UNIVERZITET U NIŠU
ELEKTRONSKI FAKULTET

AUGMENTACIJA PODATAKA KOD AUDIO ZAPISA

SEMINARSKI RAD

Predmet:

Prikupljanje i predobrada podataka za mašinsko učenje

Mentor: prof. dr Aleksandar Stanimirović

Kandidat: Milica Stojanović (1701)

Niš, 2024.

Sadržaj

1. Uvod

U savremenom svetu veštačke inteligencije i mašinskog učenja, kvalitativna analiza i obrada audio podataka postaju sve važniji aspekti u različitim aplikacijama, od prepoznavanja govora i muzike, preko analize zvuka u medicinskim istraživanjima, do automatske detekcije zvukova u industrijskim okruženjima. Ključni faktor uspeha u ovim aplikacijama je obim i raznovrsnost podataka koji se koriste za obuku modela. Ipak, često se suočavamo sa problemom nedostatka dovoljno velikih i raznovrsnih skupova podataka, što može dovesti do smanjenih performansi modela i otežati generalizaciju na nove, neviđene podatke.

Jedan od najefikasnijih pristupa za prevazilaženje ovog izazova je tehnika poznata kao augmentacija podataka (data augmentation). Augmentacija podataka obuhvata niz metoda koje se koriste za generisanje novih primera podataka iz postojećih, čime se povećava raznovrsnost skupa podataka i poboljšavaju performanse modela. U domenu audio obrade, augmentacija podataka uključuje različite tehnike kao što su dodavanje šuma, promena visine tonova, promene brzine reprodukcije, dodavanje pozadinskog šuma i mnoge druge. Ove tehnike omogućavaju modelima da se obučavaju na bogatijim i raznovrsnijim setovima podataka, što može značajno poboljšati njihovu sposobnost da prepoznaju obrasce i pravilnosti u stvarnim uslovima.

Međutim, primena tehnika augmentacije podataka nije bez svojih izazova. U nekim slučajevima, nepravilno korišćenje augmentacije može dovesti do degradacije kvaliteta podataka ili čak do pogoršanja performansi modela. Na primer, prekomerna promena visine tonova može učiniti audio zapise manje prepoznatljivim, dok dodavanje šuma može smanjiti kvalitet zvuka i otežati prepoznavanje ključnih informacija. Stoga je važno pažljivo birati i prilagođavati tehnike augmentacije u skladu sa specifičnostima zadatka i karakteristikama podataka.

Ovaj rad ima za cilj da istraži i analizira različite tehnike augmentacije za audio zapise, sa posebnim fokusom na njihove principe, prednosti i nedostatke. Kroz detaljnu analizu, biće razmotreni osnovni principi i metodologije koje stoje iza različitih tehnika augmentacije, kao i uslovi pod kojima su one najefikasnije. Takođe, biće pružene preporuke za najbolju praksu u primeni ovih tehnika kako bi se maksimizovale koristi i minimizirali potencijalni problemi. Rad će se takođe baviti praktičnim primerima i studijama slučaja kako bi se ilustrirali efekti augmentacije podataka na performanse modela u stvarnim aplikacijama.

2. Identifikacija problema

Ograničena dostupnost podataka: U mnogim realnim aplikacijama, pribavljanje velikih, visokokvalitetnih skupova podataka može biti izazovno, vremenski zahtevno i skupo. Augmentacija podataka predstavlja efikasan način da se poveća veličina i raznolikost podataka za treniranje, omogućavajući razvoj preciznijih i robusnijih modela. Ovo je posebno izraženo u domenima audio klasifikacije, gde je često teško prikupiti dovoljan broj zvučnih uzoraka u različitim okruženjima i scenarijima.[6]

Na primer, ako se razvija model za prepoznavanje govora u bučnim okruženjima, često je teško prikupiti dovoljno snimaka sa različitim tipovima pozadinske buke. Augmentacija dodavanjem veštačkih zvukova buke može značajno proširiti obim trening skupa, omogućavajući modelu da bolje prepoznaje govor u različitim uslovima.

Prevencija prenaučenosti (overfitting): Dodavanje varijacija u podatke za treniranje pomaže u prevenciji prenaučenosti, uobičajenom problemu u mašinskom učenju gde modeli dobro performišu na podacima za treniranje, ali loše generalizuju na neviđene podatke. U audio klasifikaciji, augmentacija može uključivati promene u brzini, tonu, ili dodavanje šuma, što omogućava modelima da se upoznaju sa različitim varijacijama zvuka. Na primer, ako model trenira prepoznavanje zvukova u prirodi, primena tehnike kao što je dodavanje pozadinskog šuma može pomoći modelu da prepozna te zvukove čak i kada su prisutne varijacije u uslovima snimanja, čime se smanjuje verovatnoća prenaučenosti.[6]

Poboljšanje performansi modela: Augmentacija podataka može značajno poboljšati performanse modela mašinskog učenja, naročito u zadacima poput prepoznavanja zvukova, obrade govora i muzike. Dodavanjem različitih primera za učenje, augmentacija pomaže modelu da bolje ekstraktuje karakteristike i poveća tačnost. Na primer, primena tehnike vremenskog rastezanja zvuka može pomoći modelu da prepozna iste zvučne događaje u različitim brzinama, poboljšavajući njegovu sposobnost da generalizuje na nove primere. Ovo može biti korisno za zadatke kao što su prepoznavanje muzičkih instrumenata ili identifikacija zvučnih signala u videosnimcima.[6]

Ublažavanje nebalansiranosti klasa: U mnogim klasifikacionim problemima, raspodela klasa u trening podacima može biti nebalansirana, pri čemu su neke klase zastupljene manje od drugih. Augmentacija se može koristiti za generisanje dodatnih uzoraka za manjinske klase, čime se smanjuje uticaj nebalansiranosti i poboljšava sposobnost modela da pravilno klasifikuje nedovoljno zastupljene klase. Na primer, ako model treba da prepozna zvuke različitih ptica, a neki zvuci su veoma retki u skupu podataka, augmentacija kao što je dodavanje varijacija u zvukovima ptica može pomoći u ravnoteži raspodele klasa i poboljšati tačnost prepoznavanja.[6]

Transfer učenje: Augmentacija može takođe koristiti transfer učenju, gde se prethodno trenirani model fino podešava na manjem skupu podataka za specifičan zadatak. Korišćenjem augmentacije na ciljnom skupu podataka, model može naučiti robusnije reprezentacije, poboljšavajući svoje performanse na novom zadatku. Na primer, ako se koristi model treniran za prepoznavanje govora u jednom jeziku i fino podešava za prepoznavanje u drugom jeziku sa ograničenim brojem snimaka, augmentacija može povećati raznovrsnost tog malog skupa podataka, omogućavajući bolju prilagodbu i tačnost modela.[6]

Ukratko, augmentacija audio podataka pomaže u prevazilaženju problema ograničenosti podataka, prenaučenosti, poboljšava performanse modela, ublažava nebalansiranost klasa i doprinosi boljoj primeni transfer učenja, čime se postiže robusniji i precizniji model u raznim audio analitičkim zadacima.

3. Reprezentacija audio zapisa

Reprezentacija audio signala igra ključnu ulogu u analizi i obradi zvuka. Zvukovi, uključujući govor, muziku i druge zvučne fenomene, mogu se predstaviti na različite načine kako bi se omogućila detaljna analiza i interpretacija. Svaka metoda reprezentacije pruža specifičan uvid u karakteristike zvuka, što omogućava primenu različitih tehnika za poboljšanje performansi modela u oblasti veštačke inteligencije i obrade signala.

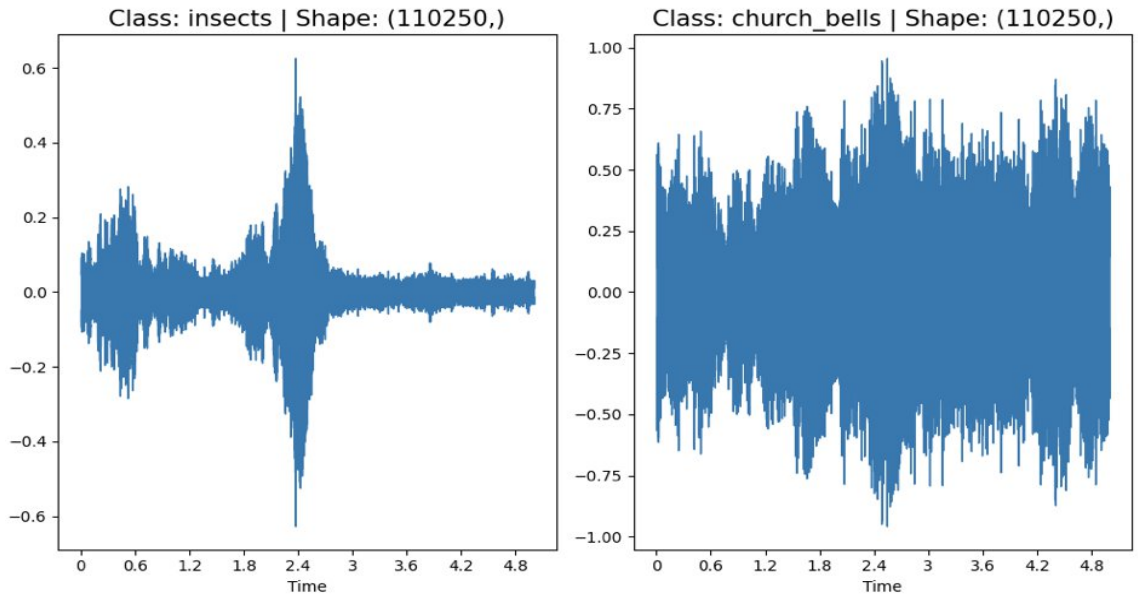
3.1. Talasni Oblik

Talasni oblik je osnovna metoda vizualizacije audio signala koja prikazuje promene u amplitudi zvuka tokom vremena. Na grafu talasnog oblika, x-osa predstavlja vreme, dok y-osa predstavlja amplitudu zvuka. Ovaj prikaz direktno odražava stvarni zvuk i omogućava lako prepoznavanje amplitudskih promena i osnovne strukture zvuka.[12]

Funkcija *librosa.display.waveshow* prikazuje talasni oblik audio signala koristeći *matplotlib*. Parametar *y* je niz podataka koji predstavlja audio signal. Uobičajeno je to *numpy* niz (array) koji sadrži amplitude zvuka tokom vremena. Ovaj niz se obično dobija iz funkcije *librosa.load*, koja učitava audio fajl u *y* i vraća njegov *sample rate*. Parametar *sr* je sample rate (uzorkovanje) audio signala, tj. broj uzoraka po sekundi. Ovaj parametar omogućava *librosa.display.waveshow* da pravilno prikaže talasni oblik u vremenskom domenu.

```
#plot the wave
librosa.display.waveshow(y,sr=sr,ax=ax[i])
```

Talasni oblik je jednostavan za interpretaciju jer vizualizuje stvarni signal u njegovom vremenskom domenu. To omogućava analizu osnovnih karakteristika zvuka, kao što su dinamika i trajanje, ali može biti manje efikasan za analizu složenih frekvencijskih komponenti. U slučajevima kada je potrebno detaljno proučiti kako se frekvencije menjaju tokom vremena, talasni oblik može biti ograničen, pa se često koristi zajedno s drugim metodama reprezentacije, kao što su spektrogrami ili mel spektrogrami, za obuhvatnije razumevanje zvučnog signala.



