzoraka, a u drugoj 50, točnost za prvu klasu će biti veća nego za drugu klasu. Ako je model izvršio ukupno 530/550 točnih klasifikacija uzoraka za prvu klasu, u usporedbi sa samo 5/50 za drugu klasu, tada je ukupna točnost modela (530+5)/600 = 0.8917, odnosno 89,17%. Ukoliko se tijekom evaluacije koristi samo ovaj podatak, čini se da model ima visoku točnost klasifikacije za obje klase, ali u stvarnosti visokom točnošću će klasificirati samo podatke u klasi koja ima više uzoraka, a za klasu s manje uzoraka klasifikacija će biti loša. [35] Stoga se uz točnost koriste i druge navedene metrike.

Preciznost je metrika koja se koristi za ograničavanje lažnih pozitiva. Računa se kao omjer broja uzoraka koji su ispravno klasificirani i ukupnog broja uzoraka koji su klasificirani kao pozitivi, bilo pravi ili lažni [36]:

$$P = \frac{TP}{TP + FP} \qquad 5-2$$

Ova metrika omogućuje računanje stope točnih pozitivnih klasifikacija, odnosno mjeri pouzdanost modela u klasificiranju uzoraka kao pozitiva. [35], [37] Što je više točnih pozitivnih klasifikacija, odnosno manje neispravnih klasifikacija uzoraka kao pozitiva, to je preciznost modela veća.

Odziv je metrika koja se koristi za ograničavanje lažnih negativa. [37] Odziv mjeri sposobnost modela da otkrije pozitivne uzorke – što je odziv veći, to je detektirano više pozitivnih uzoraka. Računa se kao omjer broja ispravno klasificiranih pozitiva i ukupnog broja pozitivnih uzoraka [36]:

$$R = \frac{TP}{TP + FN} \qquad 5-3$$

Odziv u obzir uzima samo kako su pozitivni uzorci klasificirani. Tako će odziv biti 100% kada model klasificira sve pozitivne uzorke kao pozitivne, čak i ako su svi negativni uzorci netočno klasificirani kao pozitivni. S druge strane, odziv će biti 0% kada model ne uspije detektirati niti jedan pozitivan uzorak, npr. ukoliko se svi pozitivni uzorci netočno klasificiraju kao negativni.[35]

Kada model ima visoki odziv, a nisku preciznost, ispravno će klasificirati većinu pozitivnih uzoraka, ali ima mnogo lažnih pozitiva. Kada model ima visoku preciznost, a niski odziv, tada je model točan kada klasificira uzorak kao pozitivan, ali neće klasificirati sve pozitivne uzorke. [35]

F1 je metrika koja se koristi za ograničavanje i lažnih pozitiva i lažnih negativa. [37] Koristi se za sveobuhvatnu procjenu točnosti modela, a rezultat je harmonijska sredina između preciznosti i odziva. Harmonijska sredina se koristi jer ona odbacuje udaljenije vrijednosti. F1 se računa kao [37]:

$$F1 = 2 * \frac{P * R}{P + R} \qquad 5-4$$

Makro i težinski prosjeci se koriste za rezimiranje izvedbe modela kroz više klasa računajući srednju vrijednost korištenih metrika (preciznost, odziv i F1). Razlika između ove dvije metrike je što makro prosjek svaku klasu tretira jednako bez obzira na broj uzoraka u svakoj klasi, dok težinski prosjek klasama dodjeljuje različite važnosti (težine) s obzirom na broj uzoraka u svakoj klasi. Težinski prosjek je dobro koristiti kada postoji neuravnotežen skup podataka jer će ova metrika odraziti tu neuravnoteženost i dat će realističnije rezultate. S obzirom da u ovom radu svaka baza podataka ima jednak broj uzoraka u svakoj klasi, sve klase će imati istu težinu te će obje metrike dati isti rezultat. Makro prosjek se računa na sljedeći način [38]:

$$MAvg = \frac{1}{N}\sum_{i=1}^{N} metrika_i \qquad 5-5$$

Težinski prosjek se računa kao [38]:

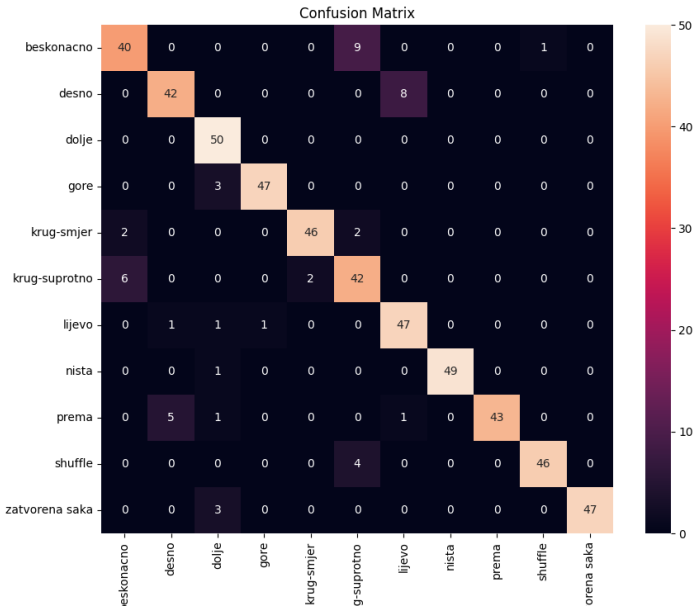
$$WAvg = \frac{1}{N}\sum_{i=1}^{N} w_i * metrika_i \qquad 5-6$$

pri čemu je težina *w* omjer broja uzoraka u klasi i ukupnog broja uzoraka u bazi podataka. Slika 5.2. prikazuje rezultate metrika za skup testnih podataka za svaku klasu posebno i ukupnu točnost te makro i težinski prosjek svake metrike. Slika 5.3. prikazuje matricu zabune za skup testnih uzoraka.

