Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Borut Budna

Platforma za klasifikacijo zvočnih posnetkov

diplomsko delo

visokošolski strokovni študijski program
prve stopnje
računalništvo in informatika

Mentor: doc. dr. Tomaž Curk
Somentor: prof. dr. Matjaž Gams

Ljubljana, 2017

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

V okviru odprtokodnega sistema za podatkovno rudarjenje Orange razvite dodatek za obdelavo zvočnih posnetkov. Dodatek naj vsebuje gradnike za branje, obdelavo in prikaz zvočnih podatkov. Predlagajte in implementirajte izračun značilk, s katerimi je možno zvok opisati na atributni način. Opišite nekaj konkretnih primerov uporabe in poročajte o uspešnosti napovednih modelov.

Zahvaljujem se mentorju doc. dr. Tomažu Curku in somentorju prof. dr. Matjažu Gamsu za koristne napotke in strokovno vodenje pri izdelavi diplomskega dela.

Zahvaljujem se Luki Stepančiču in vsem ostalim sodelavcem z Instituta Jožefa Štefana za pomoč.

Zahvaljujem se svoji družini in prijateljem, predvsem pa punci Nini za podporo in motivacijo.

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
AUC	area under curve	površina pod krivuljo
F1	weighted harmonic mean of precision and recall	uteženo harmonično povprečje preciznosti in priklica
FFT	fast Fourier transform	hitra Fourierova transformacija
STFT	short-time Fourier transform	kratko-časovna Fourierova transformacija
MIR	music information retrieval	pridobivanje informacij iz glasbe
MFCC	mel frequency cepstral coefficients	mel-frekvenčni kepstralni koeficienti
PLP	perceptual linear prediction	koeficienti percepcijskega linearnega napovedovanja

Povzetek

Naslov: Platforma za klasifikacijo zvočnih posnetkov

Avtor: Borut Budna

V diplomskem delu opišemo sistem za obdelavo in klasifikacijo zvočnih posnetkov, ki smo ga zasnovali in razvili kot dodatek za program Orange. Orange je odprtokodni sistem, namenjen podatkovnemu rudarjenju. Vsebuje gradnike za prikazovanje in manipuliranje s podatki, gradnjo napovednih modelov, in gradnike za mnoga druga opravila. Ne vsebuje gradnikov za branje in obdelovanje zvočnih zapisov. Zato smo v ta namen izdelali dodatek, imenovan "Audio - IJS," znotraj katerega smo razvili osem gradnikov, ki omogočajo manipulacijo z zvokom in integracijo z ostalimi gradniki v Orange. Izdelani gradniki omogočajo filtriranje, segmentacijo (razrez), spektralno analizo, ekstrakcijo značilk, predvajanje zvoka in še nekaj ostalih funkcij, ki so potrebne za povezovanje s preostalimi gradniki Orange. Razviti dodatek uporabniku omogoča hitro izdelavo sofisticiranih analiz zvočnih posnetkov.

Ključne besede: Orange, podatkovno rudarjenje, filtriranje, segmentacija, ekstrakcija značilk, spektrogram, klasifikacija.

Abstract

Title: Platform for audio clips classification

Author: Borut Budna

In this diploma thesis we will describe audio clip analysis and classification system that we designed and developed as an add-on for Orange software. Orange is an open-source system, designed for data mining. It contains widgets for displaying and manipulating data, building predictive models and several other tasks. It does not, however, contain widgets for reading and processing audio clips. For this purpose, we built an add-on called »Audio-IJS« within which we developed eight widgets that enable audio manipulation as well as integration with other Orange widgets. The constructed widgets enable filtration, segmentation (cutting), spectral analysis, feature extraction, audio play as well as several other functions, necessary for connecting with other Orange widgets. The developed add-on enables the user a rapid creation of sophisticated analyses of audio clips.

Keywords: Orange, data mining, filtering, segmentation, feature extraction, spectrogram, classification.

Poglavje 1

Uvod

Metode umetne inteligence in podatkovnega rudarjenja so do nedavnega zahtevale vhodne podatke v strogo pred-definirani obliki, navadno v atributnem zapisu. Nedavni razvoj novih metod strojnega učenja se usmerja na obdelavo vedno obsežnejših zbirk realnih signalov, kot sta zvok in video. Orodje, ki omogoča hitro gradnjo napovednih modelov za analizo zvoka, lahko bistveno zmanjša čas izvajanja eksperimentov tako za strokovnjake kot za neizkušene uporabnike. S takšnim orodjem lahko zlahka zgradimo osnovne napovedne modele za kakršnokoli nalogo, povezano z zvokom. Osnovne modele je mogoče pozneje uporabiti za primerjavo z naprednejšimi pristopi metod podatkovnega rudarjenja za zvok.

V študijah zoologije in biotske raznovrstnosti je pomembna naloga razpoznava posamezne živali iz zvoka. V psihologiji in medicini je pomembna naloga razpoznava človeških čustev, predvsem čustvenih motenj, kot sta depresija in bipolarna motnja. V medicini je pomembna naloga odkriti kronično srčno popuščanje, za katerim trenutno boleha več kot 26 miljonov ljudi po svetu. Pogosto se strokovnjaki na teh področjih ukvarjajo izključno z zvokom, ki v mnogih primerih niso na voljo. Poleg tega se v študijah, ki obravnavajo zvočni monitoring oziroma analizo zvoka, srečujejo z velikimi količinami podatkov. To zahteva uvedbo metod samodejnega uvrščanja v razrede oziroma klasifikacije, ki bodo pomagale strokovnjakom pri njihovem delu.

1.1 Pregled obstoječih rešitev

Po pregledu spletnih virov smo ugotovili, da že obstaja vrsta orodij za zvočno predprocesiranje oziroma klasifikacijo. Razdelimo jih v dve skupini:

- orodja v obliki knjižnic oziroma modulov,
- orodja z grafičnim uporabniškim vmesnikom.

Primeri orodij so Yaffe [12], Essentia [3], aubio [5], librosa [13], openSMILE [8], pyAudioAnalysis [9], Audacity [17] in druge. Med seboj se dodatno razlikujejo predvsem po namembnosti in naboru implementiranih funkcij. Večina naštetih orodij omogoča zgolj eno ali dve funkcionalnosti, naštetih v Tabeli 1.1. Tabela prikazuje primerjavo med tremi različnimi orodji, ki so po obsegu zvočnih analitičnih funkcij najbolj podobne našemu sistemu Audio IJS.

	Spekter	**Filtriranje**	**Segmentacija**	**Ekstrakcija značilk**	**Klasifikacija**	**GUI**
Audacity	✗	✓	✗	✗	✗	✓
pyAudioAnalysis	✓	✗	✓	✓	✓	✗
Audio IJS	✓	✓	✓	✓	✓	✓

Tabela 1.1: Primerjava implementiranih funkcionalnosti med različnimi orodji.

1.2 Motivacija

Tipičen uporabnik želi orodje za obdelavo zvoka, ki na enem mestu nudi vse osnovne funkcije obdelave in enostavno uporabo le-teh. Po pregledu obstoječih rešitev smo ugotovili, da ni orodja, ki bi ponujal celovit nabor osnovnih funkcij in njihovo uporabo v grafičnem uporabniškem vmesniku.

Zato smo želeli razviti orodje oziroma programski vmesnik, ki bi omogočal uporabo funkcij za analizo zvoka. Le-tega bi kot dodatek vključili v okolje za podatkovno rudarjenje Orange, ki obdelave zvoka še ne podpira. S tem

bi vsem uporabnikom omogočili, tudi tistim brez predznanja o zvoku in programiranju, natančno ter enostavno uporabo zgoraj omenjenih funkcij. Tako bi bistveno zmanjšali čas, ki ga uporabnik potrebuje, če želi slednje funkcije uporabiti brez programskega vmesnika.

Za razvoj v Orange smo se odločili zaradi enostavne implementacije dodatkov in ker že v osnovi vsebuje bogato zbirko metod nadzorovanega in nenadzorovanega strojnega učenja, ki jih lahko uporabimo v kombinaciji z novorazvitimi gradniki. Poleg tega gre za odprtokodni sistem, ki je podprt na vseh glavnih operacijskih sistemih (Windows, Linux, Mac OS X). Med drugim je v aktivnem razvoju že od leta 1996 in se je skozi vsa ta leta uveljavljal v panogah strojnega učenja in podatkovnega rudarjenja.