Slika 3.1.1. Talasni oblik reprezentacije audio zapisa

3.2. Spektrogram

Spektrogram je grafička reprezentacija frekvencijskog sadržaja audio signala tokom vremena. Predstavlja se kao 2D slika, gde x-osa predstavlja vreme, y-osa predstavlja frekvenciju, a intenzitet boje ili nijanse sive prikazuje amplitudu frekvencijskih komponenti u određenom trenutku.

Da bi se dobio spektrogram, koristi se Kratkoročna Fourierova Transformacija (STFT). Ova tehnika deli audio signal na kratke segmente i primenjuje Fourierovu transformaciju na svaki segment posebno. Rezultat ove transformacije su kompleksni brojevi koji predstavljaju frekvencijski sadržaj zvuka za svaki segment. Ovi rezultati se zatim vizualizuju kao spektrogram.[13]

Spektrogram omogućava detaljnu analizu frekvencijskog sadržaja zvuka i pruža uvid u promene u frekvencijama tokom vremena. Ova metoda je korisna za identifikaciju uzoraka u zvuku, kao što su tonovi, timbre, i transijentni događaji. Takođe, omogućava analizu karakteristika zvučnih signala, kao što su spektralna envelope i harmonska struktura. Spektrogram se često koristi u različitim audio analitičkim zadacima, uključujući prepoznavanje govora, odvajanje audio izvora, i detekciju audio događaja.[13]

Mel Spektrogram

Mel spektrogram je specifičan tip spektrograma koji koristi mel skalu za prikaz frekvencija. Mel skala je logaritamska skala koja odražava ljudsku percepciju visine tona, pružajući bolju interpretaciju zvuka u skladu sa načinom na koji ljudsko uvo percipira frekvencije. U mel spektrogramu y-osa predstavlja frekvenciju na mel skali, x-osa predstavlja vreme, a svetlost slike odgovara amplitudi zvuka na svakoj frekvenciji.[12]

Mel spektrogram se dobija primenom Fourierove transformacije na audio signal, a zatim se rezultantne frekvencije mapiraju na mel skalu. Mel skala je osmišljena da bude slična načinu na koji ljudsko uvo razlikuje frekvencije. Na primer, u linearnom spektrogramu, razmak između 1.000 i 2.000 Hz je polovina razmaka između 2.000 i 4.000 Hz. U mel spektrogramu, razmak između ovih opsega je približno isti. Ova logaritamska skala omogućava bolju analizu frekvencija koje su važnije za ljudsku percepciju, naročito kod niskih frekvencija gde su promene u tonu lakše prepoznatljive.[13]

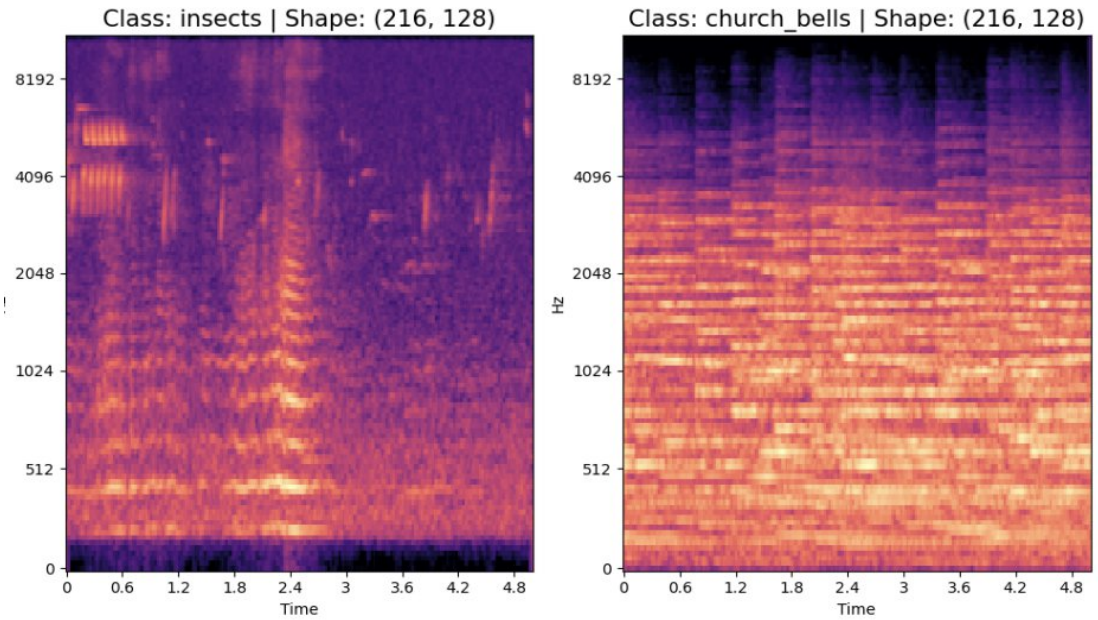
Za kreiranje Mel spektrograma audio signala koristi se funkcija *melspectrogram* iz *librosa* biblioteke. Parametar *y* predstavlja audio signal koji je učitan kao niz uzoraka. Parametar *sr* je brzina uzorkovanja (sampling rate) zvučnog signala u Hercima (Hz). *n_fft=2048*: Ova vrednost određuje veličinu prozora u kojem se primenjuje Fourierova transformacija. *hop_length*=512 je broj uzoraka između početaka uzastopnih prozora. *n_mels*=128 određuje broj mel frekvencijskih traka koje će biti korišćene za mapiranje frekvencija.

Funkcija *power_to_db* konvertuje moć (power) spektrograma u decibele. ref=np.max postavlja maksimalnu vrednost moći na referencu za 0 dB. Drugim rečima, vrednosti u spektrogramu će biti normalizovane tako da maksimalna vrednost bude 0 dB, a sve ostale vrednosti će biti izražene u odnosu na ovu referencu.

Funkcija *librosa.display.specshow* se koristi za vizualizaciju spektrograma, uključujući i Mel spektrogram. Prikazuje spektrogram kao sliku gde boje ili intenziteti predstavljaju amplitudu (nivo signala) u različitim frekvencijskim opsezima tokom vremena.[13]

```
#get the melspec in db before plotting
melspec = librosa.feature.melspectrogram(y=y, sr=sr, n_fft=2048, hop_length=512, n_mels=128)
melspec = librosa.power_to_db(melspec, ref=np.max)

#plot the melspec
librosa.display.specshow(melspec,sr=sr,ax=ax[i], y_axis='mel', x_axis='time')
```

Mel spektrogram je koristan za zadatke kao što su prepoznavanje govora i analiza muzike, jer bolje odražava ljudsku percepciju zvuka. Ovaj pristup može biti posebno koristan za prepoznavanje uzoraka u zvučnim signalima i za modeliranje karakteristika zvuka koje su ključne za ljudsko slušanje.



Slika 3.2.1. Mel spektogram audio zapisa

3.3. Mel-Frequency Cepstral Coefficients (MFCCs)

Mel-Frequency Cepstral Coefficients (MFCCs) su skup karakteristika koje se često koriste u aplikacijama za obradu govora i muzike kako bi se predstavile spektralne karakteristike audio signala. Zasnovane su na ljudskoj percepciji zvuka i dizajnirane su da uhvate spektralni omotač audio signala, koji predstavlja oblik spektra snage signala kroz vreme.

MFCC-ovi su zasnovani na Fourierovoj transformaciji, slično kao i spektrogrami i Mel spektrogrami. Dok spektrogrami daju vizuelnu reprezentaciju frekvencijskog sadržaja signala kroz vreme, MFCC-ovi idu korak dalje i izvlače apstraktne karakteristike signala koje predstavljaju njegov spektralni omotač. Ova informacija je ključna za aplikacije u kojima je potrebna robustna reprezentacija zvuka, jer MFCC-ovi pružaju sažet, ali bogat opis spektralne strukture.[12]

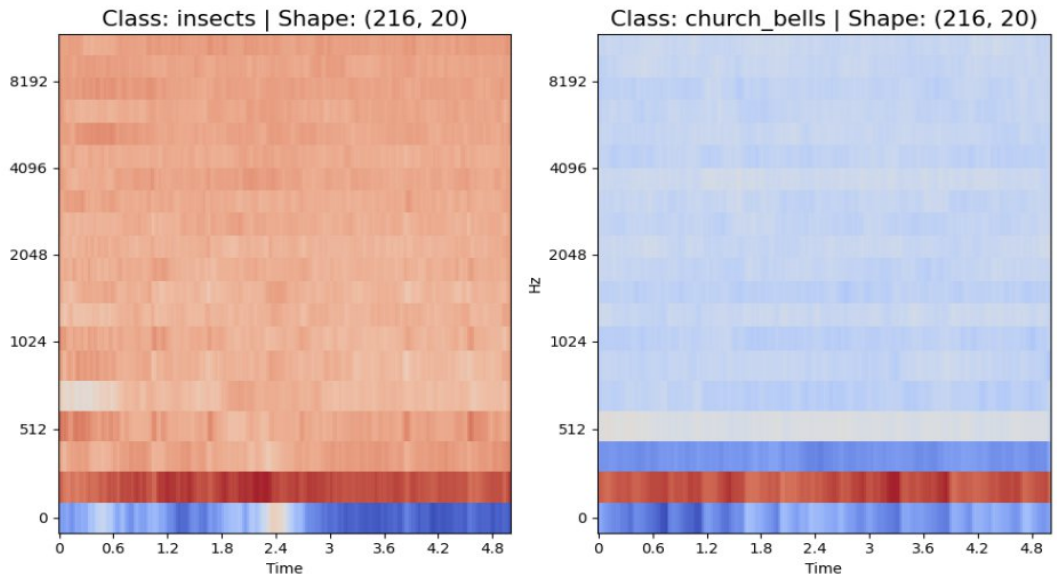
Funkcija *librosa.feature.mfcc* računa Mel-Frequency Cepstral Coefficients (MFCCs), koji predstavljaju karakteristike spektra audio signala. Parematetri su isti kao kod mel spektograma. Rezultat je matrica gde svaka kolona predstavlja jedan trenutak u vremenu, a redovi su MFCC koeficijenti. *librosa.display.specshow* prikazuje matricu MFCC-a kao sliku (grafički prikaz).[13]

```
#get the mfccs
mfcc = librosa.feature.mfcc(y=y,sr=sr,n_fft=2048, hop_length=512, n_mels=128)

#plot the mfcc
librosa.display.specshow(mfcc, sr=sr, ax=ax[i],y_axis='mel', x_axis='time')
```

Koraci za izračunavanje MFCC [13]:

1. Transformacija signala u frekvencijski domen**:** prvi korak u dobijanju MFCC-a je primena Fourierove transformacije na audio signal. Ovo omogućava prelazak iz vremenskog domena u frekvencijski domen, gde se analiziraju različite frekvencije koje čine signal.
2. Primena Mel skale: nakon što je signal prebačen u frekvencijski domen, frekvencije se preslikavaju na Mel skalu, koja oponaša nelinearnu frekvencijsku reakciju ljudskog uha. Na Mel skali, niže frekvencije su više diferencirane od visokih frekvencija, jer ljudsko uho bolje razlikuje niže tonove.
3. Logaritam energije: u svakom Mel-frekvencijskom opsegu računa se energija, a zatim se primenjuje logaritam na energiju. Ova transformacija omogućava bolju analizu, jer ljudsko uho percipira zvuk u logaritamskoj skali.
4. Diskretna kosinusna transformacija (DCT): na kraju, primenjuje se diskretna kosinusna transformacija (DCT) na Mel-frekvencijski spektrogram kako bi se dobili MFCC koeficijenti. DCT omogućava kompresiju informacija, pri čemu prvih nekoliko koeficijenata nosi najvažnije informacije o obliku spektralnog omotača signala.