Accuracy: 90.73%

	precision	recall	f1-score	support
beskonacno	0.83	0.80	0.82	50
desno	0.88	0.84	0.86	50
dolje	0.85	1.00	0.92	50
gore	0.98	0.94	0.96	50
krug-smjer	0.96	0.92	0.94	50
krug-suprotno	0.74	0.84	0.79	50
lijevo	0.84	0.94	0.89	50
nista	1.00	0.98	0.99	50
prema	1.00	0.86	0.92	50
shuffle	0.98	0.92	0.95	50
zatvorena saka	1.00	0.94	0.97	50
accuracy			0.91	550
macro avg	0.91	0.91	0.91	550
weighted avg	0.91	0.91	0.91	550

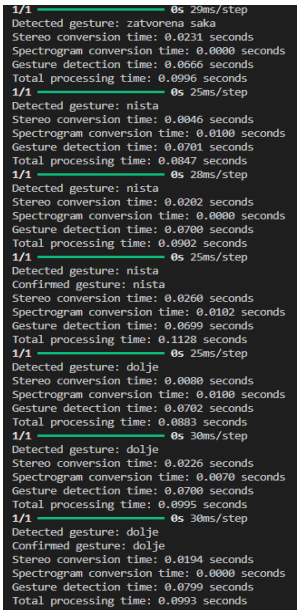
Slika 5.2. *Rezultat metrika za evaluaciju modela nad 550 uzoraka: točnost, preciznost, odziv, F1, makro i težinski prosjek svake metrike*



Slika 5.3. *Matrica zabune za testni skup od 550 uzoraka*

Iz matrice zabune je vidljivo da model zbog sličnosti spektrograma ponekad zamijeni geste *lijevo* i *desno* pri čemu gestu *desno* točno klasificira u 84% slučajeva, a gestu *lijevo* u 94% slučajeva. Gestu *dolje* model je klasificirao točno u 100%, a gestu *ništa* u 98% slučajeva. Gestu *gore* model točno klasificira u 94% slučajeva, a ponekad ju zamijeni sa gestom *dolje*. Model ponekad geste *beskonačno, krug u smjeru kazaljke na satu* i *krug u smjeru suprotno od kazaljke na satu* međusobno krivo klasificira pri čemu gestu *beskonačno* točno klasificira u 80% slučajeva, *krug u smjeru suprotno od kazaljke na satu* u 84%, a *krug u smjeru kazaljke na satu* u 92% slučajeva. Gestu *prema*, koju model točno klasificira u 86% slučajeva, ponekad zamijeni sa gestama *lijevo* i *desno*, gestu *zatvorena šaka* točno klasificira u 94% slučajeva, a zamijeni ju sa gestom *dolje* u preostalih 6% slučajeva i gestu *shuffle* točno klasificira u 92% slučajeva. Točnost modela prilikom evaluacije je 90.73%, preciznost za svaku gestu je između 74 i 100%, odziv je u rasponu od 80 do 100%, F1 vrijednost je 79 do 99%, a težinski i makro prosjeci za svaku metriku iznose 91%.

Na kraju je testiran rad sustava u stvarnom vremenu na način da je svaka gesta ponovljena 10 puta i praćen je broj točnih detekcija. Gesta *gore* točno je detektirana 8 puta, gesta *dolje* 9 puta, *lijevo* 8 puta, *desno* 7 puta, *prema* 10 puta, *krug u smjeru kazaljke na satu* 7 puta, *krug u suprotnom smjeru od kazaljke na satu* i *beskonačno* 8 puta, *zatvorena šaka* 9 puta, a geste *shuffle* i *ništa* točno su detektirane svih 10 puta. Pritom je ukupno vrijeme potrebno algoritmu da izvrši obradu signala, detekciju i pokretanje odgovarajuće naredbe za upravljanje *Spotify Web* aplikacijom ~100 ms, prikazano na slici 5.4.



Slika 5.4. *Prikaz vremena rada algoritma*

Iz navedenih rezultata je vidljivo da je razvijeni algoritam dovoljno brz, pouzdan, precizan i točan da bi bio primjenjiv u stvarnim uvjetima u vremenski nekritičnim aplikacijama kao što je upravljanje aplikacijom za reprodukciju glazbe.

6. POKRETANJE RAZVIJENOG SUSTAVA ZA DETEKCIJU GESTI RUKU

Razvijeni algoritam za detekciju gesti ruku se koristi pokretanjem skripte *master.py*. Ova skripta potom pokreće zasebne skripte *audioProcMain.py* i *soundProduction.py*. Skripta *soundProduction.py* koristi se za kontinuirano generiranje sinusnog signala frekvencije 20 kHz. U skripti *audioProcMain.py* vrše se pozivi funkcija za snimanje signala, detekciju gesti i kontrolu *Spotify Web* aplikacije iz *audioProcFunc.py* skripte i definirani su parametri potrebni za uspješno izvođenje ovih funkcija. Skripta *audioProcFunc.py* vrši obradu i snimanje signala, detekciju gesti i kontrolu *Spotify Web* aplikacije te sadrži dio za autentikaciju korisnika i određivanje područja korištenih funkcija za kontrolu ove aplikacije. Zaustavljanjem izvođenja skripte *master.py*, sustav prestaje s radom.

Za uspješno pokretanje algoritma, potrebno je instalirati sljedeće biblioteke:

- OpenCV
- Numpy
- Sounddevice
- Librosa
- TensorFlow
- Spotipy

7. ZAKLJUČAK

U ovom diplomskom radu razvijen je i testiran sustav za detekciju gesti ruku na temelju nečujnog audio signala kao alternativna metoda interakcije korisnika i računala. Razvijeni sustav sastoji se od komercijalnih zvučnika i mikrofona. Zvučnici generiraju ultrazvučni signal frekvencije 20 kHz, a mikrofoni bilježe refleksije tog signala od ruke korisnika. Pritom se iskorištava pojava Dopplerovog efekta, a primjenom kratkotrajne Fourierove transformacije dobiveni su spektrogrami snimljenih signala. Rad sustava je temeljen na dubokom učenju, gdje je konvolucijska neuronska mreža trenirana na vlastitoj bazi podataka koja sadrži 2100 spektrograma. Pritom je mreža vršila klasifikaciju uzoraka u 11 zasebnih klasa, a nakon treninga je model postigao točnost 91,43%.

Kako bi se provela dodatna procjena izvedbe modela, pet je osoba pozvano da testiraju sustav na način da je svaka osoba izvršila svih 11 gesti 10 puta, što ukupno čini 550 testnih uzoraka. Pri evaluaciji, uz točnost, korišteni su odziv, preciznost, matrica zabune, F1 vrijednost, makro i težinski prosjek. Točnost modela nad testnim uzorcima je iznosila 90.73%, preciznost za svaku gestu je između 74 i 100%, odziv je u rasponu od 80 do 100%, F1 vrijednost je 79 do 99% za svaku gestu, a makro prosjek i težinski prosjek iznose 91% za svaku od korištenih metrika po gesti. Matrica zabune je pokazala da će mreža ponekad zamijeniti geste čiji spektrogrami izgledaju slično, a obično je bila riječ o komplementarnim gestama *lijevo/desno, krug u smjeru/suprotnom smjeru od kazaljke na satu*.

