UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Miha Krašovec

Prepoznavanje sestavov v posnetkih ljudske glasbe

DIPLOMSKO DELO
NA VISOKOŠOLSKEM STROKOVNEM ŠTUDIJU

Mentor: doc. dr. Matija Marolt

Ljubljana, 2011

Univerza
v Ljubljani

Fakulteta za računalništvo
in informatiko

Tržaška 25
1000 Ljubljana, Slovenija
telefon: 01 476 84 11
faks: 01 426 46 47
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 00051/2010
Datum: 02.11.2010

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **MIHA KRAŠOVEC**

Naslov: **PREPOZNAVANJE SESTAVOV V POSNETKIH LJUDSKE GLASBE**
ENSEMBLE RECOGNITION IN FOLK SONG RECORDINGS

Vrsta naloge: Diplomsko delo visokošolskega strokovnega študija prve stopnje

Tematika naloge:

V okviru diplomske naloge preučite področje prepoznavanja inštrumentov, sestavov in žanrov v glasbenih posnetkih. Na teh temeljih razvijte sistem za prepoznavanje tipičnih sestavov v posnetkih slovenske ljudske glasbe in ga ovrednotite na testni množici terenskih posnetkov.

Mentor:

doc. dr. Matija Marolt

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Dekan:

prof. dr. Nikolaj Zimic

I Z J A V A O A V T O R S T V U

diplomskega dela

Spodaj podpisani/-a Miha Krašovec,

z vpisno številko 63010226,

sem avtor/-ica diplomskega dela z naslovom:

Prepoznavanje sestavov v posnetkih ljudske glasbe

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal/-a samostojno pod mentorstvom (naziv, ime in priimek)

  doc. dr. Matija Marolt

- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela

- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki »Dela FRI«.

V Ljubljani, dne 11. 5. 2011 Podpis avtorja/-ice:

Zahvala

Zahvaljujem se mentorju doc. dr. Matiji Maroltu za vso pomoč in napotke pri izdelavi diplomskega dela. Zahvaljujem se tudi vsem drugim za njihovo spodbudo, predvsem pa Katji za vso potrpežljivost v času pisanja.

Kazalo

Seznam uporabljenih kratic in simbolov

MIR – Pridobivanje informacij iz glasbe (angl. Music Infomation Retrieval)

ISMIR – The International Society for Music Information Retrieval

MIREX – Music Information Retrieval Evaluation eXchange

GPL – Splošno dovoljenje GNU (angl. GNU General Public License)

FFT – Hitra Fourierjeva transformacija (angl. Fast Fourier Transform)

DFT – diskretna Fourierjeva transformacija (angl. Discrete Fourier Transform)

DCT – Diskretna kosinusna transformacija (angl. Discrete Cosine Transform)

MFC – Mel frekvenčni kepstrum (angl. Mel-frequency Cepstrum)

MFCC – Mel frekvenčni kepstralni koeficienti (angl. Mel-frequency Cepstral Coefficients)

NMF – Ne-negativna matrična faktorizacija (angl. Non-negative Matrix Factorization

GMM – Gaussovi modeli (angl. Gaussian Mixture Models)

SVM – Metoda podpornih vektorjev (angl. Support Vector Machines)

MLP – Več-nivojski perceptron (angl. Multilayer Perceptron)

DBN – Deep Belief Network

Povzetek

Področju avtomatskega prepoznavanja glasbenih inštrumentov v zvočnih posnetkih se posveča vedno več raziskovalcev, predstavljene rešitve pa se zaenkrat še ne morejo kosati s sposobnostjo prepoznave inštrumentov pri ljudeh. To še posebno velja za polifonične posnetke. Na tekmovanju MIREX algoritmi dosegajo večinoma od 70- do 75-odstotno natančnost prepoznave.

V okviru tega diplomskega dela predstavljam področje avtomatskega prepoznavanja glasbenih sestavov v polifoničnih posnetkih, ki je zelo podobno prepoznavanju inštrumentov. Problem sem poenostavil z omejitvijo zvočnih posnetkov na slovensko ljudsko glasbo. Posnetke sem najprej razdelil na 10-sekundne segmente. Nad temi segmenti se izračuna devet zvočnih značilk: MFCC, tempo, frekvenca pojavljanja not, hitrost prehajanja skozi ničlo, spektralni upad, svetlost zvoka, nesimetričnost spektra, spektralni centroid in spektralna ravnost. Za zajem značilk sem uporabil MIRToolbox (dodatek za orodje MATLAB), ker ima vse najpogosteje uporabljane algoritme že implementirane. Tako dobljene značilke se podajo algoritmu za strojno učenje LMT, implementiranem v orodju Weka, ki segmente posnetkov klasificira v pet razredov (solo harmonika, Bela krajina, Prekmurje, Rezija – glasba in Rezija – petje). To je tudi končni rezultat programa.

Na ta način dobljeni rezultati so bili dobri. Pri testiranju učnih podatkov s prečnim preverjanjem z deset pregibi je bilo pravilno klasificiranih 94 % posnetkov. V drugem delu je bilo testiranje izvedeno z uporabo testnih posnetkov, ki so pripadali enemu od petih razredov. Klasifikacijska točnost je bila v tem primeru 83 %. V zadnjemu sklopu testiranj sem uporabil neobdelane terenske posnetke, kjer je bilo pravilno klasificiranih 86 % segmentov.

V zaključku sem predlagal tudi nekaj možnih izboljšav algoritma, ki bi mu mogoče lahko povečale natančnost in robustnost.

Ključne besede: prepoznavanje glasbenih inštrumentov, zajem značilk, strojno učenje, lastnosti zvoka.

Abstract

More and more researchers are starting to explore the field of automatic recognition of musical instruments within audio recordings, but so far their presented solutions cannot compete with the human ability of instrument recognition. This is especially true for polyphonic recordings. Algorithms participating in the MIREX competition usually achieve 70 to 75 percent recognition accuracy.

In my thesis I am presenting automatic recognition of musical instrument groups, which is very similar to instrument recognition. The problem was simplified by limiting the recordings to Slovene folk music. Audio recordings are first divided into 10 second segments. For each of the segments nine audio features are calculated: MFCC, tempo, frequency of note onsets, zero-crossing rate, spectral roll-off, sound brightness, spectral irregularity, spectral centroid and spectral flatness. MIRToolbox (a MATLAB plug-in) was used for feature extraction in which all of the most commonly used algorithms already implemented. A machine learning algorithm LMT, implemented in Weka, is then used on these features to classify audio segments into five classes (solo accordion, Bela krajina, Prekmurje, Resian music and Resian singing).

Results obtained by this method were good. 10-fold cross-validation used to test training data correctly classified 94% of recordings. For the next test I used recordings that belonged to one of the five classes. Classification accuracy achieved this way was 83%. In the last part, unedited field recordings were used, where 86% of segments were correctly classified.

To conclude I also suggested a few possible improvements to the algorithm which could increase its accuracy and robustness.