Kroz ove korake dobija se skup koeficijenata (MFCCs) koji predstavljaju spektralni omotač audio signala. Tipično se koristi između 12 i 40 koeficijenata, u zavisnosti od aplikacije. MFCC-ovi su otporni na promene u audio signalu kao što su promena tona, brzine ili šuma, što ih čini pogodnim za različite zadatke poput prepoznavanja govora, klasifikacije muzičkih žanrova i identifikacije govornika.



Slika 3.3.1. MFCC audio zapisa

3.4. Chromagram

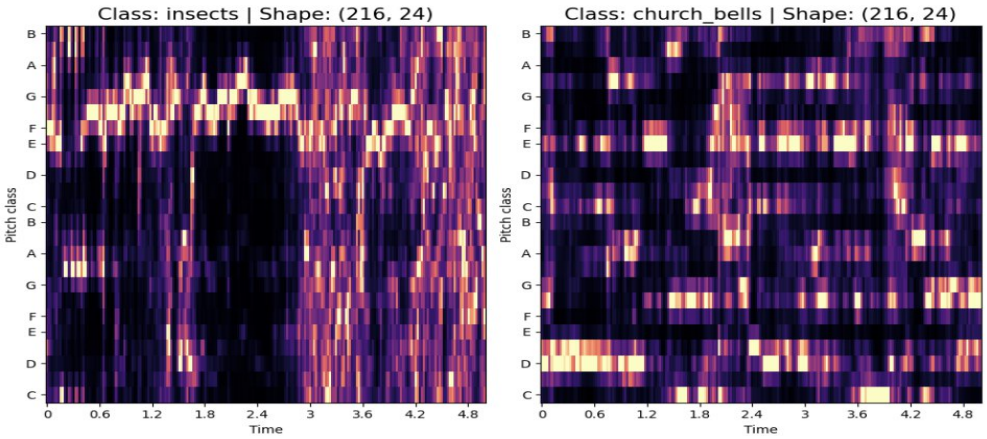
Chromagrami su specifičan tip spektrograma koji se koristi za analizu harmonske strukture audio signala, posebno u muzičkim aplikacijama. Oni omogućavaju prepoznavanje tonalnih klasa i harmonijskih obrazaca, što je korisno za različite zadatke u analizi muzike, kao što su prepoznavanje akorda, klasifikacija muzičkih žanrova i analiza tonaliteta. Glavna ideja chromagrama je da objedini sve spektralne informacije koje se odnose na određenu tonalnu klasu u jedan koeficijent. Oni se dobijaju sumiranjem svih koeficijenata frekvencije koji pripadaju istoj tonalnoj klasi.[12]

Funkcija *librosa.feature.chroma_stft* uzima audio signal i izračunava chromagram, što je 2D prikaz koji pokazuje energetski raspored po tonalnim klasama tokom vremena. Parametar *n-choroma=24* određuje broj tonalnih klasa (semitona) koje se koriste. U ovom slučaju, koristi se 24 binova, što je u skladu sa standardnim brojem semitona u oktavi. Funkcija *librosa.display.specshow* vizualizuje izračunati chromagram. Na x-osi je prikazano vreme, dok je na y-osi prikazan raspored energije po tonalnim klasama. Različite boje ili intenziteti na slici predstavljaju različite nivoe energije u svakoj tonalnoj klasi tokom vremena.[13]

```
#get the chromagram
chromagram = librosa.feature.chroma_stft(y=y,sr=sr,n_fft=2048,hop_length=512, n_chroma=24)

#plot the chromagram
librosa.display.specshow(chromagram, sr=sr, hop_length=512,x_axis='time', y_axis='chroma',ax=ax[i])
```

Koraci za izračunavanje Chromagrama [13]:

1. Transformacija u frekvencijski domen: prvi korak je primena Fourierove transformacije na audio signal, što omogućava analizu frekvencijskog sadržaja.
2. Mapiranje na Chroma prostor: nakon transformacije, signal se mapira u novi prostor karakteristika poznat kao Chroma prostor. Ovaj prostor ima 12 binova koji odgovaraju 12 semitona zapadne muzike (C, C#, D, D#, itd.).
3. Agregacija energetske informacije: svaki bin u Chroma prostoru predstavlja energiju audio signala za određenu tonalnu klasu. Energija svih tonova koji pripadaju istoj tonalnoj klasi se sabira u jedan koeficijent. Na primer, sve frekvencije koje odgovaraju tonu C će biti objedinjene u jedan koeficijent.
4. Rezultantni vektor: rezultat je 12-dimenzionalni vektor gde svaka dimenzija predstavlja energiju signala u određenoj tonalnoj klasi.



Slika 3.4.1. Chromagram audio zapisa

3.5. Tempogram

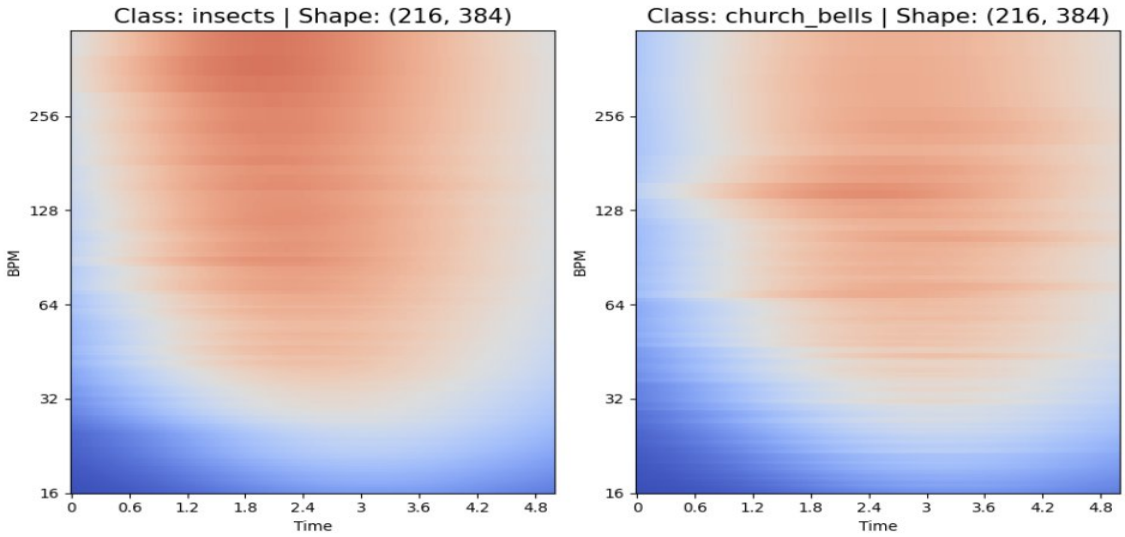
Tempogrami prikazuju kako se tempo, ili brzina ritma, menja tokom vremena u audio zapisu. Ovo može pomoći u analizi ritmičkih obrazaca i tempa u muzici, što je korisno u različitim aplikacijama, muzičkoj analizi, prepoznavanje ritma i detekcija promena u tempu.[12]

Funkciju *onset_strength* iz *librosa* biblioteke da izračuna envelope snage (onset strength envelope) iz audio signala *y*. Ovaj parametar prati promene u intenzitetu zvuka tokom vremena i pomaže u identifikaciji trenutaka kada se zvuk počinje ili menja. *hop_length=512* označava broj uzoraka između svake tačke u analizi. Funkciju *tempogram* iz librosa služi da izračuna tempogram na osnovu envelope snage oenv. Tempogram prikazuje kako se tempo (brzina) zvuka menja tokom vremena. Koristi se envelope snage da se analizira ritmička struktura i prepozna tempo u audio signalu. Parametar *hop_length* postavlja razmak između izračunatih tačaka u tempogramu. Funkcija *librosa.display.specshow* vizualizuje tempogram, vreme se prikazuje na x-osi, dok y-osa predstavlja periode tempa. Intenzitet boje u tempogramu odražava promene u ritmu, omogućavajući vizualnu analizu ritmičkih obrazaca i tempa.[12]

```
#get the tempogram
oenv = librosa.onset.onset_strength(y=y, sr=sr, hop_length=512)
tempogram = librosa.feature.tempogram(onset_envelope=oenv, sr=sr, hop_length= 512, norm=None)

#plot the tempogram
librosa.display.specshow(tempogram, sr=sr, hop_length=512,x_axis='time', y_axis='tempo',ax=ax[i])
```

Postupak kreiranje tempograma:

1. Envelope Snage Početka (Onset Strength Envelope): prvo, iz audio signala se izvlači envelope snage početka, što je mera promene glasnoće ili intenziteta zvuka u vremenskom okviru. Ova envelope pokazuje gde se u audio zapisu nalaze nagli skokovi u intenzitetu, kao što su udarci ili akcenti u ritmu.
2. Automatska Korelacija (Autocorrelation): nakon što je envelope snage početka izračunata, primenjuje se automatska korelacija. Automatska korelacija je metoda kojom se signal upoređuje sa njegovom vlastitom odloženom kopijom da bi se otkrio periodičan obrazac. U kontekstu tempograma, ovo pomaže u prepoznavanju ritmičkih ciklusa i ponavljajućih obrazaca u snazi početka zvuka.