Testiranje sustava je ukazalo i na nedostatke ove metode detekcije gesti ruku – veliki utjecaj okoline na snimljene signale i ljudski faktor. Utjecaj okoline se može smanjiti ukoliko se sustav postavi dalje od velikih prepreka koje će izazvati neželjene refleksije. Tu spadaju i ljudi koji dolaze preblizu sustavu dok se vrši detekcija. Ljudska pogreška se smanjuje ukoliko je korisnik relativno miran dok sustav vrši detekciju i ukoliko napravi gestu na odgovarajućoj udaljenosti od mikrofona. U suprotnom, svaku radnju koju osoba napravi ispred mikrofona, oni će ju zabilježiti i postoji mogućnost da će zabilježene refleksije sustav detektirati kao jednu od poželjnih 11 gesti. Još jedan nedostatak ovog pristupa je što se ne mogu pravilno detektirati uzorci u kojima je gesta napravljena jako sporo ili s malim rasponom pokreta.

U uvjetima bez prepreka u prostoru i bez prevelike neželjene aktivnosti korisnika, razvijeni sustav omogućuje dinamičnu detekciju gesti ruku visoke točnosti. Rad sustava u stvarnom vremenu je testiran primjenom na *Spotify Web* aplikaciju, pri čemu korisnici putem 11 definiranih gesti upravljaju ovim glazbenim servisom.

LITERATURA

[1] J. Weissmann, R. Salomon, *Gesture Recognition for Virtual Reality Applications Using Data Gloves and Neural Networks*, Proceedings of the IJCNN'99, International Joint Conference on Neural Networks. Proceedings (Cat. No.99CH36339), Washington, DC, USA, 10–16 July 1999; Volume 3, pp. 2043–2046.

[2] R. Xie, J. Cao, *Accelerometer-Based Hand Gesture Recognition by Neural Network and Similarity Matching*, IEEE Sens. J. 2016, 16, 4537–4545

[3] W. Ruan, Q. Z. Sheng, L. Yang, T. Gu, P. Xu, L. Shangguan, *AudioGest: Enabling Fine-Grained Hand Gesture Detection by Decoding Echo Signal*, UBICOMP '16, september 12–16, 2016, Heidelberg, Germany, DOI: 10.1145/2971648.2971736

[4] F. Adib, D. Katabi, *See through walls with WiFi!*, In Proceedings of the ACM SIGCOMM 2013 conference on SIGCOMM (SIGCOMM '13), 2013., Association for Computing Machinery, New York, NY, USA, 75–86. DOI: 10.1145/2486001.2486039

[5] R. Vehmas, N. Neuberger, *Inverse Synthetic Aperture Radar Imaging: A Historical Perspective and State-of-the-Art Survey*, IEEE Access, vol. 9, pp. 113917-113943, 2021, DOI: 10.1109/ACCESS.2021.3104799

[6] H. Abdelnasser, M. Youssef, K. A. Harras, *WiGest: A ubiquitous WiFi-based gesture recognition system*, 2015 IEEE Conference on Computer Communications (INFOCOM), Hong Kong, China, 2015., pp. 1472-1480, DOI: 10.1109/INFOCOM.2015.7218525

[7] S Tan, J Yang, *WiFinger: leveraging commodity WiFi for fine-grained finger gesture recognition*, In Proceedings of the 17th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc '16), 2016., Association for Computing Machinery, New York, NY, USA, 201–210, DOI: 10.1145/2942358.2942393

[8] K. Kalgaonkar, B. Raj. 2009. *One-handed gesture recognition using ultrasonic Doppler sonar*, In 2009 IEEE International Conference on Acoustics, Speech and Signal Processing, IEEE, 1889–1892

[9] S. Gupta, D. Morris, S. Patel, D. Tan, *Soundwave: using the doppler effect to sense gestures*, In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, 2012., ACM, 1911–1914

[10] R. Nandakumar, V. Iyer, D. Tan, S. Gollakota, *FingerIO: Using Active Sonar for Fine-Grained Finger Tracking*, CHI'16, May 7–12, 2016, San Jose, CA, USA., DOI:10.1145/2858036.2858580

[11] W. Wang, A. X. Liu, K. Sun, *Device-free gesture tracking using acoustic signals,* Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking, str. 82–94, New York City, NY, USA, 2016., DOI:10.1145/2973750.297376

[12] S. Yun, Y. C. Chen, H. Zheng, L. Qiu, W. Mao, *Strata: fine-grained acoustic-based device-free tracking,* in Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys), pp. 15–28, Niagara Falls, New York, USA, June 2017.

[13] N. Garg, N. Roy, (2020.), *Enabling Self-defense in Small Drones*, Proceedings of the 21st International Workshop on Mobile Computing Systems and Applications (HotMobile '20). Association for Computing Machinery, New York, NY, USA, 15–20, 2020., DOI:10.1145/3376897.3377866

[14] D. Sederberg, *The Doppler Effect with Sound,* Saturday Morning Astrophysics at Purdue, 2021., Purdue University

[15] Z. Mutin, *Doplerov efekt u akustici,* FIZIČAR – Fizika na dohvat ruke, 2023., dostupno na: https://fizicar.com/doplerov_efekat/ (Datum pristupa: 20.08.2024.)

[16] *zvuk*, Hrvatska enciklopedija, mrežno izdanje., Leksikografski zavod Miroslav Krleža, 2013. – 2024. dostupno na: https://enciklopedija.hr/clanak/zvuk (Datum pristupa: 19.08.2024.)

[17] D. Purves, G. J. Augustine, D. Fitzpatrick, L. C. Katz, A. LaMantia, J. O. McNamara, S. M. Williams, *Neuroscience*, treće izdanje, Sinauer Associates, Inc., 2004., poglavlje 12 – The Auditory System, str. 284., ISBN 0-87893-725-0

[18] *Reflection of Sound Waves,* Toppr, 2019., dostupno na: https://www.toppr.com/guides/physics/waves/reflection-of-sound-waves/ (Datum pristupa: 20.08.2024.)

[19] D. Willard (2018), *Understanding Microphone Patterns,* Azden, 2022., dostupno na: https://www.azden.com/blog/understanding-microphone-polar-patterns/#:~:text=Cardioid%20(kar%2Ddee%2Doid,sides%20(90%C2%BA%2F270%C2%BA) (Datum pristupa: 20.08.2024.)

[20] *White Shark DSM-02 Nagara korisnički priručnik*, White Shark, 2023.