Key words: musical instrument recognition, feature extraction, machine learning, sound properties

1. Uvod

Čeprav so začetki računskih pristopov k zvoku in glasbi opazni že v 50. letih prejšnjega stoletja, so se večji premiki pri raziskavah začeli v 70. letih, ko je bilo ustanovljenih kar nekaj (tudi mednarodnih) združenj in inštitutov. V začetku 70. let so tako nastali International Computer Music Association, International Computer Music Conference in Center for Computer Research in Music and Acoustics (CCRMA) na stanfordski univerzi, konec 70. let pa Institute for Research and Coordination Acoustic/Music (IRCAM) v Parizu. Leta 1977 je bil ustanovljen tudi Computer Music Journal. Z vse večjo specializacijo raziskav po področjih računskih pristopov k zvoku in glasbi se je ustanovilo še kar nekaj različnih konferenc. Najpomembnejše tri so nastale okrog leta 2000, in sicer International Conference on Digital Audio Effects leta 1998, (za področje tega diplomskega dela najbolj pomembna) International Conference on Music Information Retrieval (ISMIR) leta 2000 in International Conference on New Interfaces for Musical Expression (NIME) leta 2001.

MIR ali Music Information Retrieval je eno izmed specializiranih področij zgoraj omenjenih raziskav. Ukvarja se s pridobivanjem informacij iz glasbe. To področje je interdisciplinarno in tako ni omejeno samo na računalništvo, čeprav se ga v okviru diplomskega dela dotikam prav s tega vidika. Raziskave v sklopu MIR poskušajo zagotavljati robustne metode za pomoč glasbenim ljubiteljem, profesionalcem in industriji pri lociranju, pridobivanju in doživljanju glasbe. Vključeni so raziskovalci s področja muzikologije, kognitivnih in informacijskih znanosti, bibliotekarstva in še mnogi drugi. MIR med drugim pokriva: računske metode za klasifikacijo, združevanje in modeliranje, programsko opremo za pridobivanje informacij iz glasbe, formalne metode in podatkovne baze, interakcijo in vmesnike med računalnikom in človekom, analizo glasbe in predstavitev znanja, glasbene arhive, knjižnice in digitalne zbirke ter podobno.

V okviru ISMIR poteka izmenjava novic, idej in rezultatov prek teoretičnih ali praktičnih del. To je tudi njen glavni namen. Vsako leto se prirejajo istoimenske mednarodne konference. S povezovanjem raziskovalcev, razvijalcev, učiteljev in knjižničarjev, študentov ter profesionalnih uporabnikov služi ta konferenca tudi kot forum ter ponuja uvodne in poglobljene informacije glede specifičnih domen. V sklopu ISMIR je organizirano vsakoletno mednarodno tekmovanje MIREX (Music Information Retrieval Evaluation eXchange) [13], kjer se pomerijo raziskovalci s svojimi MIR-algoritmi, rezultati pa so objavljeni na spletni strani tekmovanja. Že od ustanovitve leta 2000 prevzema vodilno vlogo v svetovnem merilu na svojem področju in prav zaradi tega je zelo uporaben vir člankov in raziskav s področja MIR uradna spletna stran ISMIR [9].

Avtomatično prepoznavanje inštrumentov v zvočnih posnetkih je eno izmed področij, ki jih pokriva MIR. Ima pomembno vlogo pri razvoju aplikacij za avtomatsko indeksiranje glasbe. S tem povezano je seveda tudi posledično precej učinkovitejše in predvsem hitrejše iskanje po današnjih vse večjih digitalnih glasbenih arhivih. Hkrati pa avtomatično prepoznavanje inštrumentov zaradi velike kompleksnosti mešanice zvočnih signalov različnih inštrumentov predstavlja precejšen problem. Temu področju se kljub njegovi relativni pomembnosti ne posveča dosti raziskovalcev. Po drugi strani tudi ni veliko raziskav s področja prepoznave inštrumentov v polifonskih posnetkih. Večinoma se raziskovalci lotevajo problema na nivoju posameznih not in (v manjšem obsegu) glasbenih fraz monofonskih posnetkov. Tudi glede natančnosti prepoznave lahko iz rezultatov dosedanjih raziskav zaključimo, da pri večjem naboru inštrumentov ljudje še vedno med njimi razlikujemo bolje kakor stroji.

Še bolj se omenjena razlika pokaže pri razločevanju družin inštrumentov (npr. pihala, godala, brenkala, itd.). Leta 1999 je Judith Brown [1] pokazala, da je možno prepoznavati med štirimi pihali z natančnostjo, primerljivo človeški. Istega leta je Marques [7] izdelal sistem, ki je ločeval med osmimi inštrumenti s 70-odstotno natančnostjo. Čeprav je njegov sistem ločeval med več inštrumenti kot sistem Judith Brown, ni dosegal tako velike natančnosti.

Poleg prepoznave inštrumentov me je zanimalo tudi avtomatsko razpoznavanje žanrov glasbe. Moj pristop k problemu namreč zajema prvine obojega. Razpoznavanje žanrov je med drugimi ena izmed nalog tekmovanj MIREX. Algoritmi morajo biti sposobni prepoznati deset žanrov, med katerimi so si nekateri tudi podobni (npr. jazz in blues). Zaenkrat dosegajo večinoma okrog 70-odstotno natančnost.

Heittola, Klapuri in Virtanen [4] so predstavili nov pristop k razpoznavi posameznih inštrumentov v polifoničnih posnetkih. V polifoničnih posnetkih, ki jih sestavlja več inštrumentov, je precej verjetno, da bodo posamezni zvoki povzročali motnje pri drugih istočasnih zvokih. Te motnje se lahko zmanjša z razčlenitvijo posnetka na signale posameznih zvočnih virov, kar je bila tudi osnova te rešitve.

Postopek zajema tri korake. Najprej se določijo višine posameznih zvokov v vsakem časovnem okviru. Te višine se nato uporabijo v algoritmu za izdelavo časovno neprekinjenega toka not. Z uporabo ne-negativne matrične faktorizacije ali NMF (angl. Non-negative Matrix Factorization) se posnetek loči na signale posameznih zvočnih virov, na teh signalih pa se izračuna MFC-koeficiente, ki se jih nato uporabi pri klasifikaciji z Gaussovimi modeli ali GMM (angl. Gaussian Mixture Models). S to metodo so med 19 inštrumenti in pri šestzvočni polifoniji so dosegli 59-odstotno natančnost prepoznave.

Hamel, Wood in Eck [2] so predstavili metodo, ki je bolj sorodna temi te diplomske naloge. Njihova metoda ne prepoznava posameznih glasbil, ampak njihove razrede. Izbrali so 7 razredov: klavir, kitara, bas, orgle, pihala, trobila in godala. Na solo glasbilih so dosegli 88-odstotno, na polifonskih posnetkih pa okrog 74-odstotno uspešnost prepoznave.