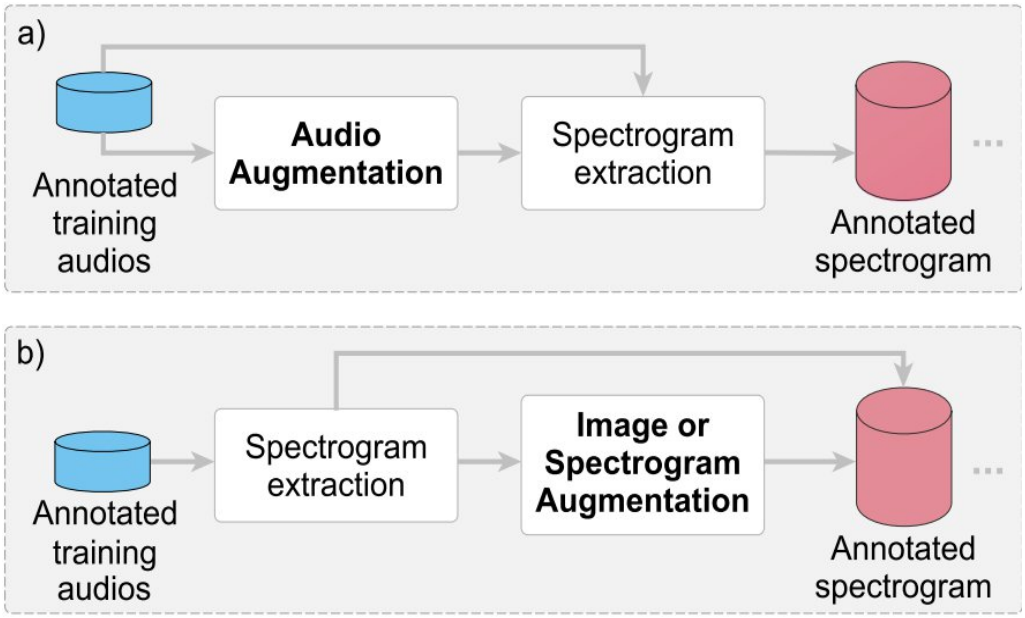
Slike 3.5.1. Tempogram audio zapisa

4. Augmentacija podataka kod audio zapisa

Augmentacija podataka je skup efikasnih metoda koje se koriste za transformaciju postojećih skupova podataka i generisanje sintetičkih podataka kako bi se poboljšala sposobnost generalizacije sistema za prepoznavanje. Iako je strategija prvobitno razvijena za obradu slika, može biti korisna i za audio podatke. Veliki broj metoda augmentacije zvuka razvijen je u oblasti veštačke inteligencije vezane za audio, sa tendencijom daljeg rasta.

U zavisnosti od formata ulaznih audio podataka, bilo da su u obliku talasnog oblika ili spektrograma, metode augmentacije podataka mogu se klasifikovati u dve vrste: Augmentacija audio podataka (Audio DA) i Augmentacija podataka na spektrogramu (Spectrogram DA). Ove metode se mogu primeniti na audio uzorke pre ili posle ekstrakcije spektrograma. U drugom slučaju, spektrogram se može tretirati kao slika, što dovodi do tri glavne grupe tehnika augmentacije audio podataka[11]:

1. Augmentacija audio podataka (Audio DA)
2. Augmentacija podataka na spektrogramu (Spectrogram DA)
3. Augmentacija slika (Image DA)

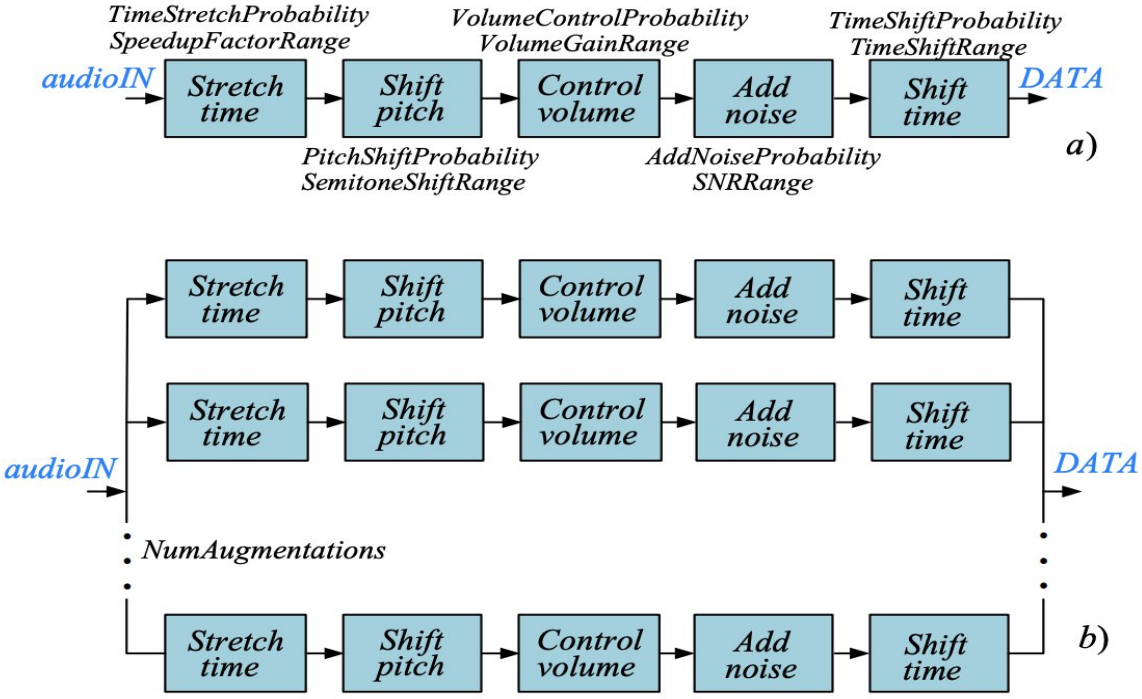


Slika 4.1. Metode augmentacije audio podataka[11]

Augmentacija audio podataka (Audio DA) obuhvata niz modifikacija koje se primenjuju direktno na talasnom obliku. Često koriste sledeće tehnike: *Pitch shifting, Time stretching, Time shifting, Noise, Volume scaling, Polarity inversion, Filters: Low-pass filter,High-pass filter, Band-pass filter, Low-shelf filter, High-shelf filter, Band-stop filter, Peak filter.*[11]

Augmentacija podataka na spektrogramu (Spectrogram DA) omogućava primenu svih tehnika koje se koriste za augmentaciju audio podataka (Audio DA), uz dodatak specifičnih tehnika kao što su *time masking* i *frequency masking*, koje pomažu u povećanju robusnosti modela prema varijacijama u vremenskom i frekvencijskom sadržaju audio signala.

Algoritmi augmentacije audio podataka mogu se primenjivati sekvencijalno (u seriji) ili nezavisno (u paraleli). Arhitektura sekvencijalne augmentacije prikazana je na slici 4.2.a. (prikazuje jedan augmentirani izlaz), dok slika 4.2.b. (prikazuje N augmentiranih izlaznih signala) za paralelne slučajne sekvencijalne augmentacije.[11]



Slika 4.2. Arhitektura sekvencijalne augmentacije audio signala za jedan (a) i N (b) augmentacija ulaznog audio signala[11]

Postoji mnogo komercijalnih i open-source biblioteka za augmentaciju audio podataka. Biblioteka *numpy* pruža jednostavan način za dodavanje šuma i pomeranje vremena. Neke biblioteke su robusnije od drugih, dok se neke fokusiraju na određene oblasti, kao što je ljudski govor. Poznate open-source biblioteke za augmentaciju zvuka su [10]:

1. **Librosa** je open-source Python biblioteka za analizu muzike i zvuka. Postala je dostupna 2015. godine i dugo je bila popularan izbor. Mnoge druge biblioteke za obrada i augmentaciju zvuka koriste funkcije iz Librosa kao gradivne blokove. Može se pronaći na GitHub-u na adresi https://github.com/librosa/librosa.

2. **Audiomentations** je Python biblioteka specifično dizajnirana za augmentaciju audio podataka. Njena ključna prednost je robusnost i jednostavna integracija u projekte. Citira se u mnogim pobednicima Kaggle takmičenja. Može se pronaći na GitHub-u na adresi https://github.com/iver56/audiomentations.

3. **Keras** je Python biblioteka za predobrada audio i muzičkih signala. Implementira konverzije frekvencije i augmentaciju podataka koristeći GPU za obrada. Može se pronaći na GitHub-u na adresi https://github.com/keunwoochoi/kapre.

4.1. Pitch Shifting

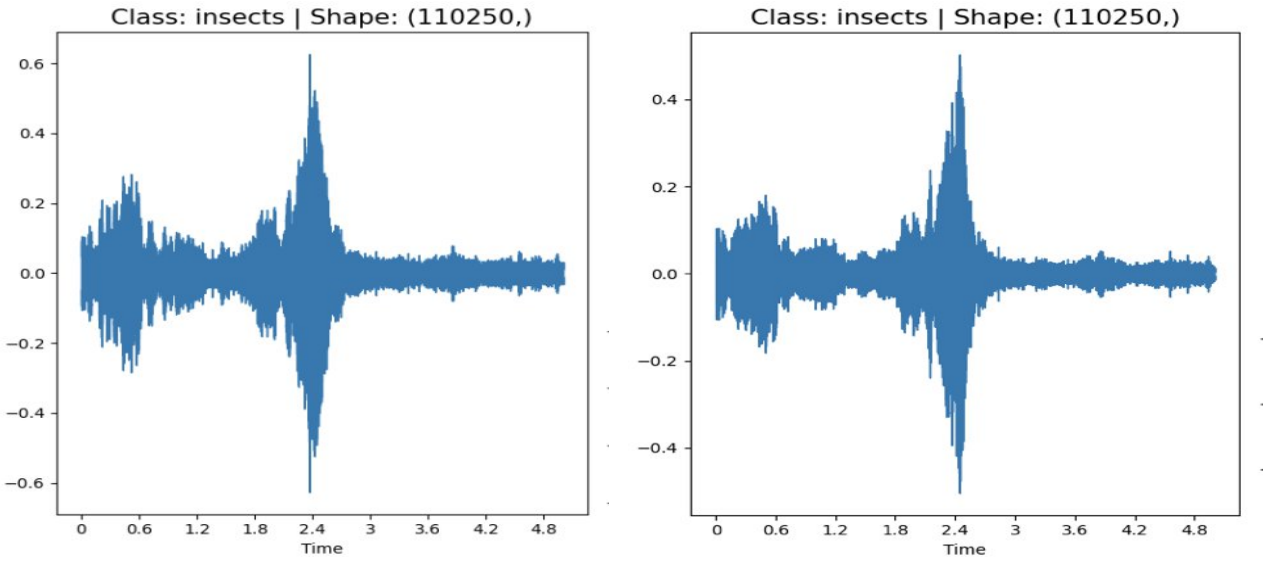
Tehnika pitch shifting je proces modifikacije frekvencijskog sadržaja audio signala bez promene njegovog trajanja. U suštini, pitch shifting podrazumeva pomeranje frekvencije audio signala, što dovodi do toga da zvuk postane "viši" ili "niži" u odnosu na original. Ova tehnika je korisna u situacijama kada je potrebno simulirati varijacije u visini tona audio zapisa, kao što je, na primer, razlika između različitih govornika ili muzičkih instrumenata. U praksi, to se postiže povećanjem ili smanjenjem frekvencija zvučnih talasa, pri čemu se struktura signala održava dovoljno stabilnom da zadrži svoju prepoznatljivost.[1]

Funkcija pitch shift uzima vremensku seriju audio signala, brzinu uzorkovanja (sample rate), i broj koraka ili semitona za promenu visine. Ovaj broj može biti pozitivan za povećanje visine ili negativan za smanjenje. Broj koraka ne mora biti ceo; funkcija omogućava pomeranje visine zvuka za delimične semitone, koristeći decimalne vrednosti. U ovom projektu, broj binova po oktavi (bins per octave) ostaje na podrazumevanoj vrednosti od dvanaest, ali ovaj parametar može biti podešen za različite raspodele oktava.[4]