[21] H. Austerlitz, *Data Acquisition Techniques Using PCs (Second Edition)*, poglavlje 4 – Analog/Digital Conversions, Elsevier Inc., 2003, stranice 51-77, ISBN: 978-0-12-068377-2

[22] *Short-time Fourier transform – MATLAB*, Mathworks, dostupno na: https://www.mathworks.com/help/signal/ref/stft.html#mw_0f51b808-a3c4-4a02-810f-f61e3b3bef43 (Datum pristupa: 20.08.2024.)

[23] *Short-Time Fourier Transform (Advanced Signal Processing Toolkit)*, Ni, 2023., dostupno na: https://www.ni.com/docs/en-US/bundle/labview-advanced-signal-processing-toolkit-api-ref/page/lvasptconcepts/aspt_stft.html#:~:text=The%20time%2Dfrequency%20resolution%20of,the%20optimal%20time%2Dfrequency%20resolution (datum pristupa: 20.08.2024.)

[24] Ahirwal, Balkrishan; Khadtare, Mahesh; Mehta, Rakesh, *FPGA based system for color space transformation RGB to YIQ and YcbCr,* International Conference on Intelligent and Advanced Systems 2007. IEEE, pp. 1345–1349, 2007., DOI:10.1109/ICIAS.2007.4658603

[25] C. Clabaugh, D. Myszewski, J. Pang, *Neural Networks – Neuron*, Eric Roberts' Sophomore College 2000 class "The Intellectual Excitement of Computer Science.", Stanford, 2019., dostupno na: https://cs.stanford.edu/people/eroberts/courses/soco/projects/neural-networks/index.html (Datum pristupa: 12.09.2024.)

[26] N. McCullum, *Understanding Neurons in Deep Learning*, NickCullum, 2024., dostupno na: https://www.nickmccullum.com/python-deep-learning/understanding-neurons-deep-learning/#what-is-a-neuron-in-deep-learning- (Datum pristupa: 12.09.2024.)

[27] S. Saxena, *Artificial Neuron Networks(Basics) | Introduction to Neural Networks*., Medium, 2017., dostupno na: https://becominghuman.ai/artificial-neuron-networks-basics-introduction-to-neural-networks-3082f1dcca8c (Datum pristupa: 12.09.2024.)

[28] *Introduction to Convolution Neural Network*, GeeksforGeeks, 2024., dostupno na: https://www.geeksforgeeks.org/introduction-convolution-neural-network/ (Datum pristupa: 12.09.2024.)

[29] J. Cardete, *Convolutional Neural Networks: A Comprehensive Guide,* The Deep Hub, 2024., dostupno na: https://medium.com/thedeephub/convolutional-neural-networks-a-comprehensive-guide-5cc0b5eae175 (Datum pristupa: 12.09.2024.)

[30] R. Parmar, *Training Deep Neural Networks*, Medium, 2018., dostupno na: https://towardsdatascience.com/training-deep-neural-networks-9fdb1964b964 (Datum pristupa: 12.09.2024.)

[31] *What are Convolutional Neural Networks?*, IBM, dostupno na: https://www.ibm.com/topics/convolutional-neural-networks (Datum pristupa: 12.09.2024.)

[32] V. Bhagat, *Overview of Convolutional Neural Networks*, Toppr, 2022., dostupno na: https://www.topcoder.com/thrive/articles/overview-of-convolutional-neural-networks?utm_source=thrive&utm_campaign=thrive-feed&utm_medium=rss-feed (Datum pristupa: 12.09.2024.)

[33] Y. Gavrilova, *What Are Convolutional Neural Networks?*, Serokell Software Development Company, 2021., dostupno na: https://serokell.io/blog/introduction-to-convolutional-neural-networks (Datum pristupa: 12.09.2024.)

[34] R. Qayyum, *Introduction To Pooling Layers In CNN*, Medium, 2022., dostupno na: https://pub.towardsai.net/introduction-to-pooling-layers-in-cnn-dafe61eabe34 (Datum pristupa: 12.09.2024.)

[35] A. F. Gad, *Evaluating Deep Learning Models: The Confusion Matrix, Accuracy, Precision, and Recall*, DigitalOcean, 2024., dostupno na: https://www.digitalocean.com/community/tutorials/deep-learning-metrics-precision-recall-accuracy (Datum pristupa: 12.09.2024.)

[36] S. Šegvić, *Učenje sličnosti – metrička ugrađivanja složenih podataka*, FER, 2024.

[37] N. Beheshti, *Guide to Confusion Matrices & Classification Performance Metrics – Accuracy, Precision, Recall, & F1 Score*, Medium, 2022., dostupno na: https://towardsdatascience.com/guide-to-confusion-matrices-classification-performance-metrics-a0ebfc08408e (Datum pristupa: 12.09.2024.)

[38] M. Natarajan, *Understanding the Intuition behind Classification Report Metrics: Macro Average vs. Weighted Average, F1 Score and Beyond*, Medium, 2023., dostupno na: https://medium.com/@megha.natarajan/understanding-the-intuition-behind-classification-report-metrics-macro-average-vs-d9c2cc04717d (Datum pristupa 17.09.2024.)

[39] R. Kundu, *F1 Score in Machine Learning: Intro & Calculation*, V7, 2022., dostupno na: https://www.v7labs.com/blog/f1-score-guide (Datum pristupa 17.09.2024.)

[40] J. Mathew, R. Kshirsagar, D. Z. Abidin, J. Griffin, S. Kanarachos, M. Alamaniotis, M. E. Fitzpatrick, *A comparison of machine learning methods to classify radioactive elements using*

prompt-gammaray neutron activation dana, Research Square, 2023., DOI: 10.21203/rs.3.rs-2518432/v1

[41] A. Ibrahim, A. El-Refai, S. Ahmed, M. Aboul-Ela. H. M. Eraqi, M. Moustafa, *Pervasive Hand Gesture Recognition for Smartphones using Non-audible Sound and Deep Learning*, Department of Computer Science and Engineering, The American University in Cairo, New Cairo, Egypt, 2021., DOI: arXiv:2108.02148v1

[42] J. Cheon, S. Choi, *Hand Gesture Classification using Inaudible Sound with Ensemble Method*, Advances in Science, Technology and Engineering Systems Journal Vol. 5, No. 6, 967-971 (2020), Special Issue on Multidisciplinary Sciences and Engineering, ISSN: 2415-6698

[43] J. Kim, S. Choi, *Hand Gesture Classification Based on Nonaudible Sound Using Convolutional Neural Network*, Kookmin University, Seoul, Republic of Korea, 2019., DOI: 10.1155/2019/1084841