1.3 Cilji in struktura diplomskega dela

Cilj diplomskega dela je razviti orodje oziroma dodatek za Orange, ki bi uporabniku omogočal naslednje funkcije:

- spektralna analiza,
- filtriranje,
- segmentacija,
- ekstrakcija značilk,
- predvajanje zvoka,
- povezljivost z ostalimi gradniki Orange.

V diplomskem delu najprej predstavimo uporabljena orodja in opišemo njihovo funkcionalnost. Podamo nekaj teoretičnega ozadja gradnikov, kjer opišemo spektrogramsko analizo, filtriranje, segmentacijo, ekstrakcijo zvočnih značilk ter klasifikacijo zvoka. Sledi predstavitev razvitih gradnikov orodja Orange, kjer predstavimo vsak gradnik posebej ter opišemo njegovo implementacijo. V nadaljevanju opišemo nekaj primerov uporabe, kjer smo orodje

tudi ustrezno preizkusili. V zadnjem delu opišemo sklepne ugotovitve in podamo nekaj predlogov za nadaljnje delo.

Poglavje 2

Opis metod in orodij za analizo zvoka

Izdelani programski vmesniki temeljijo na že znanih orodjih in tehnologijah, zato bodo v tem poglavju opisani splošno. Nato sledi nekaj teoretskega ozadja, ki je nujno potreben za razumevanje funkcionalnosti implementiranih gradnikov.

2.1 Uporabljena orodja

2.1.1 Python

Python je interpretativen, interaktiven, objektno usmerjen visokonivojski programski jezik [15]. Zasnovan je bil v poznih osemdesetih letih s strani Guida van Rossuma, ki je februarja 1991 kodo javno objavil. Python ima vdelane visokonivojske podatkovne strukture (polje, slovar, moduli, razredi, izjeme itd.), ki omogočajo zelo hitro programiranje. Ima izredno enostavno in elegantno sintakso, a je kljub temu močan in splošno namenski programski jezik. Kot mnogi drugi skriptni jeziki je brezplačen, tudi v komercialne namene in se lahko izvaja na vsakem sodobnejšem računalniku. Program v Pythonu se avtomatsko prevede v platformno neodvisno bajtno kodo, ki se nato interpretira oziroma tolmači. Za Python je bilo narejenih že mnogo

knjižnic, ki nam prihranijo veliko programiranja. Za izdelavo diplomskega dela smo uporabili vrsto knjižnic, med njimi so najpomembnejše **Numpy**, **BioSPPy** in **pyAudioAnalysis**.

Numpy je osnovna in izredno popularna knjižnica za znanstveno računanje v okviru Pythona. Med drugim vsebuje zmogljivo N-dimenzionalno polje objektov, sofisticirane funkcije, orodja za integracijo kode C/C++ in Fortran, funkcije linearne algebre, Furierjeve transformacije in podobno.

BioSPPy je orodje za obdelavo bioloških signalov. Knjižnica združuje metode procesiranja signalov in prepoznave vzorcev, usmerjenih v analizo bioloških signalov. V ozadju BioSPPy je veliko funkcij, ki imajo enačice v knjižnici Scipy. Zapisane so v nekoliko bolj berljivi in intuitivni obliki.

pyAudioAnalysis pokriva širok spekter zvočno analitičnih nalog:

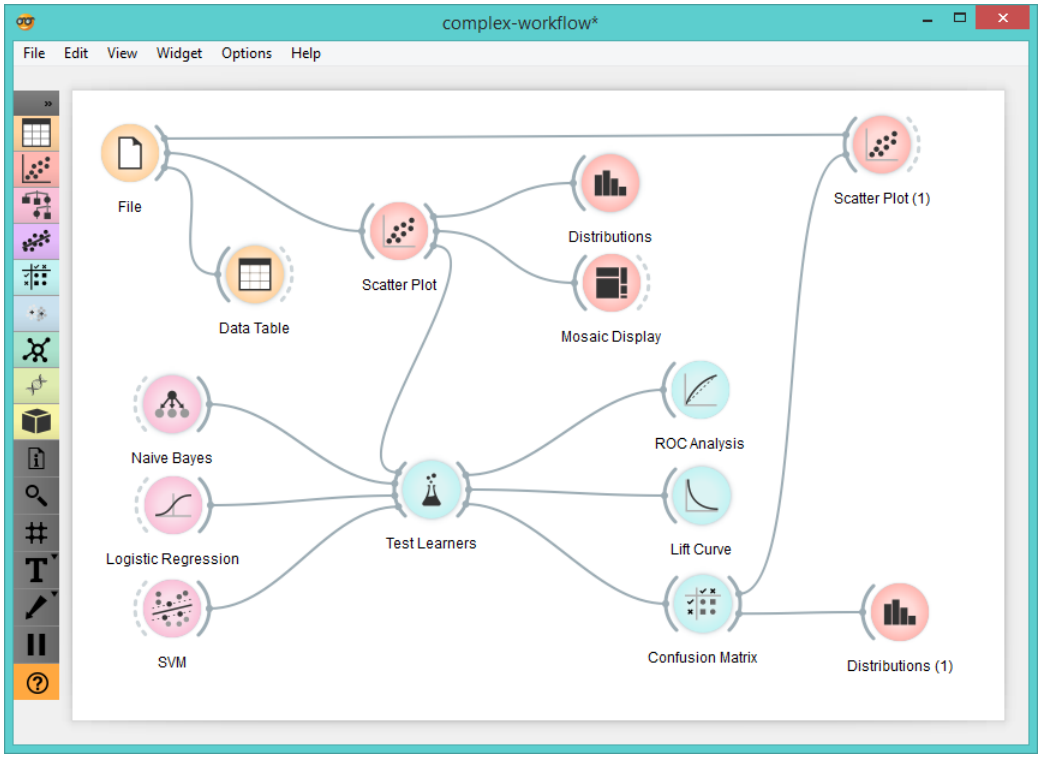
- odkrivanje zvočnih značilk in predstavitev (mfcc, spektrogram, kromogram itd.),
- klasifikacija novih zvokov,
- učenje napovednih modelov, izbiranje vrednosti parametrov in ocenjevanje klasifikatorjev zvočnih segmentov,
- detekcija zvočnih dogodkov,
- izločanje intervalov tišine iz daljših posnetkov,
- ter mnogo drugih opravil.

2.1.2 Orange

Orange je odprtokodno orodje za podatkovno rudarjenje in strojno učenje. Uporabljamo ga lahko preko uporabniškega vmesnika, imenovanega Orange Canvas ali na standarden skriptni način programiranja v Pythonu. Razvijajo ga v Laboratorij za bioinformatiko, na Fakulteti za računalništvo in informatiko, Univerze v Ljubljani, `http://orange.biolab.si`. Sprva je bil Orange

v celoti implementiran v jeziku C++, nakar je zaradi preprostostejšega razvoja in fleksibilnosti razvojni jezik postal Python. S prehodom na Python je glavna prednost postala enostavnejši razvoj grafičnega vmesnika [6].

Orange je namenjen tako izkušenim uporabnikom in programerjem kot študentom in začetnikom, ki šele spoznavajo koncepte strojnega učenja. Orange ponuja mnogo funkcij, ki so v grafičnem vmesniku združene v posamezne kategorije. Poleg osnovnih (izbira podatkov, vizualizacije itd.) je možno naložiti dodatke za analizo časovnih vrst, bioinformatiko in druge.



Slika 2.1: Primer načina povezave gradnikov v Orange.