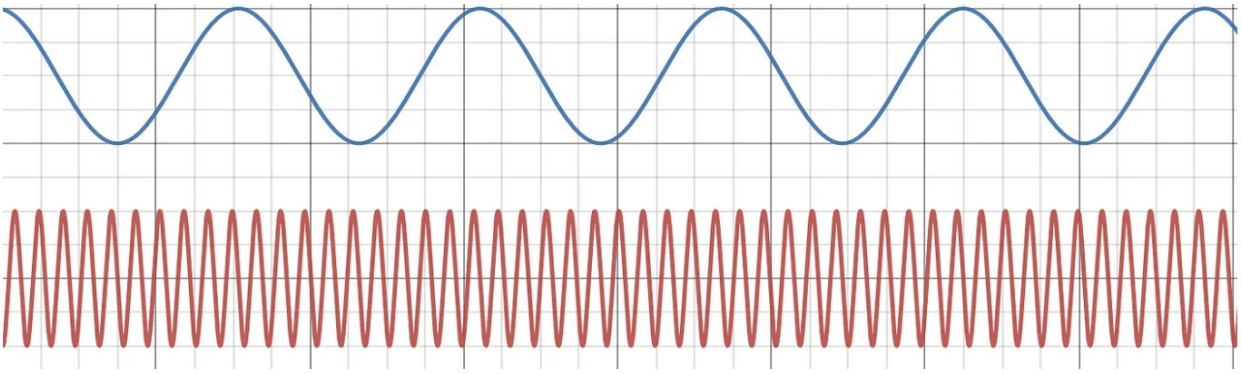
```
def pitch_shift(signal, sr, num_semitones):
    return librosa.effects.pitch_shift(signal, sr=sr, n_steps=num_semitones)
```

Slika 4.1.1. Funkcija pitch shift

Pitch shifting se razlikuje od resampliranja. Dok resampliranje komprimuje ili širi vremensku skalu zvuka - gde povećanje uzorkovanja uzrokuje višu visinu i kraće trajanje, a smanjenje uzorkovanja dovodi do niže visine i dužeg trajanja - pitch shifting samo menja visinu zvuka bez uticaja na dužinu trajanja. Ovo je postignuto pomoću tehnike koja koristi kružni bafer u kojem se uzorci upisuju i čitaju pri različitim brzinama uzorkovanja. Da bi se izbegli klikovi ili prekidi u signalu, koristi se prelaz između dva različita pokazivača za čitanje, čime se obezbeđuje da se volumen izlaza smanjuje na nulu prilikom susreta sa pokazivačem za upisivanje. [2]



Slika 4.1.2. Levo original audio zapis, desno audio zapis pitch shifting



Slika 4.1.3. Pitch shift [1]

4.2. Time Stretching

Prolongiranje vremena (Time Stretching) je tehnika obrade zvuka koja menja trajanje audio signala bez promene njegove visine tonova. Ovo omogućava ubrzavanje ili usporavanje audio signala dok tonalni kvalitet zvuka ostaje nepromenjen. Ova tehnika je korisna za simulaciju različitih ritmova ili tempa u audio sadržaju, kao što su muzika ili govor, bez uticaja na visinu tonova.[4]

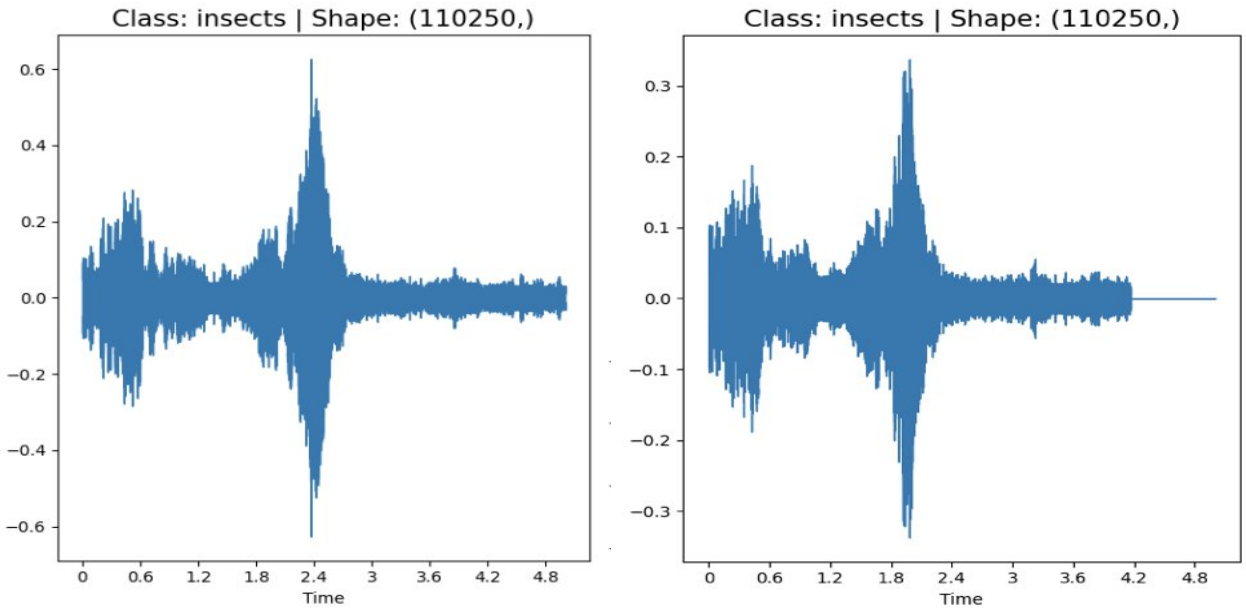
Time Stretching se postiže promenom brzine reprodukcije audio signala. Kada se brzina reprodukcije poveća, trajanje zvuka se skraćuje, dok visina tonova ostaje konstantna. Smanjivanje brzine reprodukcije produžava trajanje zvuka. Ova tehnika često koristi algoritme kao što je fazno vokodiranje, koji pomaže u smanjenju neželjenih artefakata tako što uzima u obzir fazne komponente zvuka, pored frekvencija.[7]

Funkcija stretch_sound koristi biblioteku librosa za proširenje ili skraćivanje trajanja audio signala dok visina tonova ostaje nepromenjena. Najpre, funkcija beleži početnu dužinu audio signala, a zatim primenjuje tehniku proširenja vremena koja menja trajanje signala prema zadatom faktoru brzine. Ako rezultantni signal postane duži od originala, funkcija ga skraćuje na početnu dužinu. Ako je rezultantni signal kraći, funkcija dodaje nule na kraju kako bi dužina rezultantnog signala odgovarala početnoj dužini. Ovo omogućava prilagođavanje brzine zvuka dok se dužina signala održava, što može biti korisno za različite aplikacije obrade zvuka.[4]

```
def stretch_sound(x, rate):
    input_length = len(x)
    x = librosa.effects.time_stretch(x, rate=rate)
    if len(x)>input_length:
        return x[:input_length]
    else:
        return np.pad(x, (0, max(0, input_length - len(x))), "constant")
```

Slika 4.2.1. Funkcija stretch_sound koja koristi librosa funkciju time_stretch[13]

Iako je tehnika istezanja vremena korisna, nije uvek efikasna za sve vrste zvukova. U nekim slučajevima može izazvati neprirodne distorzije, posebno kod instrumenata ili glasova sa izraženim vibrato efektima, poput violine ili ženskog vokala. Istraživanja su pokazala da kod ovih klasa zvuka, istezanje vremena može negativno uticati na prepoznavanje zvuka zbog promene vibracijskih karakteristika.



Slike 4.2.2. Levo orginal audio zapis, desno audio zapis time stretching

4.3. Time Shifting

Time shifting je tehnika obrade zvuka koja se koristi za pomeranje audio signala duž vremenske ose, bilo ulevo ili udesno, bez menjanja visine tona. Ova tehnika je korisna za simulaciju varijacija u vremenskom rasporedu zvuka i može pomoći u obogaćivanju skupa podataka za obuku modela mašinskog učenja.[3]

Kada se audio signal pomeri ulevo, dolazi do napredovanja zvuka, što znači da se zvuk pojavljuje ranije nego što je izvorno bio. Ovo može simulirati situacije kada se zvuk reprodukuje brže. Nasuprot tome, pomeranje udesno uzrokuje kašnjenje, tako da zvuk kasni u odnosu na originalno vreme. Ovo može imitirati uslove u kojima se zvuk reprodukuje sporije.

Matematički, time shifting se može prikazati kao promena vremenske varijable u audio signalu. Ako je Δt vreme pomeranja, nova pozicija zvučnog signala se dobija dodavanjem ili oduzimanjem Δt od originalnog vremena. Na primer, ako je Δt pozitivno, signal se pomera ulevo (napred), dok ako je Δt negativno, signal se pomera udesno (kašnjenje).[9]

$$y(t) = x(t + \Delta t)$$

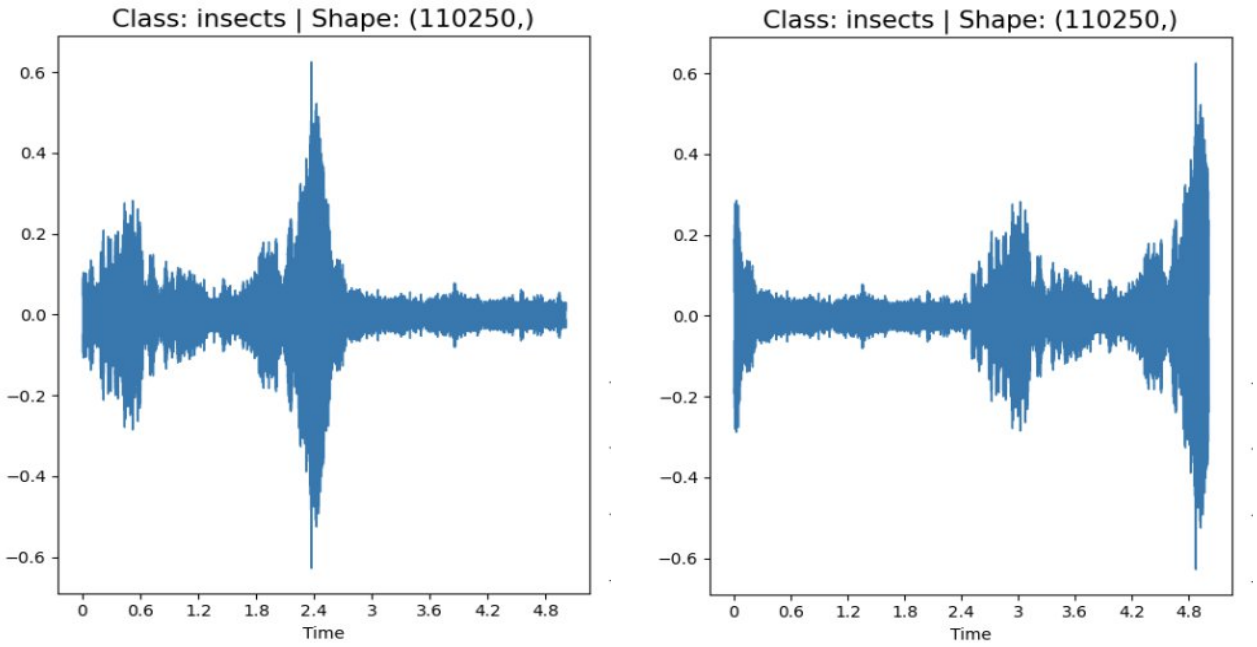
$$y(t) = x(t - \Delta t)$$

U praktičnom smislu, time shifting može se primeniti na različite načine u audio obradi. Na primer, može se koristiti za simulaciju različitih uslova snimanja ili za kreiranje varijacija u audio uzorcima za poboljšanje performansi modela u zadacima kao što su prepoznavanje govora ili klasifikacija zvukova.