[44] Q. Zeng, Z. Kuang, S. Wu, J. Yang, *A Method of Ultrasonic Finger Gesture Recognition Based on the Micro-Doppler Effect*, Univerisity of Chinese Academy of Sciences, Beijing, China, 2019. DOI: 10.3390/app9112314

[45] X. Xu, X. Zhang, Z. Bao, X. Yu, Y. Yin, X. Yang, Q. Niu, *Training-Free Acoustic-Based Hand Gesture Tracking on Smart Speakers*, Appl. Sci. 2023, 13, 11954., DOI: 10.3390/app132111954

[46] Z. Wang, Y. Hou, K. Jiang, W. Dou, C. Zhang, Z. Huang, Y. Guo, *Hand Gesture Recognition Based on Active Ultrasonic Sensing of Smartphone: A Survey*, Shandong University of Science and Technology, Qingdao, China, IEEE Access (2017), DOI: 10.1109/ACCESS.2019.2933987

[47] B. S. Moreia, A. Perkusich, S. O. D. Luiz, *An Acoustic Sensing Gesture Recognition System Design Based on a Hidden Markov Model*, Sensors 2020, Vol. 20, Issue 17, DOI:10.3390/s20174803

[48] M. Schak, A. Gepperth, *Gesture Recognition on a New Multi-Modal Hand Gesture Dataset*, Proceedings of the 11th International Conference on Pattern Recognition Applications and Methods (ICPRAM 2022), pages 122-131, ISBN: 978-989-758-549-4; ISSN: 2184-4313, DOI: 10.5220/0010982200003122

[49] J. Rajalakshmi, P. Kumar, *Gesture Recognition using CNN and RNN*, International Journal of Recent Technology and Engineering (IJRTE), ISSN: 2277-3878 (Online), Volume-9 Issue-2, July 2020, DOI:10.35940/ijrte.B3417.079220

[50] W. Wu, *Responsive audio feedback for hand gesture interaction to enhance immersion in audio-only games*, Faculty of Drexel University, 2015., DOI: 10.17918/etd-6521

[51] N. Bolf, *Osvježimo znanje: Strojno učenje, Kemija u industriji*, 70(9-10), str. 591-593., 2021., Dostupno na: https://hrcak.srce.hr/263495 (Datum pristupa: 19.08.2024.)

[52] *librosa.stft — librosa 0.10.2.post1 documentation*, Librosa, dostupno na: https://librosa.org/doc/main/generated/librosa.stft.html (Datum pristupa: 20.08.2024.)

[53] R.E. Berg, *Ultrasonics | Physics, Sound Waves & Applications*, Encyclopaedia Britannica, 2024., dostupno na: https://www.britannica.com/science/ultrasonics/Ranging-and-navigating (Datum pristupa: 19.08.2024.)

[54] *Doppler Effect*, Toppr, 2021., dostupno na: https://www.toppr.com/guides/physics/waves/doppler-effect/ (Datum pristupa: 19.08.2024.)

[55] G. Brown, *Digital Audio Basics: Sample Rate and Bit Depth*, Izotope, 2019., dostupno na: https://www.izotope.com/en/learn/digital-audio-basics-sample-rate-and-bit-depth.html (Datum pristupa: 05.09.2024.)

SAŽETAK

Prepoznavanje i razlikovanje gesti ruku korisnika postalo je važno zbog velikog porasta upotrebe uređaja nosive tehnologije (engl. *wearables*), poput pametnog sata. Jedan od načina detekcije gesti ruku korisnika je korištenjem nečujnog audio-signala na način da se takav signal generira zvučnikom te se analizira reflektirani signal korištenjem mikrofona. U okviru ovog rada razvijen je algoritam zasnovan na dubokom učenju koji koristi komercijalne zvučnike i mikrofone za generiranje nečujnog audio-signala i analizu reflektiranog nečujnog audio-signala. Nadalje, kreirana je baza nečujnih audio-signala gesti ruku na kojoj je neuronska mreža trenirana i testirana. Na kraju su testirane performanse izrađenog algoritma te je implementiran za rad u stvarnom vremenu.

Ključne riječi: Dopplerov efekt, detekcija gesti ruku, STFT, CNN, klasifikacija, Spotify Web

HAND GESTURES DETECTION BASED ON NON-AUDIBLE SOUND

ABSTRACT

Recognizing and differentiating user hand gestures has become important due to the large increase in wearable technology device usage, such as smartwatches. One of the ways to detect user hand gestures is to generate an inaudible audio signal through speakers and analize the reflected signal using microphones. In this paper, an algorithm based on deep learning was developed, that uses commercial speakers and microphones to generate an inaudible audio signal and analize its reflected audio signal. Furthermore, a database of inaudible audio signals was created and used for neural network training and testing. Finally, the algorithm's performance was tested and implemented in real time.

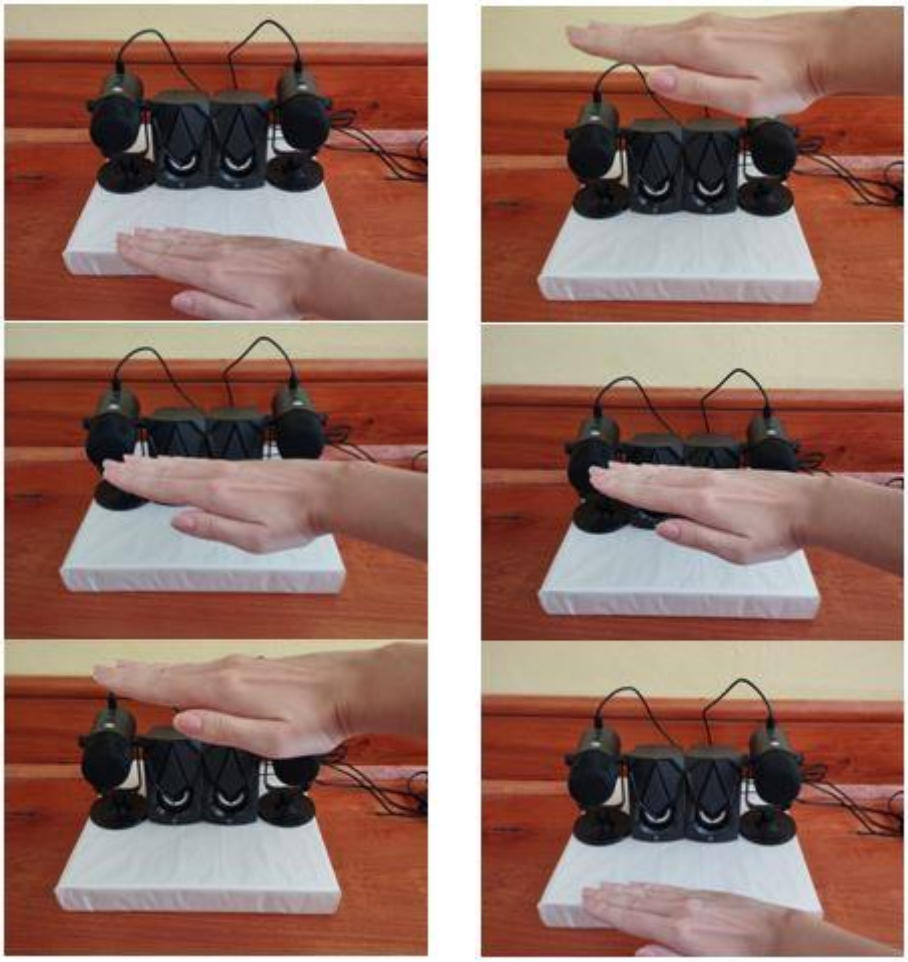
Keywords: Doppler effect, hand gesture recognition, STFT, CNN, classification, Spotify Web