2.1.3 OpenSmile

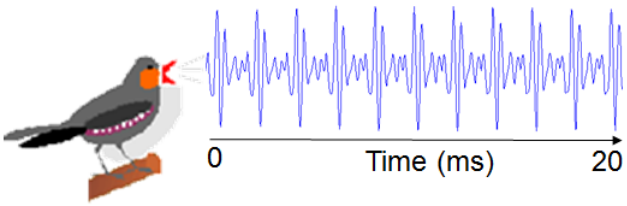
OpenSMILE [8] je orodje za ekstrakcijo velikega nabora zvočnih značilk v realnem času. Napisano je v jeziku C++ in je na voljo v obliki grafičnega vmesnika in v ukazni vrstici. Orodje podpira ekstrakcijo frekvenčnih kepstralnih koeficientov mel, značilk CHROMA in CENS, glasnosti in mnogo

drugih značilk. Glavna lastnost openSMILE je zmožnost inkrementalnega procesiranja in njegove modularnosti. Orodje uporabniku omogoča definicijo lastnih značilk s pomočjo preproste konfiguracijske datoteke.

2.2 Spektralna analiza zvoka

Pri poslušanju ptic nam je takoj jasno, da ima vsako oglašanje značilno ritmično in včasih celo melodično strukturo. Vendar do 1941 [19] ni bilo objektivnega načina potrditve teh opažanj s pomočjo fizikalnega merjenja zvoka. Izum zvočnega spektrografa (sonogram) v laboratoriju Bell je bil pomemben preboj za kvantitativno raziskovanje živalskega oglašanja in nasploh analize zvoka. Sonogram pretvori zvok v preprosto statično sliko, ki razkriva časovno-frekvenčno strukturo vsakega zloga. Sonogramske slike oziroma spektrograme je možno analizirati in primerjati med seboj. To raziskovalcu omogoča opredelitev stopnje podobnosti med različnimi zvoki in kategorizacijo zlogov v različne razrede [1].

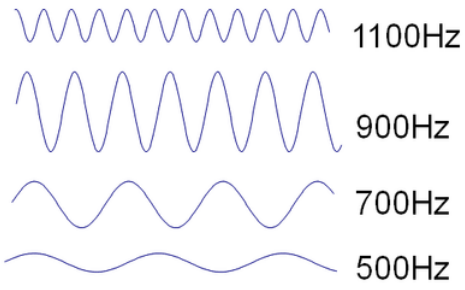
Spektralna analiza ni ravno intuitivna, zato sledi nekaj tehnične razlage, kako izračunati spektrogram. Postopek se začne s snemanjem zvoka. Mikrofon zajame drobna nihanja v zračnem tlaku, ki jih imenujemo zvočni valovi in jih pretvori v električni tok.



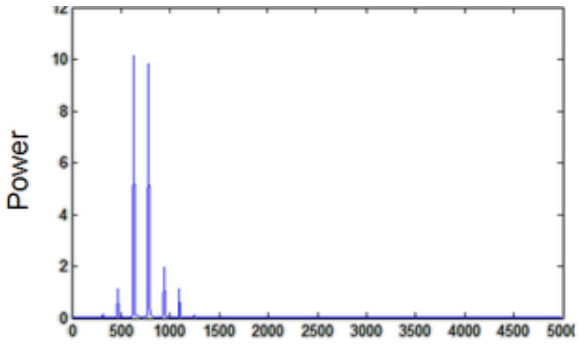
Slika 2.2: Zvok prikazan v časovni domeni.

Nato sledi izračun frekvenčnega spektra, ki ga izračunamo s pomočjo hitre Furierjeve transformacije (FFT), ki je hitra implementacija diskretne Furierjeve transformacije. S transformacijo FFT izračunamo periodično

strukturo v signalu, predstavljeno z množico sinusnih frekvenčnih komponent (Slika 2.3). Na podlagi jakosti posamezne frekvence dobimo frekvenčni spekter (Slika 2.4).



Slika 2.3: Sinusne frekvenčne komponente.



Slika 2.4: Frekvenčni spekter.

Računanje frekvenčnega spektra deluje pri številnih nalogah najbolje, kadar je signal periodičen in stacionaren, zato je pri zvoku smiselno uporabljati kratka okna, npr. 5-20ms. Če pogledamo na celoten zvočni signal, moramo povzeti frekvenčne spektre številnih majhnih oken. To naredimo tako, da vzamemo frekvenčni spekter enega okna, ki ga transponiramo in pretvorimo v barvno skalo. Postopek ponavljamo s prekrivajočimi okni oziroma korakom k, npr., 1-20ms, 2-21ms, 3-22ms za k = 1. Temu postopku pravimo tudi kratko-časovna Fourierjeva transformacija (STFT).

2.3 Filtriranje zvočnih posnetkov

Glasbeniki uporabljajo filtre za oblikovanje zvoka za potrebe njihove umetnosti. Na primer, razvoj fizične oblike violine predstavlja razvoj modela filtra. Izbira lesa, oblika razrezov in vse, kar vpliva na resonanco, določa, kako bo violinsko ogrodje filtriralo signal, ki ga na mostu povzroča vibriranje strun. Ko zvok potuje po zraku, se filtiranje izvaja še naprej, in sicer s strani okolja (človeško uho, raznolike površine ipd.) [16].

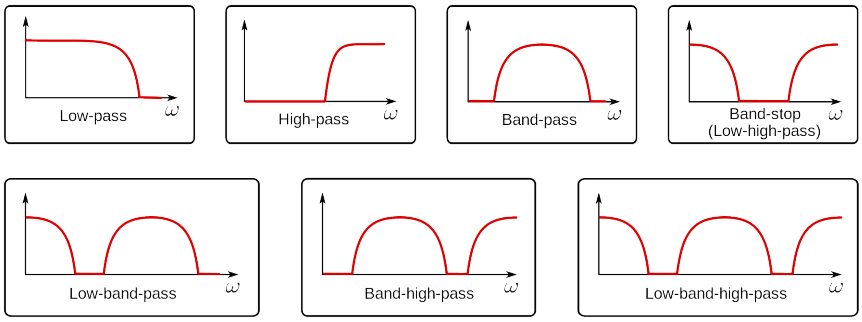
Pri procesiranju signalov je filter naprava oziroma proces, ki iz signala izloči nezaželene komponente. S tehničnega vidika to pomeni odstranitev nekaterih frekvenc ali frekvenčnih pasov, ki nastopajo kot nezaželena komponenta v signalu. Vendar vsi filtri ne delujejo samo v frekvenčni domeni. Še posebej na področju obdelave slik, obstajajo različni cilji filtriranja. Filtri se pogosto uporabljajo v elektroniki in telekomunikacijah, v radiju in televiziji, zvočnem snemanju, radarjih, različnih nadzornih sistemih itd.

V splošnem ločujemo med analognim in digitalnim filtrom. Na začetku so prevladovali analogni filtri. Digitalni filtri so bili le matematični opisi analognih. S pojavom digitalnih signalnih procesorjev so se digitalni filtri razširili na vsa elektrotehnična področja. K njihovi popularizaciji je poleg tega vplivala tudi lažja spremenljivost le-teh, saj ni bilo potrebe po fizičnem spreminjanju strukture, večja delovna zanesljivost, manjša občutljivost na vplive iz okolja [4].

Digitalne filtre lahko naprej delimo v dve skupini: z neskončnim impulznim odzivom tipa IIR in filtre s končnim impulznim odzivom tipa FIR. Poleg tega jih lahko delimo še glede na njihov frekvenčni odziv:

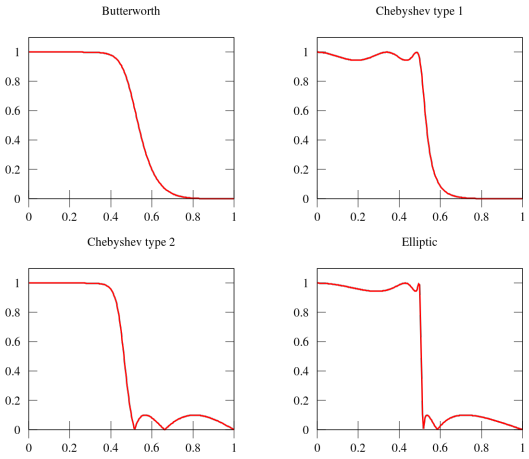
- nizkopasovni filtri (ang. low-pass): zadušijo frekvence, višje od podane mejne frekvence,
- visokopasovni filtri (ang. high-pass): zadušijo frekvence, nižje od podane mejne frekvence,
- pasovnoprepustni filtri (ang. band-pass): zadušijo frekvence izven dveh podanih mejnih frekvenc,

- pasovnozaporni filtri (ang. band-stop): zadušijo frekvence znotraj dveh podanih frekvenc.