Kot večina drugih so se tudi avtorji te metode pri izbiri značilk najbolj ozirali na MFC-koeficiente. Ti koeficienti so bili izračunani na 32ms oknih s korakom 10 ms. Uporabili so prvih 20 MFC-koeficientov in tudi njihove izpeljave ΔMFCC (delta-MFCC) in ΔΔMFCC (delta-delta-MFCC). Poleg MFC-koeficientov so uporabili še osem spektralnih značilk, in sicer spektralni centroid, razpršenost podatkov (angleško spread ali standardni odklon), koeficient simetrije (angleško skewness), koeficient sploščenosti (angleško kurtosis), koeficient zmanjševanja spektralne amplitude (angleško spectral decrease), spektralni naklon (angleško spectral slope), spektralni pretok (angleško spectral flux) in spektralni upad (angleško spectral roll-off). Posnetki so bili razdeljeni na enosekundna okna. Za vsako okno sta bili izračunana srednja vrednost in standardni odklon, zaradi česar je imela vsaka od značilk dve vrednosti. Vektor značilk je tako vseboval 136 vrednosti.

Testirali so tri različne klasifikacijske modele: Multilayer Perceptron (MLP), metodo podpornih vektorjev (SVM, angl. Support Vector Machines) in Deep Belief Network (DBN). Globokih nevronskih mrež zaradi zahtevnosti treniranja v splošnem niso veliko uporabljali. V zadnjem času pa je napredek omogočil enostavnejše treniranje teh mrež. Od takrat je DBN dal dobre rezultate na različnih področjih, kot je npr. prepoznava slik in govora. Do te raziskave se DBN na področju prepoznave glasbil ni uporabljajo, a so njeni avtorji pokazali, da se obnese bolje kot MLP in SVM. To so opazili predvsem, ko je bilo število učnih podatkov manjše. Iz tega sklepajo, da se je DBN sposoben iz podatkov drugih glasbil naučiti visokonivojske značilke za razlikovanje tudi manj pogostih glasbil.

Glavni cilj tega diplomskega dela je bil preučiti možnost klasifikacije posnetkov v nekaj dokaj splošnih kategorij. Nabor posnetkov je bil zaradi poenostavitve problema omejen

na slovensko ljudsko glasbo, iz tega nabora posnetkov pa je bilo nato izbranih pet kategorij: solo harmonika, Bela krajina, Prekmurje, Rezija – glasba in Rezija – petje. Pri izdelavi diplomske naloge sta bili uporabljeni večinoma orodji MATLAB (ter v sklopu MATLABa še MIRToolbox) in Weka. Glavni program je napisan v MATLABu, od tam pa se v ozadju uporablja Weka, ki je dostopna tudi v obliki knjižnice v Javi. Zaradi določenih omejitev MATLABove podpore programskemu jeziku Java je bilo treba za uporabo Weke izdelati še vmesno knjižnico. Sicer pa je uporaba programa preprosta. Treba je vnesti učne podatke in seznam posnetkov, ki jih želimo klasificirati. Po obdelavi posnetkov dobimo nato vrnjen seznam razredov, za katere program sklepa, da pripadajo posameznim posnetkom. Uspešnost klasifikacije, kot bo pozneje razvidno iz rezultatov, ni vrhunska, me je pa vseeno pozitivno presenetila.

2. Ozadje in algoritmi

Pri avtomatskem prepoznavanju glasbenih inštrumentov je bolj raziskano področje monofonskih posnetkov. Monofonski posnetki so posnetki, v katerih naenkrat igra le ena nota oziroma je naenkrat prisoten le en glas. Pri teh posnetkih je pridobivanje značilnosti zvoka že dokaj enostavno, hkrati pa obstaja že veliko različnih algoritmov. Obstajajo tudi metode prepoznavanja glasbil, ki na sicer dokaj omejenih množicah dajejo že precej dobre rezultate. Z novimi raziskavami se tako nabor kot tudi natančnost prepoznanih inštrumentov povečujeta.

Pri polifonskih posnetkih pa je prepoznavanje glasbil kompleksnejše. V nasprotju z monofonskimi posnetki so polifonski sestavljeni iz več med seboj neodvisnih glasov, ki so melodično sorodni. V enem posnetku je namreč več zvočnih virov sočasno, ki vplivajo drug na drugega. Večinoma se pri polifonskih posnetkih raziskovalci prepoznavanja glasbil lotevajo z iskanjem značilk neposredno iz originalnega zvočnega signala. Nekateri pa z razčlenjevanjem posnetkov na posamezne vire zvoka odstranijo ali vsaj zmanjšajo motnje, ki jih posamezni zvočni viri povzročajo drug drugemu. Razčlenjevanje že samo po sebi predstavlja precejšen problem. Rezultat razčlenitve namreč ni nujno najboljši, kar seveda v veliki meri vpliva na končni rezultat. Po drugi strani pa se s tem iz enega polifonskega dobi več monofonskih posnetkov, kjer je, kot je že omenjeno, prepoznavanje glasbil precej bolj raziskano področje.

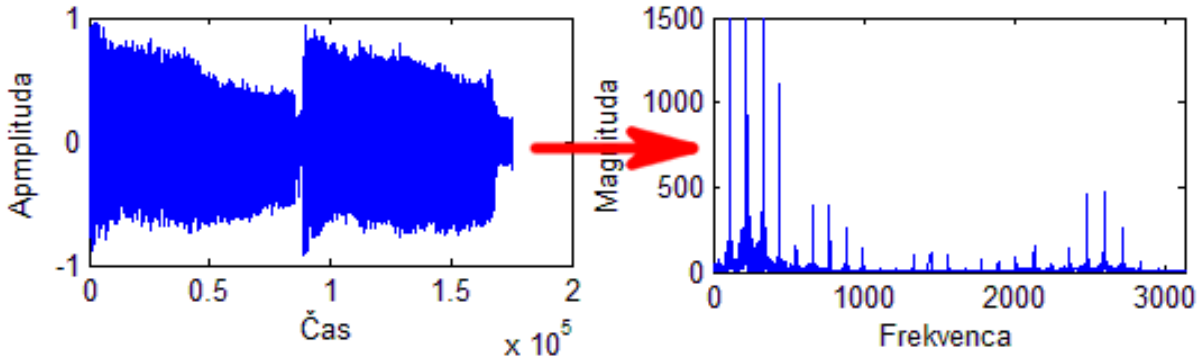
V okviru mojega diplomskega dela pa je bil pristop k problemu drugačen in enostavnejši. Zanimalo me je, ali je možno dobiti uporabne rezultate, če se polifonske posnetke obravnava kot monofonske brez razčlenitve na sestavne zvoke. Že na začetku je bil predviden zelo omejen nabor kategorij posnetkov. Precej omejen je bil tudi sam izvor posnetkov, saj so vsi s področja slovenske ljudske glasbe. Kategorij ne sestavljajo samo posamezna glasbila, ampak tudi celi sestavi. Razlike med posnetki se kažejo tudi glede na regijo izvora posnetka in ne samo glede na uporabljena glasbila. S tega vidika ima moj pristop nekako zabrisano mejo med prepoznavanjem glasbil in prepoznavanjem žanrov. Vsebuje namreč elemente obojega.

Omenjene kategorije se obravnava vse enako. Pri vseh se predpostavlja, da so na posnetkih različni glasbeni sestavi, čeprav je v kategoriji lahko tudi en sam inštrument. S tega vidika seveda lahko govorimo o prepoznavanju glasbil. Tu so me posnetki zanimali glede na njihove zvočne značilnosti, predvsem barva zvoka.