ŽIVOTOPIS

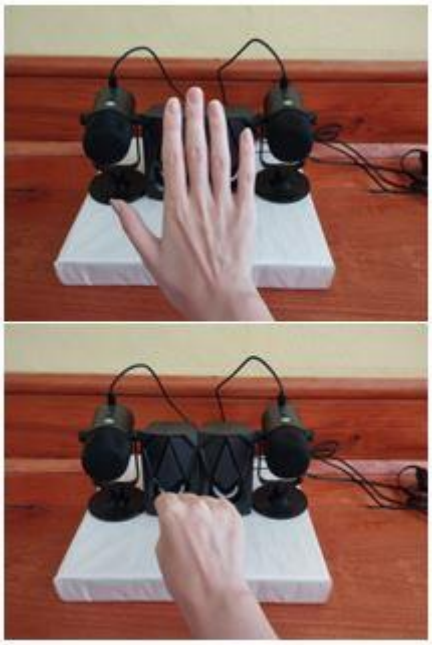
Ena Jurkić, rođena 20.05.1998. u Osijeku, završava Osnovnu školu Ivana Kukuljevića u Belišću paralelno s Osnovnom glazbenom školom Matije Petra Katančića u Valpovu. Nakon završetka osnovne škole pohađa Opću gimnaziju u Valpovu koju završava s odličnim uspjehom. Tijekom školovanja dobiva nagradu za izvrsnost, sudjeluje na natjecanjima iz matematike, fizike, geografije, stranih jezika i povijesti, a osvaja nagrade iz područja glazbe i likovnih umjetnosti. Daljnje obrazovanje nastavlja upisom preddiplomskog studija Elektrotehnike i informacijske tehnologije na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija u Osijeku. Od 2021. godine na istom fakultetu obavlja studentski posao demonstratora u laboratoriju za fiziku. 2022. godine sudjeluje na projektu „Učimo kroz igru“ u organizaciji udruge Igra znanja s ciljem razvoja videoigre u svrhu poticanja mladih na STEM zanimanja. Te iste godine završava „Seeds for the Future“ program u organizaciji tvrtke Huawei, sudjeluje na GoSTEM sajmu u organizaciji FERIT-a kao voditelj radionica u području fizike, završava preddiplomski studij i upisuje diplomski studij Elektrotehnike i informacijske tehnologije, smjer Komunikacije i informatika, modul Mrežne tehnologije. 2023. godine dio je delegacije sveučilišta J.J. Strossmayera u Osijeku tijekom posjeta sjedištu tvrtke Huwei u Kini radi realizacije projekta „Huawei for Digital Slavonia“ između tvrtke Huawei, Fakulteta agrobiotehničkih znanosti i FERIT-a. Te iste godine sudjeluje na konferenciji za kibernetičku sigurnost „Cybersecurity Conference – CSC 23“ koju organizira Hrvatski institut za kibernetičku sigurnost i postaje stipendist tvrtke TTTech Auto d.o.o.