Funkcija shift_sound (Slika 4.3.1) je dizajnirana da pomeri audio signal u vremenu. Prima audio talas x i parametar rate, koji određuje koliko će signal biti pomeren. Funkcija koristi NumPy-ovu metodu np.roll za ostvarivanje ovog pomeranja. Konkretno, izračunava broj uzoraka za pomeranje tako što deli dužinu audio signala sa rate. Funkcija np.roll zatim pomera celu talasnu formu za ovaj broj uzoraka. Kao rezultat, funkcija efektivno pomera audio signal unapred ili unazad u vremenu. Važno je napomenuti da np.roll operacija omotava kraj signala oko početka, što znači da deo signala koji je pomeren van kraja ponovo se pojavljuje na početku.[4]

```
def shift_sound(x, rate=2):
    return np.roll(x, int(len(y)//rate))
```

Slika 4.3.1. Funkcija shift_sound[13]



Slika 4.3.2. Levo original audio zapis, desno audio zapis time shifting

4.4. Noise

Dodavanje šuma Noise predstavlja ključnu tehniku u augmentaciji audio podataka koja služi za unapređenje robusnosti modela u prepoznavanju i analizi zvuka. Ova metoda uključuje dodavanje različitih vrsta šuma originalnim audio zapisima kako bi se obučeni modeli prilagodili varijacijama koje se mogu javiti u stvarnim uslovima snimanja. Povećanje otpornosti na buku iz stvarnog sveta pomaže u unapređenju tačnosti i generalizacije modela, što je od suštinskog značaja za aplikacije u realnim okruženjima.

Pozadinski šum (Background Noise) predstavlja jednu od najčešće korišćenih vrsta šuma u augmentaciji audio podataka. Ovaj šum uključuje kombinovanje originalnog zvuka sa svakodnevnim zvukovima iz okoline, kao što su zvuci saobraćaja, razgovori ljudi, zvukovi iz parkova ili rad sa uličnim radnicima. Cilj ove metode je da modelima omogući da se bolje nose sa realnim okruženjima u kojima zvučni zapisi često sadrže različite pozadinske smetnje. Dodavanje pozadinskog šuma omogućava modelima da prepoznaju ključne karakteristike zvuka i u prisustvu dodatnih smetnji, čime se poboljšava njihova sposobnost da funkcionišu u različitim stvarnim scenarijima.[5]

Gausovski šum (Gaussian Noise) se generiše prema normalnoj (Gausovoj) distribuciji, gde su vrednosti šuma raspoređene oko srednje vrednosti (obično nula) sa određenom varijansom. Ovaj tip šuma dodaje varijacije u zvučni signal koje su distribuirane prema normalnoj raspodeli. Gausovski šum je koristan za simulaciju realnih scenarija u kojima buka može biti raspoređena prema određenoj statističkoj distribuciji. Na primer, ovaj šum može pomoći u obuci modela da se bolje nosi sa različitim vrstama smetnji koje su prisutne u stvarnom svetu, kao što su varijacije u pozadini koje nisu uniformno raspodeljene.[9]

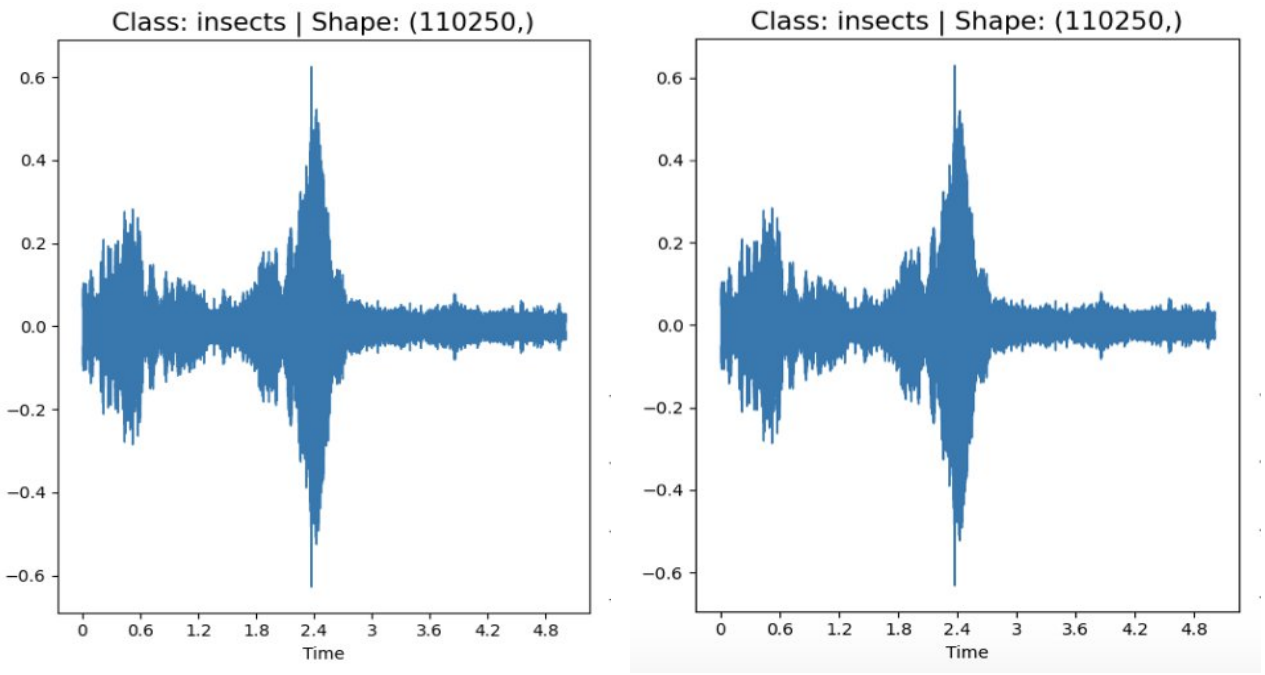
```
noise = Compose([
  AddGaussianNoise(min_amplitude=0.005, max_amplitude=0.15, p=1),
])

bg_noise = Compose([
    AddBackgroundNoise(
        bg_noise_dir,
        min_snr_db=5,
        max_snr_db=30,
        p=1
    )
])
```

Slika 4.4.1. Pozadinski i Gausov šum

Sintetički šum (Synthetic Noise) je šum koji se generiše umetnički, često u obliku bele buke. Bela buka je vrsta slučajnog šuma koji ima ravnomernu raspodelu energije preko svih frekvencija, što znači da sve frekvencije doprinose ukupnom šumu podjednako. Ova vrsta šuma se koristi kada je potrebno umetnuti kontrolisane varijacije u zvučni signal kako bi se stvorile specifične karakteristike šuma. Sintetički šum može se koristiti za testiranje modela u kontrolisanim uslovima, gde se precizno definišu parametri šuma, omogućavajući sistematsko proučavanje efekata šuma na performanse modela.[5]

```
# data augmentation: add white noise
def add_white_noise(x, rate=0.002):
    return x + rate*np.random.randn(len(x))
```

Slika 4.4.2. Beli šum



Slika 4.4.3. Levo original audio zapis, desno audio zapis sa šumom

Slučajni šum (Random Noise) odnosi se na bilo koju vrstu šuma koja se dodaje originalnom audio signalu generisanjem slučajnih vrednosti. Karakteristike slučajnog šuma mogu varirati u zavisnosti od metode generisanja i uključuju promene u frekvencijama i amplitudama. Ova vrsta šuma može uključivati različite oblike smetnji koje nisu prisutne u drugim vrstama šuma, pružajući dodatne varijacije u signalu koje mogu pomoći u poboljšanju performansi modela u realnim uslovima.[3]

Sve ove vrste šuma ne menjaju ključne karakteristike zvuka, kao što su visina tona (pitch) ili tempo, ali uvode varijacije u kvalitetu signala. Ove varijacije pomažu u stvaranju robusnijih modela koji mogu bolje da se nose sa stvarnim uslovima. Dodavanje šuma pomaže u obuci modela da prepoznaje važne informacije iz zvučnih zapisa čak i kada su prisutni dodatni smetnji, čime se poboljšava njegova sposobnost da funkcioniše u stvarnom svetu. Implementacija ove tehnike obično uključuje dodavanje nasumičnih vrednosti svakom uzorku signala, a količina šuma se može kontrolisati koeficijentom koji definiše intenzitet šuma. Na taj način, model se obučava da bude otporniji na nepredviđene smetnje i poboljšava svoju sposobnost da se prilagodi različitim scenarijima u stvarnom svetu.

4.5. Volume Scaling

Volume Scaling je jedna od osnovnih tehnika za augmentaciju podataka u domenu obrade audio signala, a njen cilj je prilagođavanje glasnoće audio zapisa. Ova tehnika menja amplitudu signala, čime se postiže efekat pojačavanja ili smanjenja jačine zvuka. Korišćenjem ove metode možemo simulirati razne situacije u kojima bi zvuk mogao biti snimljen na različitim nivoima glasnoće, kao što su udaljenost od izvora zvuka ili promene u jačini okruženja.

Tehnika se zasniva na skaliranju amplitude audio signala. Kada se glasnoća poveća, svaka tačka signala se množi sa vrednošću većom od 1. Kada se glasnoća smanji, koristi se vrednost manja od 1. Ovim postupkom se kreiraju različiti nivoi zvuka, što omogućava modelu da postane otporniji na varijacije u jačini. Na primer, glasnoća se može skalirati za -10 ili +10 dB (decibela), čime se zvuk utišava ili pojačava. Ova skala obuhvata širok opseg glasnoće, što omogućava bolju obuku modela u prepoznavanju signala snimljenih u različitim uslovima.