Po drugi strani pa je lahko kategorija določena ne samo z različnimi glasbenimi sestavi, temveč tudi z značilnostmi glasbe iz določene regije. Ko kategorije ločimo še na ta način, pa že lahko začnemo razmišljati tudi v smeri prepoznavanja žanrov. Različne regije namreč lahko uporabljajo enake inštrumente, vendar jih igrajo na različne načine. Tu sem v programu upošteval samo tempo.

V nadaljevanju bom podrobneje opisal algoritme in metode, ki sem jih uporabil v svojem programu oziroma so pomembne za boljše razumevanje te diplomske naloge.

2.1. Fourierjeva transformacija



Slika 1: Prikaz pretvorbe zvočnega signala iz časovnega v frekvenčni prostor, kot se to zgodi z uporabo Fourjerjeve transformacije. Na levi je signal prikazan kot sinusno valovanje v času. Na desni so prikazane frekvence, ki sestavljajo ta signal v celotnem trajanju signala, torej neodvisno od časa.

Fourierjeva transformacija je matematična operacija, ki razcepi signal na njegove sestavne frekvence. V primeru glasbenega inštrumenta predstavlja Fourierjeva transformacija matematično predstavitev amplitud posameznih frekvenc, ki sestavljajo zaigrano noto. Prvotni signal je odvisen od časa, kar pomeni, da je predstavljen v časovnem prostoru. Fourierjeva transformacija pa je odvisna od frekvence in je tako predstavljena v frekvenčnem prostoru. Signal torej prestavi iz časovnega v frekvenčni prostor. Možno jo je tudi posplošiti na diskretne strukture, kot so npr. končne grupe. Učinkovito izračunavanje takih struktur omogoča algoritem hitre Fourierjeve transformacije (FFT ali fast Fourier transform), ki je tudi nujen pri visoko hitrostnih izračunih v računalništvu.

Računanje diskretne Fourierjeve transformacije (DFT) po definiciji le-te je velikokrat prepočasno za praktično uporabo. Enačba zanjo je:

$$F_n = \sum_{k=0}^{N-1} f_k e^{\frac{-2\Pi i n k}{N}} \qquad (1)$$

FFT rešuje prav ta problem, saj do enakega rezultata pride veliko hitreje. Izračun DFT za N točk po definiciji metode ima časovno zahtevnost $O(N^2)$. FFT lahko enak rezultat doseže s časovno zahtevnostjo $O(N \log N)$. Razlika v hitrosti je znatna, še posebno pri izračunavanju večjega števila točk.

Na področju MIR je Fourierjeva transformacija (oziroma natančneje njena izpeljanka FFT) nepogrešljiva pri spektralni analizi.

2.2. Višina tona

Ena od definicij višine tona je naslednja: »Za zvok lahko rečemo, da ima določeno višino, če se zanesljivo ujema s frekvenco čistega tona poljubne amplitude,« [3]. V nasprotju s frekvenco, ki je določena objektivno in jo lahko izmerimo, je višina tona subjektivna. Je slušna zaznava, ki ji poslušalec določi ton, za kar poskrbijo možgani. Šumom in pokom višine

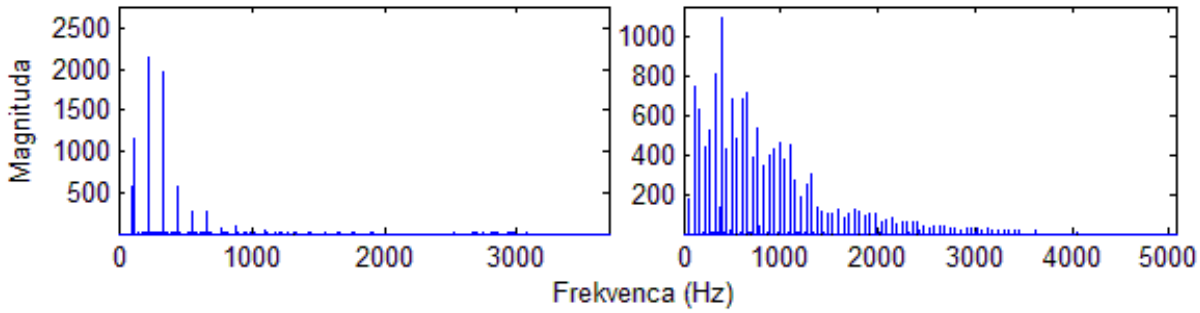
tona ni mogoče določiti.

Določanje višine tona kompleksnih zvokov je večinoma možno s primerjanjem s čistimi toni, a ne vedno. Npr. v zvoku, sestavljenem iz tonov pri 1000 Hz in 1200 Hz, je možno slišati do štiri različne višine: osnovni 1000 Hz in 1200 Hz, obenem pa še 200 Hz in 2200 Hz.

Najnižja harmonična frekvenca v zvoku se imenuje osnovna frekvenca. Ta je zelo povezana z višino tona, a ne povsem. Tudi če osnovna frekvenca ni prisotna, se po navadi še vedno zaznava enako višino tona.

Višina tona je povezana tudi z glasnostjo, čeprav v manjšem obsegu kot s frekvenco. Najbolj je to opazno pri frekvencah pod 1000 Hz in nad 2000 Hz. Višina tona pri nizkih frekvencah se niža ob povišanju glasnosti. Npr. ton pri 200 Hz se bo pri veliki glasnosti zdel nižji, kot kadar je komaj slišen. Podobno se zgodi nad 2000 Hz, kjer se pri višji glasnosti višina tona poveča.

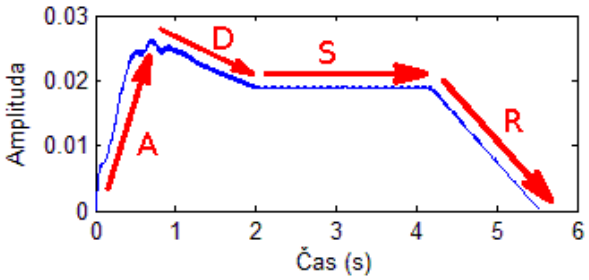
2.3. Barva zvoka



Slika 2: Levo je prikazan frekvenčni spekter tona A2 (110 Hz), zaigranega na flavti. Desno je prikazan frekvenčni spekter tona A2, zaigranega na trobenti.

Zaradi barve zvoka je mogoče razločevati inštrumente, tudi če vsi igrajo isto noto. Barva zvoka je torej neodvisna od višine tona. Enotne definicije barve zvoka zaenkrat še ni.

Eden od pomembnih atributov pri določanju barve zvoka je prisotnost višjih harmoničnih frekvenc in njihova amplituda. Na sliki 2 sta prikazana frekvenčna spektra dveh posnetkov zvoka višine 110 Hz. Spekter zvoka flavte je prikazan na levi in je dokaj podoben čistemu tonu, tako da ima njegov spekter malo višjih harmoničnih frekvenc. Na desni je prikazan spekter zvoka trobente. Razločno se vidi veliko večje število višjih harmoničnih frekvenc, zaradi česar tudi zvok ni tako čist kot zvok flavte.



Slika 3: Ovojnica ADSR šestsekundnega posnetka trobente.