PRILOZI

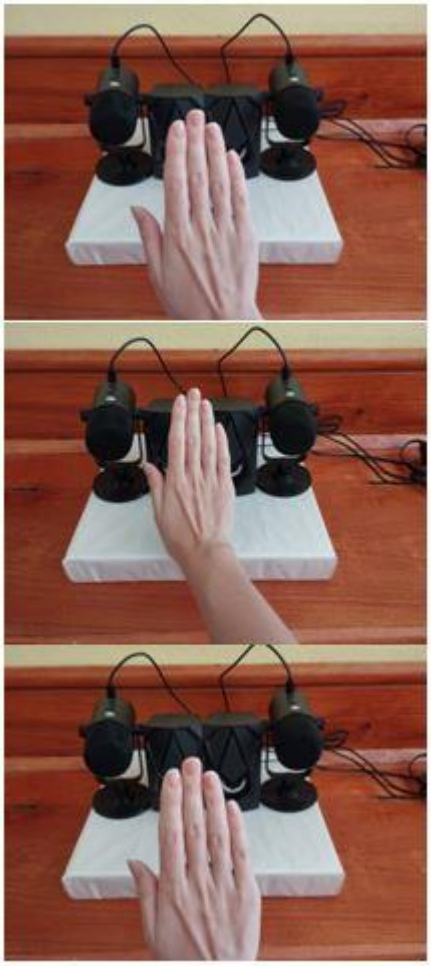
Prilog P.3.1. Prikaz izvođenja gesti



a) gesta *gore* b) gesta *dolje*



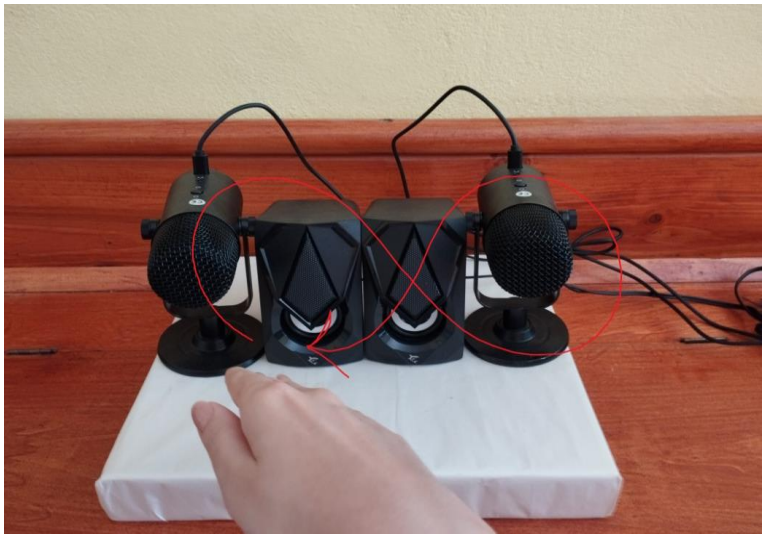
a) gesta *zatvorena šaka*



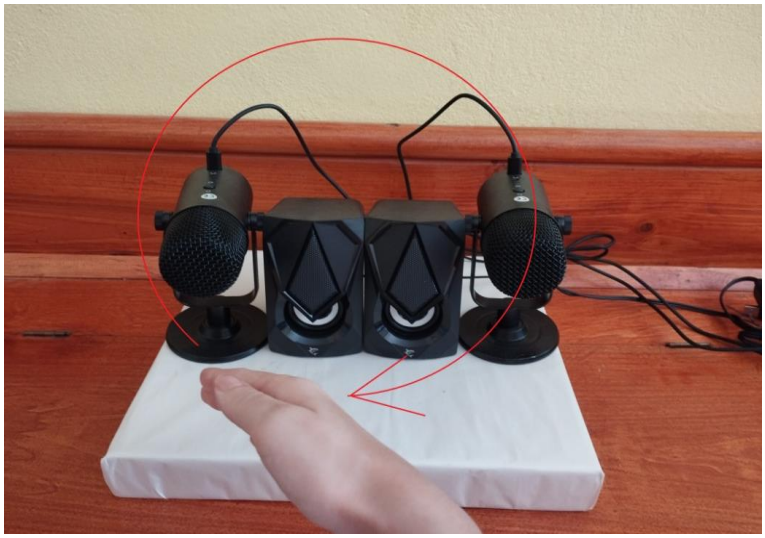
b) gesta *prema*

gesta *desno*

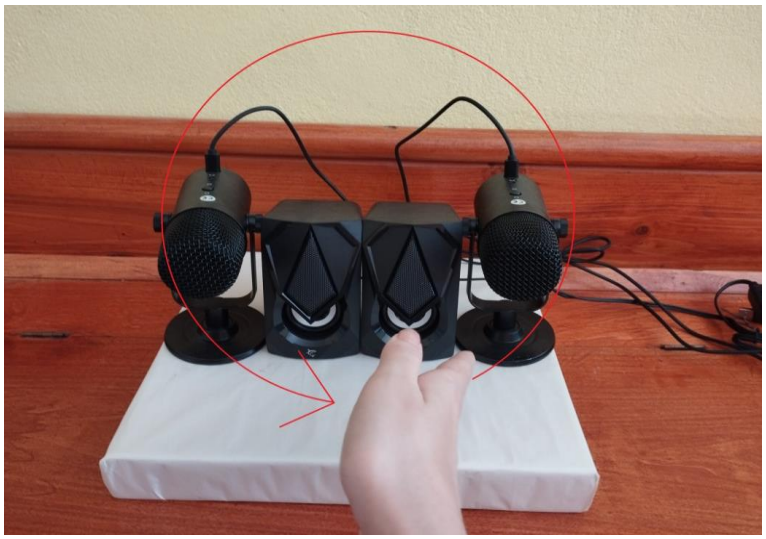
gesta *lijevo*



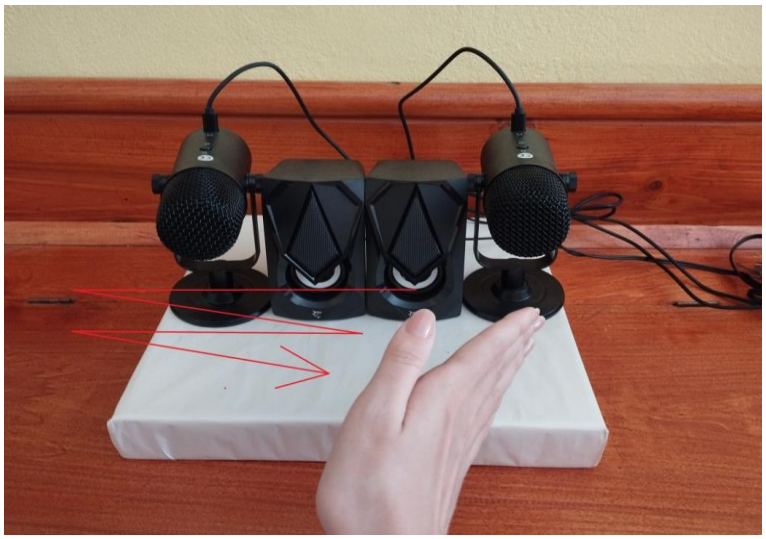
gesta *beskonačno*



gesta *krug u smjeru kazaljke na satu*



gesta *krug u smjeru suprotnom od kazaljke na satu*



gesta *shuffle*

Prilog P.3.2. Dio *Python* skripte za generiranje signala 20 kHz

```
def generate_sine_wave(frequency, samplerate):
    t = np.linspace(0, 1, int(samplerate), endpoint=False)
    wave = 0.8*np.sin(2*np.pi*frequency*t).astype(np.float32)


def play_continuous_sine_wave(frequency,samplerate,stop_event):
    stream = sd.OutputStream(samplerate=samplerate,channels=1,
dtype='float32',blocksize=8192)
    stream.start()
    generator = generate_sine_wave(frequency, samplerate)
```

Prilog P.3.3. Dio *Python* skripte za snimanje signala s oba mikrofona istovremeno i konverziju u stereo signal

```
def record_device_continuous(device_id, audio_q, stop_event, sr,
channels):
    def audio_callback(indata, frames, time, status):
        if status:
            print(status, file=sys.stderr)
        if not stop_event.is_set():
            audio_q.put(indata.copy())
    blocksize = 8192
    with sd.InputStream(samplerate=sr, device=device_id,
channels=channels, callback=audio_callback, blocksize=blocksize):
        while not stop_event.is_set():
            time.sleep(0.1)


def conversion_to_stereo(left_channel, right_channel):
    max_length = max(len(left_channel), len(right_channel))
    left_channel_padded = np.pad(left_channel,(0, max_length -
len(left_channel)))
    right_channel_padded = np.pad(right_channel,(0, max_length -
len(right_channel)))
    stereo_signal = np.column_stack((left_channel_padded,
right_channel_padded))
    return stereo_signal
```