Slika 2.5: Grafični prikaz delovanja filtrov s prikazanim območjem prepuščanja.

Poleg osnovnih filtrov, katerih oblika krivulje je takšna kot na Sliki 2.5, poznamo filtre, katerih krivulje se precej razlikujejo med sabo (Slika 2.6). Leti so določeni z aproksimacijskim polinomom in vsak določa značilnosti prenosne funkcije filtra. Poznamo Butterworthove, Eliptične, Besseljeve, Chebyshejeve tipe filtrov, in mnogo drugih.



Slika 2.6: Prikaz tipov filtrov in oblik njihove polinomske krivulje.

2.4 Segmentacija zvoka

Segmentacija zvoka je izredno pomembna naloga v različnih aplikacijah za predprocesiranje in obdelavo zvoka, kot so [14]:

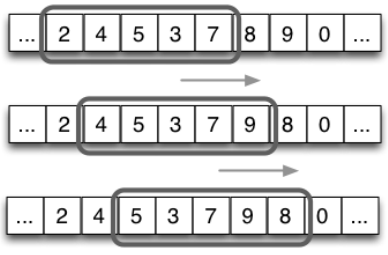
- diarizacija govorca,
- sledenje govorca,
- avtomatsko prepoznavanje govora,
- klasifikacija zvočnih posnetkov.

Segmentacija zvoka se osredotoča na delitev neprekinjenega zvočnega signala v segmente homogene vsebine. Izraz homogene je mogoče definirati na več različnih načinov, zato obstaja inherentna težava pri zagotavljanju globalne definicije tega koncepta [9].

V grobem delimo segmentacijo na statično in dinamično. Primer **statične** je implementacija algoritma drsečega okna (Slika 2.7), ki na vhod sprejme dva parametra:

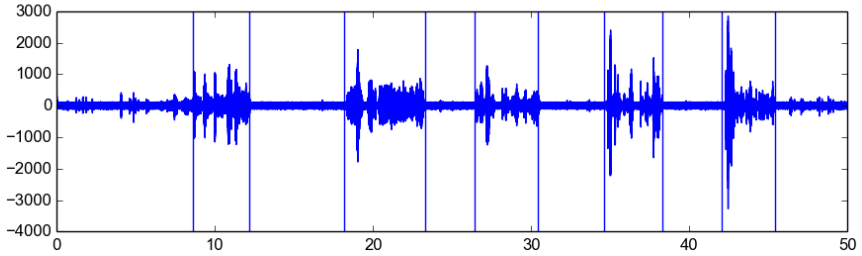
- velikost okna, ki nam pove želeno velikost segmenta,
- korak, ki nam pove, s kakšnim zamikom bomo premikali okno.

Slabost tega preprostega algoritma je izbira parametrov. V primeru, da pred segmentacijo nimamo vpogleda v strukturo posnetka, težko določimo parametre, ki bodo posnetek dobro razdelili na homogene segmente. To slabost v določeni meri rešimo s prej opisano spektralno analizo.



Slika 2.7: Prikaz delovanja algoritma drsečega okna z dolžino okna 5 in korakom 1.

Dinamična segmentacija (Slika 2.8) odpravi zgoraj opisani problem. V osnovnem principu deluje na podlagi odstranitve intervalov tišine iz posnetka. Na ta način dobimo segmente, ki vsebujejo polno informacijo. Pri tem žal ne moremo vedeti ali gre za del šuma ali ne. To lahko v določeni meri odpravimo s pregledom zvočnega spektra in nadaljnjim ustreznim filtriranjem.



Slika 2.8: Prikaz delovanja dinamične segmentacije z odstranjenim območjem tišine.

Nadgradnja obeh opisanih načinov je nadzorovana oziroma nenadzorovana segmentacija. Pri nadzorovani je potrebno predznanje za klasifikacijo in segmentacijo vhodnega signala. To dosežemo z uporabo naučenega klasifikatorja, ki razvrsti segmente v posamezne razrede ali z uporabo pristopa skritega modela Markova za doseganje skupne razvrstitve segmentacije.

Pri nenadzorovani segmentaciji nimamo klasifikatorja, zato se nad detektiranimi segmenti, uporabi metoda gručenja.

2.5 Ekstrakcija zvočnih značilk

Avtomatska analiza zvoka oziroma glasbe, ki je shranjena v obliki digitalnega zvočnega signala, zahteva prefinjen proces pridobivanja informacij. Na primer, triminutni zvočni posnetek, shranjen v nestisnjeni digitalni obliki, je predstavljen v zaporedju skoraj 16 milijonov številk (3 [minute] * 60 [sekund] * 2 kanala * 44100 [frekvenca vzorčenja]). V primeru razumevanja tempa mora biti vseh 16 miljonov številk pretvorjenih v eno numerično vre-

dnost [18].

Ekstrakcija zvočnih značilk je osnova za kakršnokoli nadaljnjo podatkovno rudarjenje. To je proces filtriranja ogromne količine surovih podatkov v veliko kompaktnejšo predstavitev, ki zajame višji nivo informacij o osnovni zvočni vsebini. Najbolj osnovna digitalna zvočna predstavitev je zaporedje kvantiziranih pulzov v času, ki ustreza diskretiziranemu premiku zračnega tlaka, ki se je zgodil, ko je bil posnet zvok. Ljudje (in mnogi drugi organizmi) zaznavajo svoje slišno okolje z določanjem periodičnih zvokov s specifičnimi frekvencami. Na zelo osnovni ravni, glasba sestoji iz zvokov, ki se začenjajo in ustavljajo v različnih časovnih trenutkih. Zato so predstavitve zvoka, ki imajo ločen pomen časa in frekvence, najpogosteje uporabljena kot prvi korak pri ekstrakciji zvočnih značilk.

Način združevanja zvočnih značilk, je odvisen od vrste informacije, ki jih poskušajo zajeti. Na abstraktni ravni lahko identifirciramo različne visokonivojske vidike zvočnih posnetkov. Hierarhična organizacija v času se nanaša na ritem, hierarhična organizacija v frekvenci se imenuje harmonija. Timbre je zvočna kakovost oziroma obarvanost, ki razlikuje zvoke iste frekvence in glasnosti različnih virov [2]. Na primer, isti odsek glasbe, odigrane v kitarističnem kvartetu bi imelo v primerjavi s saksofonskim kvartetom, pri istem odigranem tempu, različne timbralne oziroma barvne značilnosti.

Zvočne značilke v grobem delimo na:

- timbralno teksturne značilke,
- ritemske značilke,
- višinske oziroma harmonične značilke,
- druge zvočne značilke.

Timbralno teksturne značilke [7] so najbolj razširjene zvočne značilke in so do sedaj zagotavljale najboljše rezultate. Še en dejavnik njihove popularnosti je njihova dolga zgodovina na področju prepoznave govora. Obstaja veliko variant v ekstrakciji timbralnih značilk, a vse imajo nek skupni slog. Najprej se opravi določena oblika časovno-frekvenčne analize, kot

je STFT, kateremu sledijo koraki povzemanja, ki imajo za posledico bistveno manjši vektor značilk. Podoben pristop lahko uporabimo z drugimi časovno-frekvenčnimi reprezentacijami zvoka. V to skupino spadajo spektralne značilke, mel-frekvenčno kepstralne značilke in mnogo drugih.

Ritemske značilke so pomembne komponente v MIR sistemih in so že več kot 20 let aktivno področje raziskovanja. Najbolj osnoven pristop izračuna teh značilk je iskanje povprečnega tempa celotnega posnetka, ki ga je mogoče opredeliti kot pogostost, s katero bi človek tapkal z nogo ob poslušanju zvočnega posnetka. Ritmične informacije so hierarhične narave in tempo je samo ena stopnja hierarhije. Druge stopnje, ki se pogosto uporabljajo v MIR sistemih so tatum (najkrajši, najpogostejši časovni interval), doba in takt.