Primerjava frekvenčnih spektrov pa ni zadosten pogoj za določanje barve zvoka. Posnetek, ki ga predvajamo naprej in nazaj, ima v obeh primerih enak spekter, poslušalcu pa se zvok zdi drugačen. Tu se izkaže kot zelo pomembna tudi ovojnica ADSR (angl. ADSR Envelope), prikazana na sliki 3. ADSR je angleška kratica za vzpon (angl. attack), spust (angl. decay), trajanje (angl. sustain) in sprostitev (angl. release). Na sliki 3 so s puščicami označeni deli ovojnice, kakor si sledijo v času. Najprej pride vzpon (A) do najvišje amplitude signala. Vzponu sledi spust (D) do amplitude, ki se vzdržuje (S), dokler glasbenik piha v inštrument. Nato pride na vrsto sprostitev (R), kjer glasbenik postopoma zmanjšuje moč pihanja, dokler trobenta ne utihne. Ovojnica opisuje, kako se amplituda zvoka spreminja s časom.

V eni od raziskav [8] so raziskovalci usposobljenim poslušalcem predvajali posnetke inštrumentov, katerim so odrezali prvo polovico sekunde. S tem so iz ovojnice odstranili vzpon in vsaj del spusta. Nekaj inštrumentov, kot npr. oboa, so poslušalci zanesljivo prepoznali, veliko drugih pa so pomešali. Velikokrat je bil tako tenorski saksofon zamenjan s klarinetom, altsaksofon pa z rogom.

V drugi raziskavi [8] pa so raziskovalci ustvarili zvoke z mešanjem frekvenčnega spektra enega glasbila z ovojnico ADSR drugega. V nekaterih primerih se je izkazalo, da je za prepoznavo pomembnejši spekter (oboa, tuba, fagot, klarinet), pri drugih (predvsem flavta) ovojnica, pri določenih inštrumentih (pozavna, rog) pa sta bila podobno pomembna oba atributa.

Barvo zvoka določajo tudi drugi atributi, vendar sta frekvenčni spekter in ovojnica ADSR najpomembnejša.

2.4. Lestvica mel

Lestvico mel (angleško mel scale) so poimenovali Stevens, Volkman in Newman leta 1937. Ime izhaja iz besede melodija, kar nakazuje, da je lestvica osnovana na primerjavi višine tonov. To je lestvica višine tonov, za katere poslušalec oceni, da so enako razmaknjeni med seboj. Referenčna točka med lestvico mel in normalnimi meritvami frekvence je določena tako, da je zaznana višina tona pri 1000 melih enaka tonu pri 1000 hertzih pri 40 decibelih nad poslušalčevo mejo sluha.

Do približno 500 Hz se lestvici približno ujemata, od 500 Hz naprej pa so razmaki med toni, ki se poslušalcem zdijo na enakih razdaljah, vedno večji. Štiri oktave na lestvici hertzev nad 500 Hz poslušalci na mellestvici ocenijo kot približno dve oktavi. Za lestvico mel obstaja več enačb in pristopov.

Popularna enačba za pretvorbo frekvence *f* hertzev v *m* melov je

$$m = 1127 \log_e\left(\frac{f}{700} + 1\right) \qquad (2)$$

njen inverz pa

$$f = 700\left(\mathrm{e}^{\frac{m}{1127}} - 1\right) \qquad (3)$$

2.5. MFCC

Kratica MFCC izhaja iz angleške besedne zveze mel-frequency cepstral coefficients in predstavlja koeficiente, ki skupaj tvorijo MFC ali mel-frequency cepstrum. MFC temelji na linearni kosinusni transformaciji logaritmov spektra moči na nelinearni lestvici melfrekvenc. MFC koeficienti opisujejo obliko spektra z majhnim številom koeficientov.

Koeficienti so izpeljani iz kepstralno prikazanega zvočnega posnetka. Kepstrum si lahko predstavljamo kot spekter spektra. Ime izhaja besede spectrum, ki ima obrnjen vrstni red prvih štirih črk. V grobem ga dobimo tako, da rezultat Fourierjeve transformacije zvočnega signala vzamemo kot nov signal, nad katerim ponovno izvedemo Fourierjevo transformacijo. Razlika med kepstrumom in melfrekvenčnim kepstrumom je, da so pri MFC frekvenčni pasovi na enakomernih razdaljah na lestvici mel. To omogoča boljši približek delovanju človeškega sluha v primerjavi z linearno razmaknjenimi frekvenčni pasovi normalnega kepstruma. Koeficienti melfrekvenčnega kepstruma so po navadi izpeljani na naslednji način:

1. Za zvočni posnetek dobimo Fourierjevo transformacijo.
2. Tako dobljeni spekter preslikamo na lestvico mel z uporabo trikotnih prekrivajočih se oken.
3. Izračunamo logaritme za vsako od velikosti melfrekvenc iz spektra.
4. Seznam logaritmov iz prejšnje točke obravnavamo kot signal in nad njim izvedemo diskretno kosinusno transformacijo.
5. Amplitude tako dobljenega spektra so koeficienti melfrekvenčnega kepstruma.

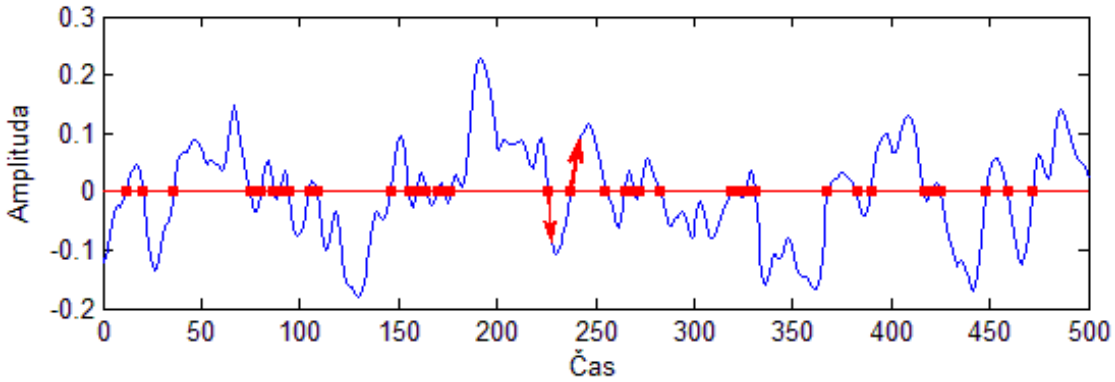
Obstajajo tudi variacije pridobivanja koeficientov MFC (npr. različne oblike in dolžine oken pri preslikavi v lestvico mel). Koeficienti MFC niso pretirano robustni, kadar je prisoten dodaten šum, zato se po navadi njihove vrednosti normalizira in s tem zmanjša vpliv šuma.

Ker MFC-koeficienti ne vsebujejo časovnih informacij, sta pomembna še njihova prvi in drugi odvod, ki ju po navadi imenujemo ΔMFCC in ΔΔMFCC. Izračunamo jih z uporabo enačbe za linearno regresijo

$$\Delta c[m] = \frac{\sum_{i=1}^{k} i\,(c[m+i] - c[m-i])}{2\sum_{i=1}^{k} i^2} \qquad (4)$$

kjer je *2k+1* velikost regresijskega okna in je *c[m]* m-ti MFC-koeficient.