Prilog P.3.4. Dio *Python* skripte za konverziju stereo signala u spektrogram

```
def convert_audio_to_png(input_folder_spectrogram,
output_folder_spectrogram):
    os.makedirs(output_folder_spectrogram, exist_ok=True)
    for filename in os.listdir(input_folder_spectrogram):
        input_path = os.path.join(input_folder_spectrogram, filename)
        output_path = os.path.join(output_folder_spectrogram,
f"{os.path.splitext(filename)[0]}.png")


        y, sr = librosa.load(input_path, sr=44100)
        n_fft = 8192
        hop_length = 512
        D = librosa.amplitude_to_db(np.abs(librosa.stft(y,
n_fft=n_fft, hop_length=hop_length)), ref=np.max)
```

Prilog P.3.5. Dio *Python* skripte za ROI

```
def preprocess_image(image):
    image_gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    roi = image_gray[48:353, 200:650]
    roi_resized = cv2.resize(roi, (530, 310))
    roi_resized = np.expand_dims(roi_resized, axis=-1)
    roi_resized = np.expand_dims(roi_resized, axis=0)
    return roi_resized
```

Prilog P.3.6. Dio *Python* koda kojim je definirana struktura modela neuronske mreže

```
model = Sequential()
model.add(Conv2D(4, (3, 3), activation='relu', input_shape=(420, 310, 1)))
model.add(MaxPooling2D(pool_size=(2, 2), strides = (2, 2)))
model.add(Conv2D(8, (3, 3), activation='relu', padding = 'same'))
model.add(MaxPooling2D(pool_size=(2, 2), strides = (2, 2)))
model.add(Conv2D(16, (3, 3), activation='relu', padding = 'same'))
model.add(MaxPooling2D(pool_size=(2, 2), strides = (2, 2)))
model.add(Conv2D(32, (3, 3), activation='relu', padding = 'same'))
model.add(MaxPooling2D(pool_size=(2, 2), strides = (2, 2)))
model.add(Conv2D(64, (3, 3), activation='relu', padding = 'same'))
model.add(MaxPooling2D(pool_size=(2, 2), strides = (2, 2)))
model.add(Conv2D(128, (3, 3), activation='relu', padding = 'same'))
model.add(MaxPooling2D(pool_size=(2, 2), strides = (2, 2)))
model.add(Conv2D(256, (3, 3), activation='relu', padding = 'same'))
model.add(MaxPooling2D(pool_size=(2, 2), strides = (2, 2)))
model.add(Conv2D(512, (3, 3), activation='relu', padding = 'same'))
model.add(AveragePooling2D(pool_size=(2, 2), strides = (2, 2)))
model.add(Flatten())
model.add(Dropout(0.2))
model.add(Dense(11, activation='softmax'))
```

Prilog P.3.7. Dio *Python* skripte za treniranje mreže

```
X_train, X_temp, y_train, y_temp = train_test_split(X, y, test_size=0.3, random_state=42, stratify=y)
X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp, test_size=0.3333, random_state=42, stratify=y_temp)
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
history = model.fit(X_train, y_train, epochs=30, batch_size=32, validation_data=(X_val, y_val))
test_loss, test_accuracy = model.evaluate(X_test, y_test)
y_pred = model.predict(X_test)
```

```
y_pred_classes = np.argmax(y_pred, axis=1)
y_true = np.argmax(y_test, axis=1)
cm = confusion_matrix(y_true, y_pred_classes)
```

Prilog P.3.8. Dio *Python* skripte za detekciju gesti u stvarnom vremenu

```
def spectrogram_and_predict_thread(audio_q1, audio_q2, stop_event, sr,
n_fft, hop_length, win_length, window, output_folder_spectrogram):
     detection_history = []
     buffer_lock = Lock()

     if not initial_spectrogram_created:
               initial_spectrogram =
     librosa.amplitude_to_db(np.abs(librosa.stft(stereo_signal.T[0] +
     stereo_signal.T[1], n_fft=n_fft, hop_length=hop_length,
     win_length=win_length, window=window)), ref=np.max)
          previous_spectrogram = initial_spectrogram
          initial_spectrogram_created = True
     else:
          new_audio_chunk = stereo_signal[-chunk_size:]
          D_new = librosa.amplitude_to_db(np.abs(librosa.stft
     (new_audio_chunk.T[0] + new_audio_chunk.T[1], n_fft=n_fft,
     hop_length=hop_length, win_length=win_length, window=window)),
     ref=np.max)
         D_shifted = previous_spectrogram[:, D_new.shape[1]:]
         previous_spectrogram = np.hstack((D_shifted, D_new))

     processed_image = preprocess_image(filtered_spectrogram)
     prediction = model.predict(processed_image)
     gesture_index = np.argmax(prediction, axis=1)[0]
     gesture = gesture_labels[gesture_index]
     print(f"Detected gesture: {gesture}")
     detection_history.append(gesture)

     if len(detection_history) >= 3:
        if detection_history[-3:] == [gesture] * 3:
```

```
            print(f"Confirmed gesture: {gesture}")
            control_spotify(gesture)
            detection_history = []
        else:
            detection_history.pop(0)
```

Prilog P.4.1. Dio *Python* skripte za kontrolu *Spotify Web* aplikacije

```
def volume_up():
    playback = sp.current_playback()
    if playback is not None:
        current_volume = playback['device']['volume_percent']
        sp.volume(min(current_volume + 5, 100))
    else:
        print("No active playback device found.")
def volume_down():
    playback = sp.current_playback()
    if playback is not None:
        current_volume = playback['device']['volume_percent']
        sp.volume(max(current_volume - 5, 0))
    else:
        print("No active playback device found.")
def next_song():
    sp.next_track()
def previous_song():
    sp.previous_track()

gesture_to_control = {
    'dolje': volume_down,
    'gore': volume_up,
    'desno': next_song,
    'lijevo': previous_song,
    'prema': pause_play,
    'krug-smjer': repeat_song,
    'krug-suprotno': start_from_beginning,
```

```
    'shuffle': shuffle_playlist,
    'beskonacno': repeat_playlist,
    'zatvorena saka': mute
}

def control_spotify(gesture):
    if gesture in gesture_to_control:
        gesture_to_control[gesture]()
    else:
        print(f"Gesture '{gesture}' not recognized.")
```

Prilog P.4.2. Autentikacija korisnika i područje dozvoljenih aktivnosti pri upotrebi aplikacije *Spotify Web*

```
sp = spotipy.Spotify(auth_manager=SpotifyOAuth(client_id='**********',
                     client_secret='**********',
                     redirect_uri='http://localhost:65026',
                    scope='user-modify-playback-state,user-read-
          playback-state'))
```