Harmonične značilke pridejo v poštev pri identifikaciji priredb pesmi oziroma avtomatski detekciji akordov. V teh primerih želimo imeti predstavo o višini glasbe in ne o tem, kateri instrumenti oziroma zvoki igrajo. Možno bi bilo, da bi v ta namen uporabili programsko opremo, ki bi avtomatsko prevedla glasbo v notni zapis, a le-ta trenutno še ni dovolj robustna, da bi jo lahko zanesljivo uporabili. Najpogostejša predstavitev te kategorije so višina in razred tona (PCP). Cilj PCP značilk je opisati tonaliteto kosa glasbe, neodvisno od dejanske višine. Tem deskriptorjem pravimo tudi CHROMA značilke. V najbolj preprostem primeru, ko je poltonski spekter usmerjen v eno oktavo, je rezultat 12 dimenzionalni vektor, neodvisno od tega čez koliko oktav se poltonski spekter razteza [7].

Poleg zgoraj opisanih deskriptorjev obstajajo še **druge zvočne značilke**, ki imajo drugačen vidik na glasbeno vsebino. Kot primer lahko navedemo kategorizacijo instrumentov v določenem zvočnem intervalu.

2.6 Klasifikacija zvoka

Klasifikacija je pomembna naloga v okviru zvočnega podatkovnega rudarjenja. Nekateri raziskovalci se osredotočajo na klasifikacijo iz zvočnih posnet-

kov, medtem ko se drugi ukvarjajo s kategorizacijo v različne pripadnostne razrede. Najbolj splošna klasifikacija se osredotoča na klasifikacijo glasbenega žanra oziroma sloga. Poleg tega obstaja kar nekaj drugih nalog klasifikacije, kot so klasifikacija razpoloženja oziroma čustev, instrumentov, bolezni itd.

Izraz klasifikacija zvoka se navadno nanaša na opis naloge, povezane z govorom oziroma videom, kjer je glavni cilj identifikacija zvoka v tri različne razrede: govor, glasba ali okolje. Ta groba klasifikacija se lahko uporabi za pomoč pri segmentaciji videoposnetkov in odločanju, kje uporabiti funkcijo prepoznave govora.

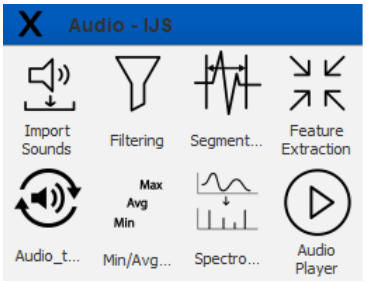
A pogosto klasifikacija zvoka ni trivialna naloga, saj se velikokrat srečamo z ogromno količino podatkov oziroma posnetkov, kateri so si lahko zelo podobni, zašumljeni, različnih dolžin in z drugimi napakami. V takšnem primeru je posnetke težko razvrstiti in se je treba lotiti ustreznega predprocesiranja, ki je ključen za dobro nadaljnjo klasifikacijo.

Faza predprocesiranja zajema stopnje, opisane v zgornjih podpoglavjih. Najprej opravimo **spektralno analizo**, kjer razberemo zašumljenost signala in intervale zlogov. Po potrebi nadaljujemo s **filtracijo**, ki zaduši morebiten šum, ki se pojavlja v posnetkih. Na podlagi razbranih dolžin zlogov nadaljujemo s **segmentacijo**, kjer zagotovimo konsistentnost dolžin med posameznimi posnetki, kjer je lahko eden dolg 1 drug pa 30 sekund. V zadnji stopnji ostane **ekstrakcija značilk**, ki je bistvena za nadaljnjo klasifikacijo. Z njo pridobimo zgoščene informacije o posnetkih, ki vsebujejo veliko več informacij, kot jih dobimo v osnovni predstavitvi zvoka (amplituda v odvisnosti od časa). Poleg tega v veliki meri odpravimo problem časovne kompleksnosti izvedbe klasifikacije, ki je napram surovim podatkov, veliko večja.

Poglavje 3

Razviti gradniki za procesiranje zvoka v sistemu Orange

Za dosego cilja diplomskega dela smo za novo kategorijo "Audio - IJS" (Slika 3.1) izdelali osem gradnikov za predprocesiranje in analizo zvoka, ki bodo natančneje opisani v tem poglavju.

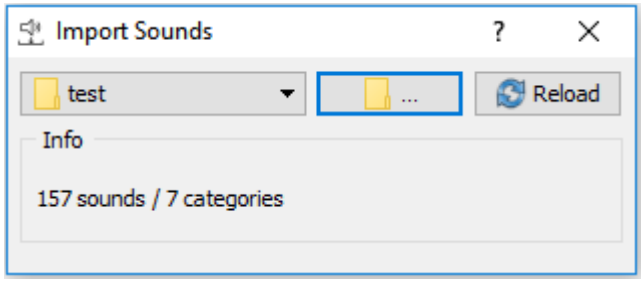


Slika 3.1: Razvita kategorija "Audio - IJS" z osmimi gradniki za predprocesiranje in analizo zvoka.

3.1 Vnos zvočnih posnetkov

Import Sounds je gradnik (Slika 3.2), ki omogoča vnos zvočnih posnetkov. Struktura le-tega temelji na gradniku Import Images kategorije Image Embedding, ki omogoča vnos slik. Uporabnik ima na voljo tri funkcije:

- izbira mape s posnetki,
- ponovno nalaganje,
- preklic izvajanja vnosa posnetkov.



Slika 3.2: Prikaz gradnika Import Sound.

Pri izbiri mape mora biti pozoren, da so posnetki z več razredi urejeni po podmapah, kjer ime mape predstavlja ime kategorije oziroma pripadnostnega razreda. Uporabniku se ob prvi izbiri map, le-ta shrani v seznam nedavno uporabljenih map, ki jih ima v nadaljevanju možnost izbrati neposredno iz menija v gradniku.

Iskanje datotek s posnetki je implementirano v razredu *SoundScan*, ki na vhod v konstruktor sprejme dva argumenta:

- začetna mapa,
- seznam podprtih formatov.

Skeniranje poteka rekurzivno skozi vse podmape od začetne podane poti. Na poti so upoštevani samo posnetki, katerih format je v seznamu podprtih formatov. Trenutno je podprto samo branje posnetkov wav, ki jih preberemo s pomočjo knjižnice Scipy (koda 3.1).

Izpis 3.1: Branje zvočnih posnetkov.

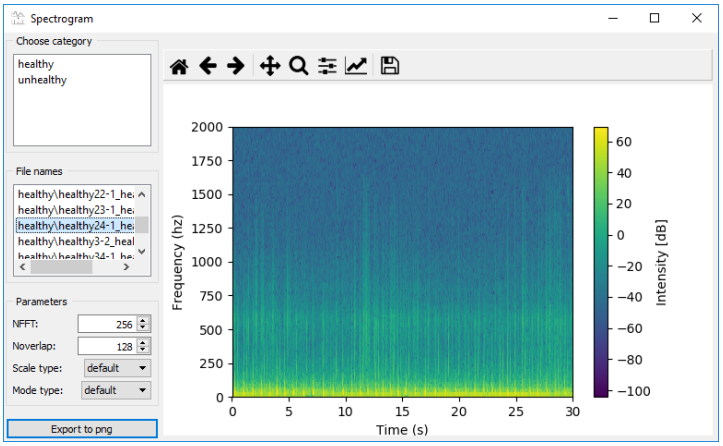
```
framerate, X = read(dirname)
```

Metoda nam vrne celoštevilsko vrednost (ang. integer), ki predstavlja frekvenco vzorčenja in prebrane podatke v obliki polja, ki jo podpira knjižnica Numpy. Ob uspešnem branju posnetkov, gradnik pošlje na izhod tabelo v formatu Orange s podatki o kategoriji posnetka (razredni atribut), ime in absolutno pot do datoteke s posnetkom, številu vzorcev, dolžino in frekvenco vzorčenja (meta atributi). Slednja oblika tabele Orange se prenaša med vsemi drugimi gradniki in se lahko uporabi pri snovanju novih dodatkov oziroma funkcionalnosti.