2.6. Hitrost prehajanja skozi ničlo



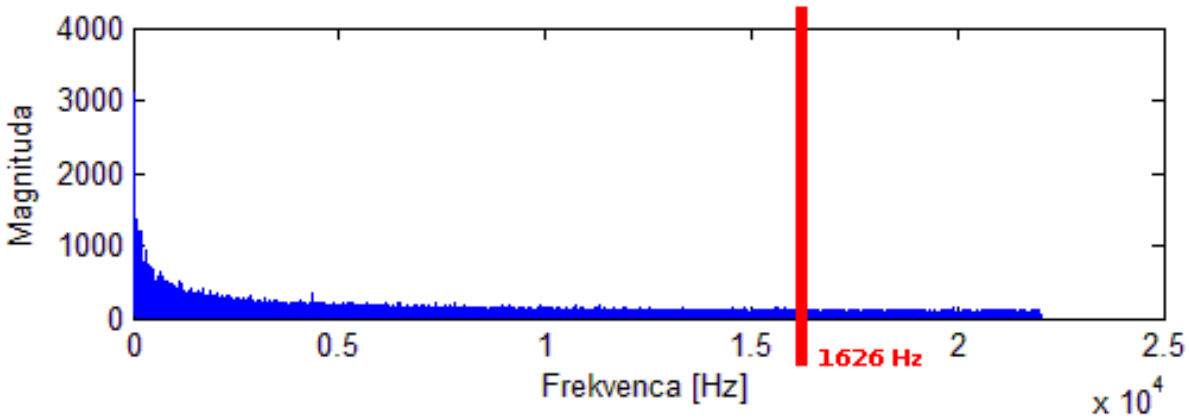
Slika 4: S pikami so prikazani prehodi skozi ničlo, ki je označena z vodoravno črto. S puščicama sta prikazana primera prehodov za vsako od obeh možnih front prehoda skozi ničlo.

Hitrost prehajanja skozi ničlo je značilka, ki predstavlja hitrost prehajanja zvočnega signala skozi ničlo. Pri procesiranju signalov je prehod skozi ničlo točka, kjer signal spremeni predznak. Signal z veliko šuma bo predznak zamenjal večkrat kot čistejši signal. Hitrost prehajanja skozi ničlo torej v grobem pomeni čistost signala. To značilko se veliko uporablja tako pri prepoznavi govora kot tudi pri MIR. Pri monofonskih tonskih signalih jo lahko uporabimo tudi kot primitiven algoritem za ugotavljanje višine tona. Upošteva se lahko vse prehode ali pa prehode samo na določeni fronti (prehod amplitude iz pozitivne na negativno vrednost ali obratno). Hitrost prehajanja skozi ničlo je definirana z naslednjo enačbo:

$$zcr = \frac{1}{T-1}\sum_{t=1}^{T-1} I\{s_t s_{t-1} < 0\} \qquad (5)$$

kjer je s dolžina signala T in je funkcija $I\{A\}$ enaka 1, kadar je pogoj A resničen, sicer je enaka 0.

2.7. Spektralni upad



Slika 5: Prikazan je spekter moči in na njem s črto označena upadna frekvenca spektra pri 1626 Hz.

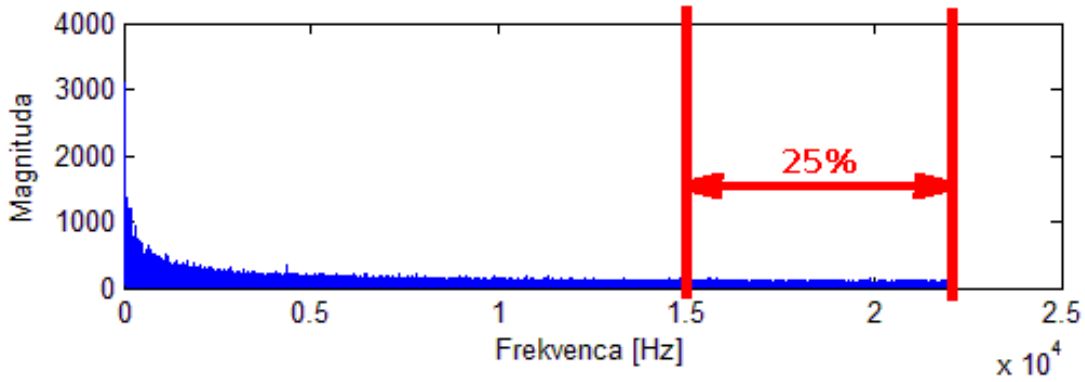
Spektralni upad (angl. Spectral Roll-off) je en od načinov za oceno količine energije pri visokih frekvencah signala. Določimo ga tako, da poiščemo frekvenco, pod katero se nahaja izbran odstotek vse energije signala. Ta odstotek se med raziskovalci razlikuje, večinoma pa se uporablja meje pri 85 % ali 95 % energije. To je eno izmed meril oblike spektra. Spektralni upad je povezan z mejno frekvenco med šumom in harmoničnim delom spektra. Določen je z enačbo

$$\sum_{0}^{f_c} a^2(f) = 0{,}85 \sum_{0}^{\frac{sr}{2}} a^2(f) \qquad (6)$$

kjer je f_c upadna frekvenca spektra in *sr/2* Nyquistova frekvenca.

Nyquistova frekvenca je polovica frekvence vzorčenja digitalnega signala in obenem minimalna frekvenca vzorčenja, pri kateri signal še lahko popolnoma rekonstruiramo. Če je frekvenca vzorčenja manj kot dvakrat višja od najvišje frekvence, v analognem signalu pride do prekrivanja (angleško aliasing). Takrat analognega signala iz digitalnega zapisa ni več mogoče popolnoma rekonstruirati brez popačenja.

2.8. Svetlost zvoka

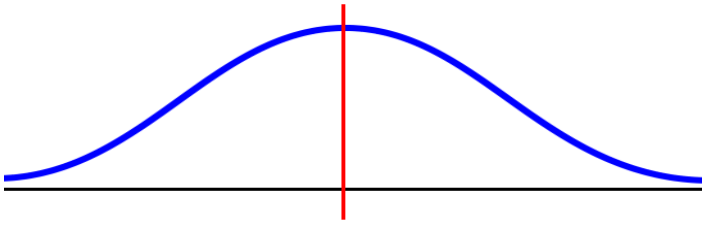


Slika 6: Na spektru moči označen del med 1500 Hz in koncem spektra. Na tem delu se v prikazanem primeru nahaja 25 % celotne moči.

Pri tej značilki računamo odstotek energije, ki je prisoten v signalu nad podano mejno frekvenco. V našem primeru je meja postavljena na 1500 Hz, zanima pa nas odstotek energije, ki je v signalu nad to frekvenco.

Za raziskovalce, ki se ukvarjajo z barvo zvoka, je svetlost zvoka ena izmed najmočnejših značilnosti s stališča zaznave, ki prispevajo k razlikovanju med zvoki. Predstavlja količino vsebnosti visokih frekvenc v zvoku. Več visokih frekvenc je v zvoku v primerjavi s srednjimi in nizkimi frekvencami, svetlejši je zvok.