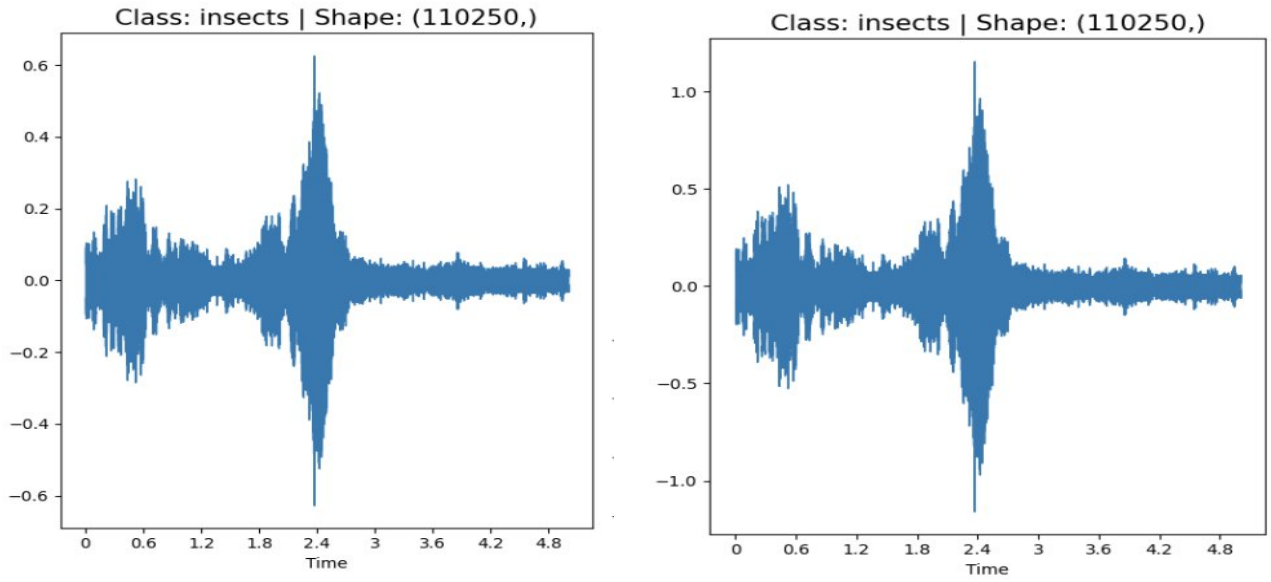
Varijante Volume Scaling-a:

1. Loudness Adjustment (Podešavanje glasnoće): Ova tehnika podrazumeva promenu jačine svih uzoraka signala za određenu količinu decibela. Na primer, u studiji Aguiar et al. (2018), glasnoća je povećavana i smanjivana za fiksne vrednosti, kao što su -10 i +10 dB. Ovo omogućava modelu da prepozna signale čak i kada su pretiho ili preglasno snimljeni.[5]
2. Dynamic Range Compression (DRC): Kompresija dinamičkog opsega koristi se za promenu glasnoće uzorka dodavanjem šuma ili izobličenja. Ova tehnika je korisna kada se želi model obučiti da prepoznaje zvukove u složenim scenarijima. Na primer, Salamon i Bello (2017) koristili su kompresiju kako bi model prilagodili različitim standardima za muziku, filmove, govor i radio.[5]

Funkcija *volume_scaling* nasumično povećava glasnoću audio signala za faktor između 1.5 i 2.5 puta, simulirajući različite nivoe glasnoće u audio podacima, što je korisno za augmentaciju podataka pri obučavanju modela.[13]

```
def volume_scaling(data):
    sr  = 16000
    dyn_change = np.random.uniform(low=1.5,high=2.5)
    data = data * dyn_change
    return data
```

Slika 4.5.1. Funkacija volume_scaling[13]



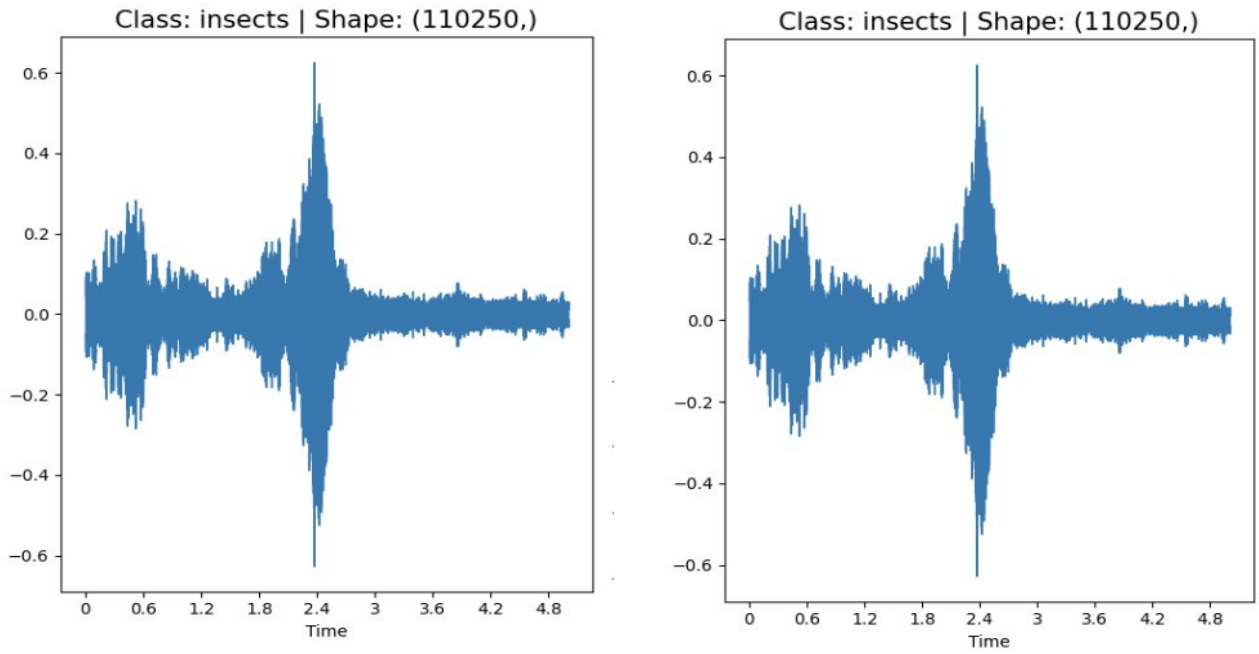
Slika 4.5.2. Levo original audio zapis, desno audio zapis volume scaling

4.6. Polarity inversion

Polarity inversion (ili obrnuta polaritet) je tehnika u audio obradi koja menja fazu audio signala za 180 stepeni. To znači da se svaki deo signala okrene za pola perioda, što može imati različite efekte na zvuk zavisno od konteksta.

Audio signal se obično prikazuje sa pozitivnim i negativnim delovima talasnog oblika. Polarity inversion menja sve pozitivne delove u negativne, i obrnuto. Matematički, ovo se postiže množenjem amplitude sa negativnom vrednošću.

Za većinu ljudi, zvuk posle obrtanja polariteta zvuči isto kao originalni audio. Ova tehnika je najkorisnija za mašinsko učenje kada se koristi u modelima koji su svesni faze. Ne postoji sigurna oblast jer se ili koristi ili ne koristi. Polarity inversion se može koristiti za rešavanje problema sa faznim poremećajem u snimanju ili miksovanju, kada su dva mikrofona snimila isti izvor zvuka sa različitim fazama. Ako se koristi na celom snimku, obrnuta polaritet može značajno promeniti zvuk, često uzrokujući da neki delovi zvuka nestanu ili postanu veoma tihi zbog destruktivne interferencije.[10]



Slika 4.6. Levo original audio zapis, desno audio zapis polarity inversion

4.7. Filteri

Audio filteri su alati koji se koriste za uklanjanje neželjenih šumova ili smetnji iz audio snimka, čime se poboljšava kvalitet zvuka i omogućava čistija reprodukcija govora, muzike, zvukova prirode ili okruženja. Filteri funkcionišu tako što prilagođavaju određene frekvencije u zvuku – mogu pojačati, povećati ili smanjiti jačinu određenog frekvencijskog opsega. Na primer, koristeći niskopropusni filter, možete ukloniti zvuke visokih frekvencija, kao što je buka saobraćaja, i tako omogućiti da se razgovor dvoje ljudi snimljen u gradskom okruženju čuje jasnije i bez smetnji.[10]

Low-pass filter

Niskopropusni filter uklanja ili smanjuje zvuke niskih frekvencija, poput buke saobraćaja, brujanja motora ili poziva slonova. Tipično, minimalna frekvencija sečenja je 150 Hz, maksimalna 7,5 kHz, minimalno prigušenje je 12 dB, dok je maksimalno prigušenje 24 dB. Zanimljivost: pozivi slonova imaju frekvencije niže od 20 Hz, što spada u oblast infrazvuka. [10]

High-pass filter

Visokopropusni filter, slično niskopropusnom, uklanja zvuke visokih frekvencija, poput zviždanja, plača beba, grebanja noktima ili zvona. Tipično, minimalne i maksimalne frekvencije sečenja su 20 Hz i 2,4 kHz, dok su minimalna i maksimalna prigušenja 12 dB i 24 dB. Zanimljivost: čovek može zviždati na frekvenciji od oko 3 do 4 kHz.[10]

Band-pass filter

Propusni opseg filter ograničava zvučni talas na određeni raspon frekvencija. Drugim rečima, on kombinuje niskopropusne i visokopropusne filtere. Na primer, propusni opseg filter može učiniti razgovor dvoje prijatelja u prometnom restoranu na otvorenom u Parizu jasnijim za slušanje. Slično tome, može se koristiti za izolaciju ptičijeg peva u bučnoj amazonskoj džungli. Minimalne i maksimalne središnje frekvencije su 200 Hz i 4 kHz, minimalni i maksimalni delovi širine opsega su 0.5 i 1.99, a minimalna i maksimalna prigušenja su 12 dB i 24 dB.[10]

Low-shelf filter

Niskopojasni filter (low-shelf filter), poznat i kao polica ekvilajzer, pojačava ili smanjuje frekvencije na donjem kraju spektra. Na primer, možete koristiti ovaj filter da smanjite bas u pesmi heavy metal žanra. Obično, minimalne i maksimalne središnje frekvencije su 50 Hz i 4 kHz, a minimalno i maksimalno pojačanje iznosi od -18 dB do 18 dB.[10]

High-shelf filter

Visokopojasni filter (high-shelf filter) takođe pojačava ili smanjuje amplitudu frekvencija na višem kraju spektra. Na primer, možete koristiti ovaj filter da osvetlite (povećate jasnoću) muzičkog zapisa. Uobičajeno, minimalne i maksimalne središnje frekvencije su 300 Hz i 7,5 kHz, dok minimalno i maksimalno pojačanje iznosi od -18 dB do 18 dB.[10]

Band-stop filter

Band-stop filter, takođe poznat kao notch filter ili ban-reject filter, uklanja frekvencije između dva tačno određena preseka ili s obe strane opsega. U osnovi, koristi niskopojasni i visokopojasni filter kao osnovu. Na primer, band-stop filter može eliminisati neželjene pikove i šumove iz muzike na vašem backyard sesiji. Obično, minimalne i maksimalne središnje frekvencije su 200 Hz i 4 kHz, dok minimalne i maksimalne širine opsega su od 0.5 do 1.99, a minimalni i maksimalni roll-off iznosi od 12 dB do 24 dB.[10]