3.2 Spektrogram

Gradnik za izris spektograma (Spectrogram), prikazan na Sliki 3.3, omogoča enostaven vpogled v frekvenčno domeno zvočnega posnetka (frekvenca v odvisnosti od časa). Iz izrisa je možno razbrati zvočne zloge in zašumljenost, kar olajša izbiro parametrov filtriranja in segmentacije. Gradnik omogoča več funkcionalnosti:

- filtriranje posnetkov na podlagi kategorije,
- izbira parametrov za izračun spektra,
- izvoz spektrogramov (označenih posnetkov) v formatu png,
- manipuliranje z izrisanim spektrogramom.



Slika 3.3: Prikaz gradnika Spectrogram.

Izris spektrograma je implementiran s pomočjo modula pyplot knjižnice Matplotlib, ki z dodano orodno vrstico omogoča manipulacijo s spektrogramom. Ob izbiri imena posnetka iz seznama "File names" se v vhodni tabeli Orange poišče pripadajoča instanca in iz nje se izlušči absolutno pot posnetka. Le-ta se nato uporabi za branje posnetka (isto kot v Import Sound). Izhod metode (poleg ostalih parametrov) se uporabi kot vhod v funkcijo izrisa (Izpis 3.2).

Izpis 3.2: Izris spektrograma.

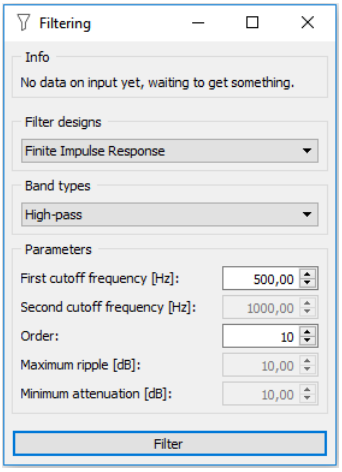
```
plt.specgram(data, NFFT=self.nfft, noverlap=self.noverlap,
    Fs=frameset, mode=self.mode_combo.currentText(),
    scale=self.scale_combo.currentText())
```

3.3 Filtriranje

Po pregledu spektrograma posnetkov lahko začnemo s filtriranjem ("Filtering"), prikazanem na Sliki 3.4. Namen gradnika je odstraniti oziroma zadušiti moteče frekvenčne komponente zvočnega posnetka. Gradnik omogoča naslednje funkcije:

- izbiro med štirimi tipi filtrov (nizkopasovni, visokopasnovni itd.),

- izbiro med šestimi dizajni (chebyshev, butterworth itd.),
- prilagajanjem različnih parametrov posameznih filtrov.



Slika 3.4: Prikaz gradnika Filtering.

Ob kliku na gumb "Filter" se sproži metoda *call_filter*, ki je implementirana znotraj razreda gradnika. Kot argument sprejme podatke o tipu in dizajnu filtra ter parametre, ki so potrebni za izbrano kombinacijo tipa in dizajna. V glavnem delu metode se najprej preveri vrsta dizajna ter tip, nakar sledi klic funkcije filtriranja, ki je implementirana znotraj knjižnice BioSPPy. Funkcija poleg izbranih parametrov sprejme še surove podatke posnetka in frekvenco vzorčenja.

Izpis 3.3: Klic funkcije filtriranja s knjižnico BioSPPy.

```
filtered = st.filter_signal(self.data.X[i], ftype=filterType,
    band=filterBand, order=order, frequency=first_cutoff,
    sampling_rate=self.data.metas[i][-1])
```

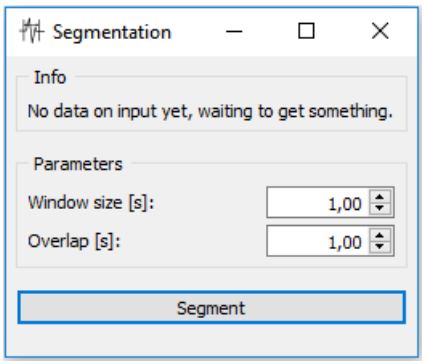
Na podoben način, kot je prikazano v zgornjem izpisu 3.3, lahko uporabnik gradniku doda poljuben filter. Ob uspešnem zaključku filtriranja metoda vrne filtrirane podatke v Numpy arrayu, nakar se le-ti zakodirajo nazaj v format wav ter začasno zapišejo na disk. Na ta način so ob nadaljnji uporabi

dostopni drugim gradnikom.

3.4 Segmentacija

Po filtriranju ali pregledu spektrograma lahko uporabnik s pomočjo gradnika segmnentacije ("Segmentation"), prikazanega na Sliki 3.5, posamezne posnetke razreže oziroma segmentira na različno velike enote. Segmentacija je implementirana na osnovi algoritma drsečega okna, opisanega v Poglavju 2.4. Gradnik zahteva, tako kot algoritem, dva parametra:

- Window size: velikost želenega okna, v sekundah,
- Overlap: velikost prekrivanja oziroma zamike med posameznimi segmenti, v sekundah.



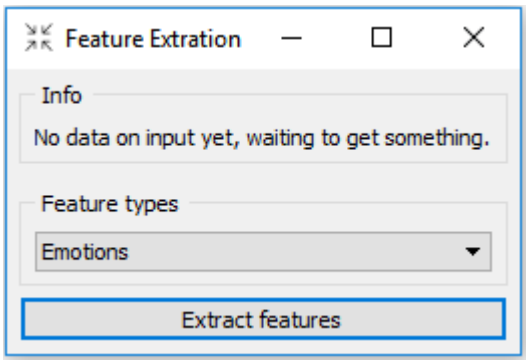
Slika 3.5: Prikaz gradnika Segmentation.

S klikom na gumb "Segment" se sproži funkcija *call_segmentation* znotraj katere opravimo vse delo. Funkcija najprej zgradi nov objekt tipa *Segmentation* modula *segmentation.py*, ki ga je možno uporabljati tudi izven razreda gradnika. Nato se kliče metoda *segment_all* objekta *Segmentation*, ki sproži nadaljnjo segmentacijo s funkcijama *segmentation* in *sliding_window*. Prva funkcija najprej zgenerira segmente z uporabo druge in le-te zakodira v format wav ter začasno zapiše na disk. Segmenti so ločeni med seboj z uporabo zaporednega indeksa, ki ga dodamo na konec imena posnetka.

3.5 Ekstrakcija značilk

Gradnik "Feature Extraction", prikazan na Sliki 3.6, je namenjen ekstrakciji značilk, katere v nadaljevanju uporabimo pri klasifikaciji posameznih posnetkov oziroma segmentov.

Z uporabo knjižnice OpenSMILE smo implementirali štiri konfiguracije oziroma značilke MFCC, PLP, Chroma in Emotions. Za implementacijo značilke pyAudio je bila uporabljena knjižnica pyAudioAnalysis.



Slika 3.6: Prikaz gradnika Segmentation.

Ob kliku na gumb "Extract features" se sproži funkcija *call_feature_extraction* z argumentom id-ja tipa značilke. Funkcije najprej zgenerira objekt *FeatureExtraction* (modul *feature_extraction.py*), v katerem so implementirane klicoče metode zgoraj naštetih značilk. Vsaka izmed konfiguracij izračuna vektorje istih dimenzij za vse posnetke in le-te vrne v obliki polja Numpy. S tem se izognemo težavam pri klasifikaciji, saj v primeru nekonsistentnih dolžin vektorjev, napovedni modeli ne morejo biti zgrajeni. Na izhod gradnika je poleg standardnih meta podatkov in kategorije podana tudi matrika vseh vektorjev značilk. Na ta način je uporabniku omogočena direktna uporaba gradnikov klasifikacije, ki jih Orange vsebuje že v standardnem paketu.

Uporabniku je omenjen modul na voljo tudi izven dosega gradnikov, kjer lahko poljubno dodaja svoje ali druge funkcije ekstrakcije značilk.