2.9. Spektralni centroid



Slika 7: Z navpično črto je označen centroid frekvenčne porazdelitve spektra.

Spektralni centroid je vrednost, uporabljena pri digitalnem procesiranju signalov za karakterizacijo spektra. Pokaže namreč, kje je njegovo geometrično središče. S stališča poslušalčeve zaznave ima močno povezavo z vtisom svetlosti zvoka. Izračunan je kot utežena srednja vrednost frekvenc v signalu, pridobljenih z uporabo Fourierjeve transformacije. Magnitude frekvenc so uporabljene kot uteži pri izračunu spektralnega centroida.

Enačba je naslednja:

$$centroid = \frac{\sum_{n=0}^{N-1} f(n)x(n)}{\sum_{n=0}^{N-1} x(n)} \qquad (7)$$

kjer *x(n)* predstavlja uteženo magnitudo frekvence n-tega pasu Fourierjeve transformacije in *f(n)* predstavlja frekvenco tega pasu.

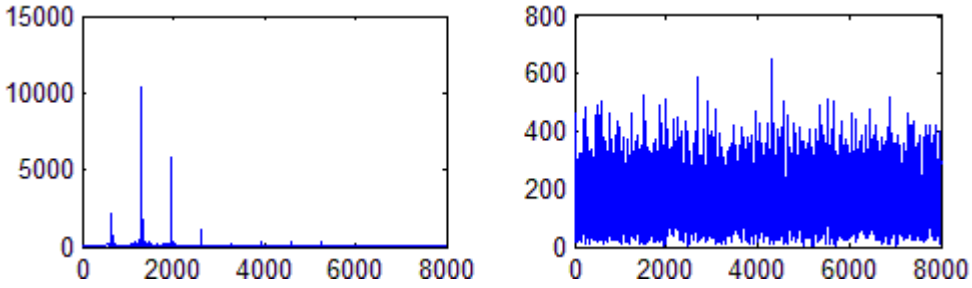
2.10. Nesimetričnost spektra

Vrne oceno nesimetričnosti spektra oziroma stopnjo spreminjanja med zaporednimi vrhovi spektra. V našem primeru se uporablja Jensenova metoda [5]. Po tej metodi je nesimetričnost spektra definirana kot vsota kvadratov razlike amplitud soležnih sotonov, amplituda N+1 sotona pa naj bi bila 0. Enačba se tako glasi

$$irregularity = \frac{\sum_{k=1}^{N} (a_k - a_{k+1})^2}{\sum_{k=1}^{N} a_k^2} \qquad (8)$$

Ob spremembi nesimetričnosti spektra se spremeni tudi barva zvoka. V večini primerov je njena vrednost manjša od 1, po definiciji pa je vedno manjša od 2.

2.11. Spektralna ravnost



Slika 8: Leva slika prikazuje spekter frekvenc pri 110 Hz tonu A, zaigranem na kitari. Spektralna ravnost za ta posnetek je 0,022. Desna slika prikazuje spekter frekvenc belega šuma. Spektralna ravnost tega posnetka je 0,845.

Značilka predstavlja spektralno ravnost ali koeficient tonalnosti, ki je tako kot spektralni centroid vrednost uporabljena pri digitalnem procesiranju signalov za karakterizacijo zvočnega spektra. Merimo jo v decibelih. Z njo izmerimo, ali je zvok bolj podoben tonu ali šumu. Tonskost je v tem kontekstu mišljena v smislu števila vrhov v spektru

moči signala. Nasprotno od tona je šum v tem spektru namreč raven.

Visoka spektralna ravnost (vrednosti blizu 1) kaže na to, da ima spekter po vseh pasovih podobno moč, kar se sliši kot šum. Nizka spektralna ravnost (vrednosti blizu 0) pa kaže na to, da je moč zbrana v relativno majhnem številu pasov spektra, kar se tipično sliši kot mešanica sinusoid. V tem primeru se spekter kaže kot bolj razgiban.

Spektralno ravnost definira enačba

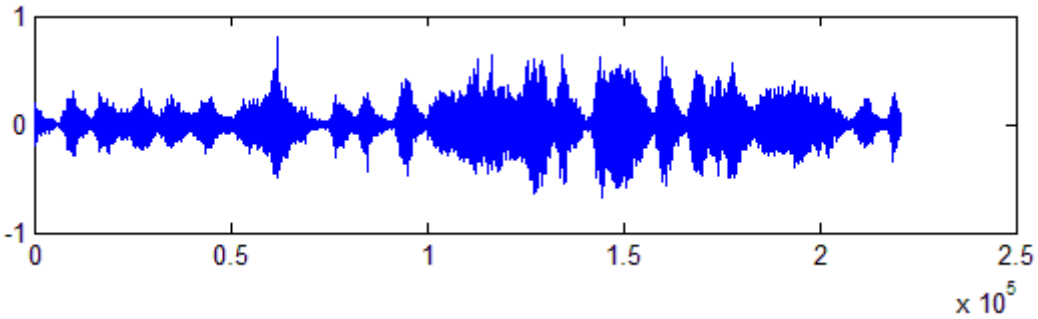
$$flatness = \frac{\sqrt[n]{\prod_{n=0}^{N-1} x(n)}}{\frac{\sum_{n=0}^{N-1} x(n)}{N}} \qquad (9)$$

kjer *x(n)* predstavlja magnitudo n-tega pasu Fourierjeve transformacije.

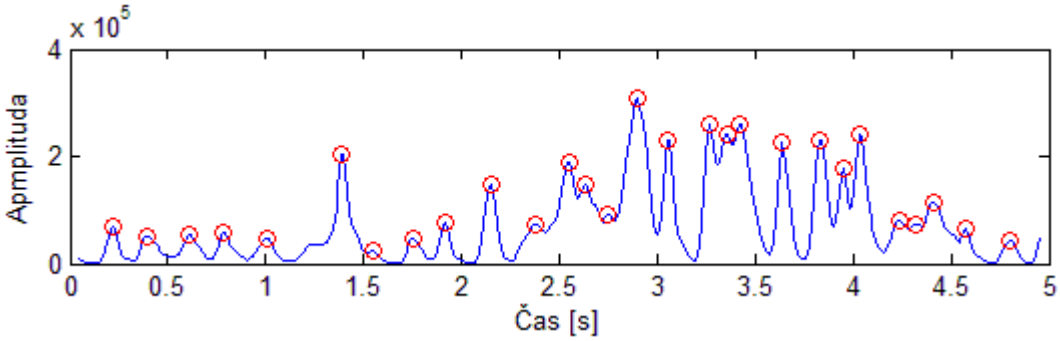
2.12. Frekvenca pojavljanja začetkov not

Pri tej značilki nas zanima ocena povprečne frekvence pojavljanja začetkov not v posnetku. Angleški izraz za to je onset, ki pomeni začetek. Na področju zvoka gre za začetek note ali zvoka, kjer amplituda signala iz ničle doseže vrh. Detekcija začetkov not je na področju MIR zelo dejavno področje.