Peak filter

Peak filter, takođe poznat kao bell filter, je suprotan band-stop filteru. Drugim rečima, pojačava frekvencije u uskom opsegu sa višim pojačanjem ili omogućava pojačavanje ili smanjenje oko određene centralne frekvencije. Obično, minimalne i maksimalne središnje frekvencije su 50 Hz i 7.5 kHz, dok minimalne i maksimalne vrednosti pojačanja variraju od -24 dB do 24 dB.[10]

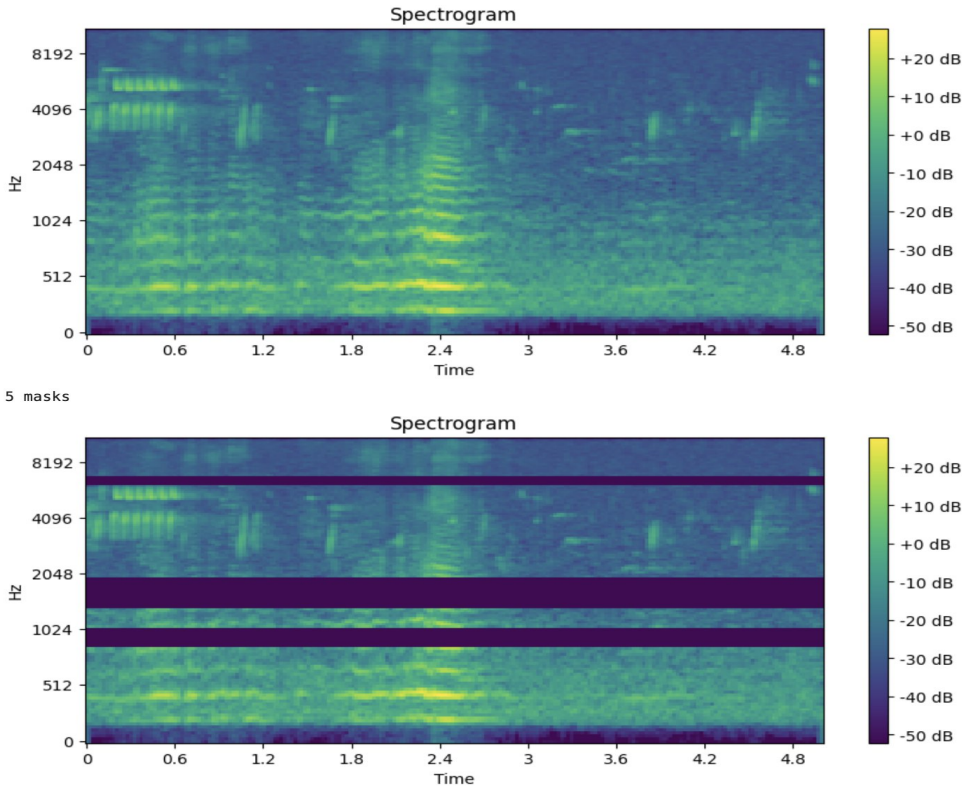
4.8. Frequency Masking i Time Masking

Frequency Masking se primenjuje na frekvencijskom spektru audio signala. U ovoj tehnici, određeni opseg frekvencija [f0, f0 + f) se maskira, gde je f opseg frekvencija izabran iz uniformne raspodele između 0 i parametra frekvencijske maske F, a f0 je izabran iz intervala (0, v − f), gde v predstavlja broj frekvencijskih kanala.

```
def freq_mask(spec, num_masks=1):
    import random
    spec = spec.copy()
    num_mels = spec.shape[0]
    for _ in range(num_masks):
        f = random.randint(0, num_mels // 4)
        f0 = random.randint(0, num_mels - f)
        f1 = f0 + f

        spec[f0:f1, :] = spec.min()
    return spec
```

Slika 4.8.1. Funkcija frequency masking

Maskiranje može biti potpuno ili parcijalno, i može se koristiti nulta ili minimalna vrednost signala za maskirane frekvencije. Ovo može simulirati odsustvo određenih frekvencija i pomoći modelu da nauči da prepoznaje zvukove i u uslovima gde su neki delovi frekvencijskog spektra izgubljeni ili oštećeni.[8]



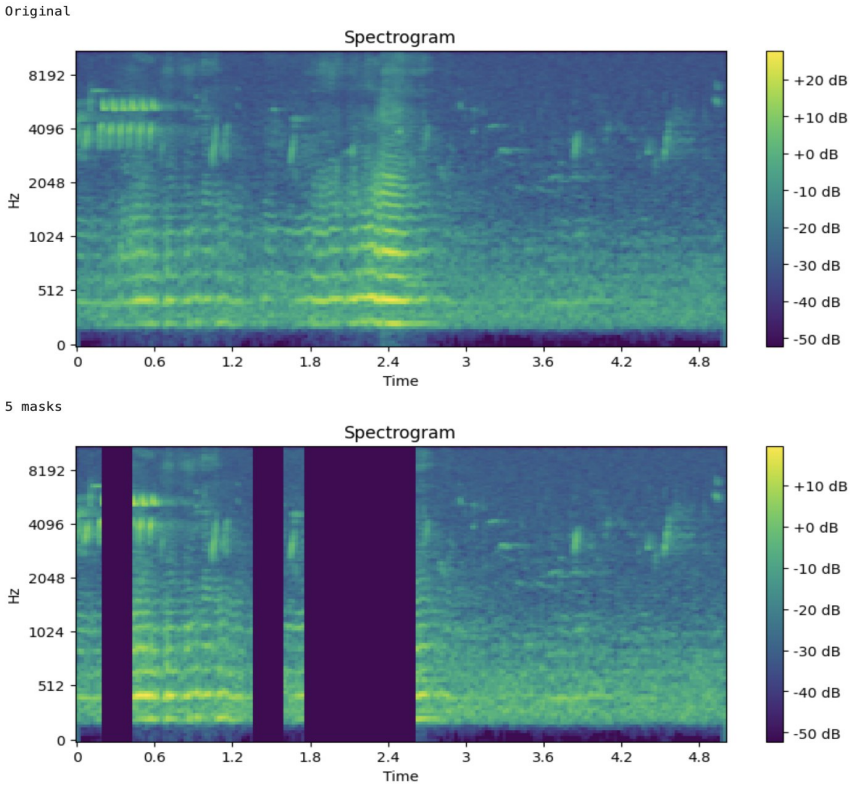
Slika 4.8.2. Frequency masking audio zapisa

Time Masking se odnosi na maskiranje određenih vremenskih segmenata audio signala. U ovoj tehnici, t uzastopnih vremenskih koraka [t0, t0 + t) se maskira, gde je t izabran iz uniformne raspodele između 0 i parametra vremenske maske T, a t0 je izabran iz intervala[0, τ − t), gde τ predstavlja ukupan broj vremenskih koraka u signalu.

```
def time_mask(spec, num_masks=1):
    import random

    spec = spec.copy()
    num_frames = spec.shape[1]
    for _ in range(num_masks):
        t = random.randint(0, num_frames // 4)
        t0 = random.randint(0, num_frames - t)
        t1 = t0 + t

        spec[:, t0:t1] = spec.min()
    return spec
```

Slika 4.8.3. Funkcija time masking

Maskiranje u vremenskom domenu može se postići primenom binarne vremenske maske, koja može zameniti određene delove signala tišinom ili nultom vrednošću. Ova tehnika simulira privremeno odsustvo delova signala i pomaže modelu da se fokusira na druge delove signala.[8][9]



Slika 4.8.4. Time masking audio zapisa

Kombinovanjem Time Masking i Frequency Masking tehnika, moguće je simulirati različite oblike oštećenja i nedostataka u audio signalu, što omogućava modelima da postanu robusniji i precizniji u prepoznavanju zvukova u stvarnom svetu.

5. Zaključak

Augmentacija podataka za audio zapise predstavlja ključan alat u unapređenju performansi modela mašinskog učenja, naročito u kontekstu obrade zvuka i prepoznavanja obrazaca. Ova tehnika omogućava kreiranje raznovrsnih i robusnijih modela tako što proširuje i obogaćuje postojeće skupove podataka, čime se poboljšava generalizacija i otpornost modela na različite varijacije u ulaznim podacima.

U okviru rada, razmatrane su različite metode augmentacije audio podataka, svaka sa svojim prednostima i izazovima. Ove tehnike su pokazale da mogu značajno doprineti u prevazilaženju problema sa malim skupovima podataka i prekomernom pristrasnošću, kao i da poboljšaju sposobnost modela da prepozna i klasifikuje zvuke u različitim uslovima.

Iako su tehnike poput promena tempa, promena tona, istezanja vremena i dodavanja šuma veoma korisne, važno je napomenuti da izbor odgovarajuće tehnike zavisi od specifičnih zahteva zadatka i karakteristika ulaznih podataka. Svaka metoda ima svoje prednosti, ali i ograničenja koja je potrebno uzeti u obzir pri njenom primenjivanju.

Preporuke za dalji rad uključuju ispitivanje kombinacija tehnika augmentacije kako bi se postigla optimalna poboljšanja u performansama modela, kao i istraživanje novih pristupa u augmentaciji koji mogu dodatno unaprediti rezultate. Takođe, važno je uzeti u obzir i efekte augmentacije na kvalitet podataka i moguće neželjene posledice, kao što su gubitak važnih informacija ili uvođenje veštačkih obrazaca koji mogu uticati na tačnost modela.

U zaključku, data augmentation za audio ostaje dinamično polje sa velikim potencijalom za unapređenje tehnologija obrade zvuka i prepoznavanja obrazaca. Korišćenje ovih tehnika može značajno doprineti razvoju robusnijih i preciznijih modela, što je ključno za napredak u različitim aplikacijama audio analize i prepoznavanja.

6. Literatura

1. https://medium.com/strategio/using-librosa-to-change-the-pitch-of-an-audio-file-49efdb2dd6c
2. https://dsp-book.narod.ru/Pitch_shifting.pdf
3. https://blog.pangeanic.com/audio-data-augmentation-techniques-and-methods
4. https://medium.com/@makcedward/data-augmentation-for-audio-76912b01fdf6
5. https://www.sba.org.br/cba2022/wp-content/uploads/artigos_cba2022/paper_5085.pdf
6. https://www.ccslearningacademy.com/what-is-data-augmentation/
7. https://trepo.tuni.fi/bitstream/handle/10024/117251/EklundVille-Veikko.pdf?sequence=2&isAllowed=y
8. https://towardsdatascience.com/data-augmentation-for-speech-recognition-e7c607482e78
9. https://medium.com/@bikashojha904/from-sound-waves-to-spectrograms-a-comprehensive-guide-to-preparing-audio-datasets-for-asr-1-fa324452f523
10. https://annas-archive.org/md5/e14d038439a13c13f638337dbbec8fe0
11. https://www.researchgate.net/publication/373636822_Exploring_the_Impact_of_Data_Augmentation_Techniques_on_Automatic_Speech_Recognition_System_Development_A_Comparative_Study
12. https://medium.com/@notabelardoriojas/environmental-sound-classification-investigating-different-spectrograms-and-audio-augmentation-95f6989d0ae5
13. https://www.kaggle.com/code/salimhammadi07/esc-50-environmental-sound-classification