3.6 Najmanjša, povprečna in največja vrednost

Gradnik "Min\Avg\Max" uporabimo pri izvajanju klasifikacije nad posameznimi segmenti. Le-ta uporabniku omogoča, da zgradi nove značilke na podlagi klasifikacijskih napovedi za vsak segment. S pomočjo zgoraj opisanih modulov oziroma gradnikov je potrebno posnetke najprej »razrezati« na poljubno veliko segmentov ter nato le-te z že obstoječimi gradniki v Orange ustrezno klasificirati in tako pridobiti ustrezne napovedi. V nadaljevanju gradnik napovedi posameznega klasifikatorja uporabi pri izračunu minimalne, maksimalne in povprečne vrednosti za vse segmente posameznega posnetka. Na ta način pridobimo tri nove značilke za vsak uporabljen klasifikator.

Vse uporabljene funkcije so implementirane znotraj razreda gradnika. Izračun značilk se začne takoj, ko gradnik dobi podatke na vhod. Izvajanje se nadaljuje s funkcijo *make_classification_dict*, ki iz vhodnih podaktov zgenerira slovar slovarjev, kjer je ključ prvega ime posnetka (brez indeksa segmenta), vrednost je nov slovar, kjer je ključ ime klasifikatorja in vrednost, seznam petih vrednosti: seštevek klasifikacijskih napovedi, minimum, maksimum napovedi, števec in ime razreda. Če je slovar uspešno zgrajen, izračun nadaljuje funkcija *process_data*, ki iz danega slovarja s pomočjo seštevka klasifikacijskih napovedi in števca, izračuna povprečno vrednost ter zgradi matriko Y (razred) in matriko metas (ime posnetka). V končni fazi funkcija *make_orange_table* zgradi tabelo Orange, ki jo gradnik pošlje na izhod.

3.7 Pretvornik

Gradnik pretvornik "Audio to raw" je namenjen pretvorbi zvočnih posnetkov v surovo obliko. Le-ta pride v poštev pri kombiniranju s preostalimi gradniki paketa Orange, ki branja posnetkov na vhodu ne podpirajo.

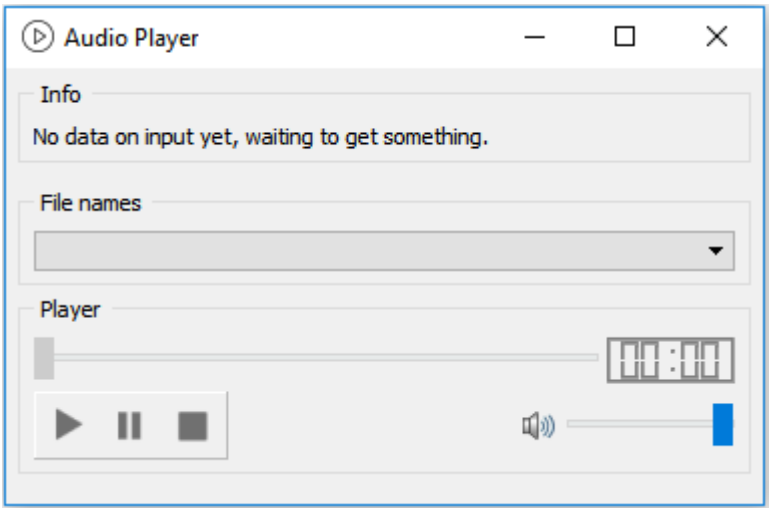
Glavna funkcija znotraj razreda gradnika je *make_square_array*, ki poskrbi za generiranje kvadratne matrike iz podatkov prebranih posnetkov, saj

so le-ti v večini primerov nekonsistentih dolžin. Znotraj funkcije se najprej preveri dolžina najdaljšega posnetka. Nato se le-ta uporabi ob izračunu števila nedefiniranih vrednosti, ki jih je potrebno dodati na konec vsakega polja prebranih vrednosti posnetka. Ob koncu izvedbe algoritma funkcija vrne kvadratno matriko surovih podatkov z dodanimi nedefiniranimi vrednostmi. Matrika se nato direktno uporabi ob konstrukciji tabele Orange, ki na izhod poleg omenjene matrike pošlje tudi meta podatke, uporabljene pri vseh ostalih gradnikih.

3.8 Predvajalnik zvoka

Predvajalnik zvoka "Audio Player", prikazan na Sliki 3.7, je zadnji izmed osmih gradnikov. Uporabniku omogoča poslušanje posnetkov. Znotraj gradnika so implementirane naslednje kontrole oziroma prikazi:

- izbira posnetka s seznama imen,
- časovni drsnik, s katerim se lahko postavite na poljuben odsek posnetka,
- gumb za začetek predvajanja, prekinitev in skok na začetek,
- reguliranje glasnosti,
- prikaz časa predvajanja.



Slika 3.7: Prikaz gradnika "Audio Player."

Vse naštete kontrole so implementirane znotraj razreda gradnika s pomočjo modula Phonon. Gre za multimedijsko ogrodje, namenjeno prezentaciji zvočnih in video vsebin v aplikacijah PyQt. Najpomembnejši funkciji razreda sta *setupActions* in *setupGUI*. Znotraj prve so implementirane zgoraj naštete kontrole, znotraj druge pa prikaz grafičnega vmesnika.

Poglavje 4

Primer uporabe: Predprocesiranje in klasifikacija posnetkov EKG srca

Kronično srčno popuščanje predstavlja globalno pandemijo, za katero trenutno boleha več kot 26 miljonov ljudi po celem svetu. Je en glavnih dejavnikov smrtnosti bolnikov s kardiovaskularnimi boleznimi. V Evropi in Severni Ameriki letno povzroči več kot milijon hospitalizacij. Metode za odkrivanje kroničnega srčnega popuščanja se lahko uporabijo kot preventiva za izboljšanje zgodnje diagnoze in izogibanje hospitalizacijam, kar močno izboljša kakovost bolnikovega življenja.

V tem poglavju je predstavljeno predprocesiranje in klasifikacija posnetkov EKG srca, ki so bili v veliko raziskavah uporabljeni za detekcijo kroničnega srčnega popuščanja.

Podatki

Domena podatkov so posnetki elektrokardiograma oziroma EKG srca ljudi z (razred *unhealthy*) in brez (razred *healthy*) srčnega popuščanja. Le-te smo dobili v že naprej razdeljeni testni in učni množici v razmerju 30/70. Posnetkov je 153, kjer jih 113 pripada zdravim in 39 obolelim ljudem. V povprečju

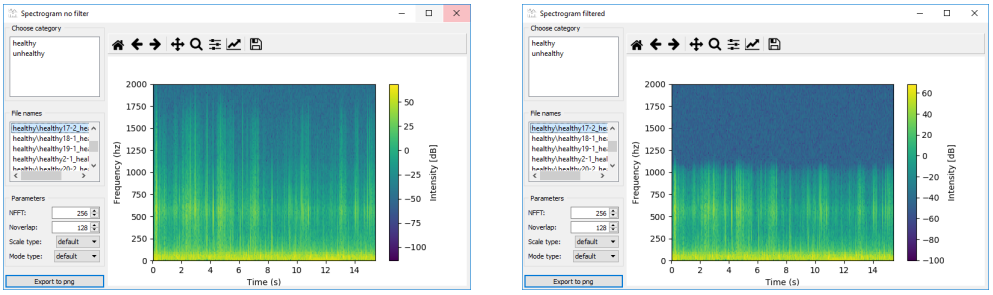
so le-ti dolgi okoli 20-30 sekund.

Predprocesiranje

V fazi predprocesiranja, ki je ustrezno označena na spodnji delovni shemi 4.2, smo najprej začeli z vnosom učnih in testnih posnetkov, katere smo ločeno naložili v dve instanci gradnika "Import Sound."

Z nadaljnjim pregledom spektrograma smo ugotovili dolžine zlogov in zašumljenost signala, kar nam je bilo v pomoč pri izbiri parametrov filtriranja in segmentacije.

Za izvedbo filtriranja smo uporabili nizkofrekvenčni filter butterworth s frekvenco dušenja 1 kHz. Rezultat filtriranja je prikazan na Sliki 4.1



(a) Originalen posnetek (b) Filtriran posnetek

Slika 4.1: Razliko med spektrogramoma originalnega in filtriranega posnetka, v katerem so zadušene vse frekvence nad 1000 Hz.

Pri izvedbi segmentacije smo uporabili premikajoče okno dolžine ene sekunde, saj se le-ta najbolj ujema s frekvenco bitja srca in 0.5 sekundni zamik segmentov, ki v veliki meri prepreči izgubo informacij [11].

V zadnji fazi, fazi ekstrakcije značilk, smo uporabili konfiguracijo MFFC, ki vrne 38 mel-frekvenčnih komponent.

S tem postopkom smo končali s predprocesiranjem in nadaljevali s klasifikacijo.

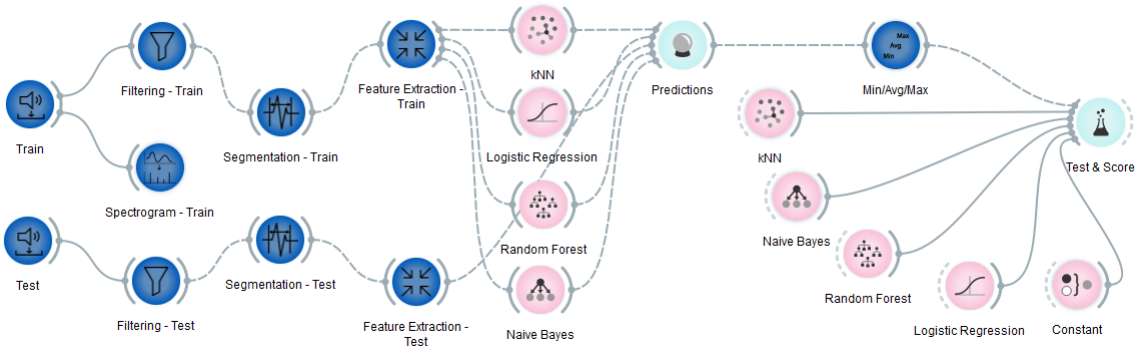
Klasifikacija

V fazi klasifikacije, prikazane na Sliki 4.2, smo začeli z učenjem različnih kla-

sifikacijskih modelov. V nadaljevanju smo s pomočjo gradnika "Predictions" in povezave modelov s testnimi primeri, dobili napovedi posameznih klasifikatorjev za vsak posnetek. Izhod gradnika smo preusmerili na vhod gradnika Min/Avg/Max, ki nam služi kot del "Stacking," saj iz rezultatov oziroma napovedi zgradi tri nove (minimum, povprečje, maksimum) značilke za vsak posamezen klasifikator. Rezultate gradnika smo preusmerili v gradnik za vrednotenje napovednih modelov "Test & Score," kjer smo za ocenjevanje rezultatov uporabili prečno preverjanje na pet različnih modelih. Pri nastavljanju parametrov modelov smo ohranili privzete nastavitve, kakršne so v 3.4.5 verziji Orange. Rezultati klasifikatorjev so prikazani v Tabeli 4.1.

Metoda	**AUC**	**klasifikacijska točnost**	**F1**	**preciznost**	**priklic**
logistična regresija	0.975	0.932	0.955	0.941	0.970
naključni gozdovi	0.969	0.909	0.939	0.939	0.939
naivni Bayes	0.969	0.886	0.921	0.967	0.879
k-najbližjih sosedov	0.963	0.909	0.939	0.939	0.939
večinski klasifikator	0.500	0.750	0.000	0.000	0.000

Tabela 4.1: Rezultati vrednotenja napovednih modelov.



Slika 4.2: Prikaz primera končne delovne sheme.

Poglavje 5

Sklepne ugotovitve

V okviru diplomskega dela smo se seznanili s predprocesiranjem in klasifikacijo zvoka. Razvili smo dodatek za Orange, znotraj katerega je implementiranih osem gradnikov za analizo zvoka. S tem smo obogatili nabirko dodatkov v Orangu in omogočili vnos in analizo zvočnih posnetkov. Pripadajoča koda je zaenkrat še pod okriljem Instituta Jožef Stefan, kjer je bila izdelana. Za pridobitev kode se lahko obrnete na odsek za inteligentne sisteme E9, kjer boste dobili vsa potrebna navodila za namestitev dodatka "Audio - IJS."

Tekom izdelave diplomskega dela je bil na to temo napisan članek [10] z naslovom "JSI Sound – a machine-learning tool in Orange for simple biosound classification,', ki je bil objavljen v delavnici IJCA, imenovani "Bioinformatics and Artificial Intelligence (BAI)." V članku je opisano delovanje in arhitektura izdelanega orodja in odlični rezultati, ki so bili doseženi s testiranjem na različnih domenah posnetkov.

V diplomskem delu smo se določenih funkcionalnosti lotili bolj grobo, zato obstaja nekaj možnosti za izboljšave. Le-te so vidne pri gradnikih filtriranja, kjer bi lahko dodali več filtrov. Pri gradniku segmentacije bi lahko implementirali dinamično segmentacijo. Pri gradniku za ekstrakcijo značilk bi lahko dodali še druge znane značilke.

Kljub naštetim možnostim izboljšave menimo, da smo zadostili zastavljenim ciljem dela.

Literatura

[1] Introduction to spectral analysis. `http://soundanalysispro.com/manual-1/spectral-analysis/introduction-to-spectral-analysis`.

[2] Timbre. `https://en.wikipedia.org/wiki/Timbre`.

[3] Dmitry Bogdanov, Nicolas Wack, Emilia Gómez, Sankalp Gulati, Perfecto Herrera, Oscar Mayor, Gerard Roma, Justin Salamon, José R Zapata, Xavier Serra, et al. Essentia: An audio analysis library for music information retrieval. In *ISMIR*, pages 493–498, 2013.

[4] Bojan Bratuž. *Detekcija in klasifikacija zvokov bobnov v realnem času*. PhD thesis, Univerza v Ljubljani, 2007.

[5] Paul M Brossier. The aubio library at mirex 2006. *Synthesis*, 2006.

[6] Janez Demšar and Blaž Zupan. Orange: Data mining fruitful and fun-a historical perspective. *Informatica*, 37(1), 2013.

[7] Florian Eyben. *Real-time speech and music classification by large audio feature space extraction*. Springer, 2015.

[8] Florian Eyben, Martin Wöllmer, and Björn Schuller. Opensmile: the munich versatile and fast open-source audio feature extractor. In *Proceedings of the 18th ACM international conference on Multimedia*, pages 1459–1462. ACM, 2010.

[9] Theodoros Giannakopoulos. pyaudioanalysis: An open-source python library for audio signal analysis. *PloS one*, 10(12):e0144610, 2015.

[10] Martin Gjoreski, Borut Budna, Anton Gradišek, and Matjaž Gams. Jsi sound – a machine-learning tool in orange for simple biosound classification. IJCA, Bioinformatics and Artificial Intelligence (BAI), 2017.

[11] Martin Gjoreski, Monika Simjanoska, Anton Gradišek, Ana Peterlin, Gregor Poglajen, and Matjaž Gams. Chronic heart failure detection from heart sounds using a stack of machine-learning classifiers. 2017.

[12] Benoit Mathieu, Slim Essid, Thomas Fillon, Jacques Prado, and Gaël Richard. Yaafe, an easy to use and efficient audio feature extraction software. In *ISMIR*, pages 441–446, 2010.

[13] Brian McFee, Colin Raffel, Dawen Liang, Daniel PW Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto. librosa: Audio and music signal analysis in python. In *Proceedings of the 14th python in science conference*, pages 18–25, 2015.

[14] David Rybach, Christian Gollan, Ralf Schluter, and Hermann Ney. Audio segmentation for speech recognition using segment features. In *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, pages 4197–4200. IEEE, 2009.

[15] Michel F Sanner et al. Python: a programming language for software integration and development. *J Mol Graph Model*, 17(1):57–61, 1999.

[16] Julius Orion Smith. *Introduction to digital filters: with audio applications*, volume 2. Julius Smith, 2008.

[17] Audacity Team. Audacity (r): Free audio editor and recorder [computer program], 2017.

[18] George Tzanetakis. Audio feature extraction. *Music data mining*, page 43, 2011.

[19] Lisa Yount. *Forensic science: From fibers to fingerprints.* Infobase Publishing, 2007.