Pristopi k problemu so različni, raziskovalci pa se jih lotevajo z iskanjem rasti energije v delu spektra, sprememb v porazdeljenosti spektralne energije, sprememb v zaznani višini zvoka ob uporabi algoritmov za zaznavo višine zvoka v polifonskih signalih in z uporabo strojnega učenja za prepoznavo vzorcev v spektru. Uporaba preprostejših metod, kot npr. zaznavanje višanja amplitude v času, večinoma daje zelo nezanesljive rezultate.



Slika 9: Prikaz signala petsekundnega posnetka solo harmonike.



Slika 10: Za posnetek iz slike 9 prikazana krivulja začetkov not z označenimi vrhovi, ki predstavljajo začetke not. Frekvenca pojavljanja začetkov not je bila za ta primer izračunana kot 2,2 note na sekundo.

2.13. Tempo

Tempo v glasbi pomeni hitrost glasbenega dela in je en izmed ključnih elementov kompozicije. Določa število zaigranih not v minuti, od njega pa je odvisno tudi vzdušje v glasbi.

V zvočnih posnetkih se tempo lahko med drugim oceni z zaznavanjem periodičnosti začetkov not (oziroma pri prej opisanih onsetih). Ta način je uporabljen tudi v tem diplomskem delu. Najprej se določi krivulja pojavljanja not (slika 10). Nad to krivuljo se nato izračuna avtokorelacija. Avtokorelacija je matematično orodje za iskanje ponavljajočih vzorcev, kot je npr. ponavljajoč signal, skrit med šumom. V našem primeru se avtokorelacija izračuna nad krivuljo pojavljanja začetkov not. Rezultat bi tako moral biti groba ocena tempa, saj bi se v glasbi note načeloma morale ponavljati ob taktu. Na posnetku sliki 9 je tempo, izračunan po tej metodi, 156 udarcev na minuto, kar je tudi dejanski tempo tega posnetka.

2.14. LMT

LMT ali Logistic Model Tree [6] je algoritem za nadzorovano strojno učenje. Vsebuje logistično regresijo in drevesno indukcijo. To sta dve priljubljeni metodi za klasifikacijo, ki imata ravno obratne prednosti in slabosti. Prva za opis podatkov uporablja preprost linearni model. Proces prilagajanja modela je dokaj stabilen, kar prinaša nizko varianco, a je lahko precej pristranski. Druga metoda pa ravno obratno prinaša po navadi visoko varianco, a ni tako pristranska. Preiskuje manj omejen modelski prostor, kar ji omogoča zajem nelinearnih vzorcev, vendar zaradi tega ni tako stabilna. V splošnem nobena od obeh metod ni boljša od druge, saj je učinkovitost obeh odvisna od velikosti in karakteristik nabora podatkov.

Obe metodi bi seveda lahko združili in tako dobili metodo, ki bi uporabila preprostejše regresijske modele, kadar bi bilo na voljo malo podatkov ali ko bi bilo v podatkih veliko šuma. Kompleksnejše drevesne strukture pa bi uporabili, kadar bi bilo na voljo dovolj podatkov. Za napovedovanje numeričnih vrednosti so na ta način izvedena modelska drevesa, ki dajejo dobre rezultate. To so odločitvena drevesa z modeli linearne regresije na listih. Načeloma bi se jih dalo prirediti tudi za klasifikacijo, vendar ta način zaradi binarizacije razredov ni najbolj uporaben. Rezultat je namreč težje predstavljiv, ker za vsakega od razredov dobimo svoje drevo.

Boljši način je uporaba kombinacije drevesne strukture in linearne logistične regresije

v enem samem drevesu. To je tudi osnova za algoritem LMT. Dodatna prednost uporabe logistične regresije je, da namesto samo klasifikacije vrne eksplicitne ocene verjetnosti za razrede.

Osnovna ideja algoritma za izgradnjo LMT je naslednja:

- Gradnja drevesa se začne z gradnjo logističnega modela v korenu z uporabo algoritma LogitBoost. Število iteracij se določi z uporabo prečnega preverjanja s petimi pregibi. V tem postopku se podatke petkrat loči na učno in testno množico, vsakič se nad testno množico uporabi algoritem LogitBoost z največjim številom iteracij. Pojavitve napak prek vseh iteracij po pregibih se seštevajo. Število iteracij, kjer je seštevek pojavitev napak najmanjši, se uporabi pri algoritmu LogitBoost za vse podatke. S tem se dobi logistični regresijski model v korenu drevesa.
- Razdelitev podatkov v korenu se izvede z uporabo C4.5-delitvenega kriterija. Upošteva se tako binarne delitve številskih atributov kot tudi večsmerne delitve za nominalne atribute. Gradnja drevesa se nadaljuje s sortiranjem podmnožic podatkov po vozliščih in grajenjem logističnih modelov sinov vozlišč z uporabo algoritma LogitBoost na podatkih, ki pripadajo sinu. Tudi tukaj se optimalno število iteracij za LogitBoost določi s prečnim preverjanjem s 5 pregibi.
- Delitev podatkov se nadaljuje po tem postopku, dokler je na vozlišču več kot 15 primerkov in je hkrati možna uporabna delitev po C4.5-delitveni metodi.
- Izvede se rezanje drevesa z uporabo CART-algoritma za rezanje.

Pri manjkajočih vrednostih se izračuna aritmetična sredina za številske atribute oziroma modus za nominalne atribute za vse učne podatke. Tako izračunane vrednosti se nato uporabijo za zamenjavo manjkajočih vrednosti. Enake vrednosti za aritmetično sredino in modus se uporabijo za zamenjavo manjkajočih vrednosti tudi pri klasifikaciji novih primerkov.

Časovna zahtevnost gradnje LMT je $O(n*v^2*d*c+k^2)$, kjer je n število učnih primerov, v število atributov, c število razredov, d višina drevesa in k število vozlišč v začetnem nerezanem drevesu. V zgoraj opisanem algoritmu se optimalno število iteracij določa za vsako od vozlišč posebej. To je računsko najbolj potraten del algoritma. Za zmanjšanje časa gradnje drevesa so avtorji uporabili dve hevristiki, ki sta vključeni v končni verziji algoritma.

- V izogib prečnemu preverjanju v vsakem vozlišču se optimalno število iteracij določi s samo eno izvedbo prečnega preverjanja na začetku algoritma. To število iteracij se nato uporablja na celotnem drevesu. Ta pristop nikoli ni vrnil rezultatov, ki bi bili opazno slabši kot v originalnem algoritmu.
- Ob izvajanju začetnega prečnega preverjanja je treba izbrati število iteracij z najmanjšo napako na testni množici. Večinoma bo napaka na začetku vedno manjša, nato pa se bo začela povečevati. Zaradi tega je možno začetno število iteracij izbrati visoko, nato pa s spremljanjem napake iteracijo predčasno prekiniti, ko se napaka začne povečevati.

Razvijalci so algoritem LMT primerjali z algoritmi C4.5, SimpleLogistic, PLUS in AdaBoostM1. LMT je bil v določenih primerih slabši le od AdaBoostM1. V primerjavi z drugimi algoritmi se je LMT v nekaj primerih izkazal vsaj enako, sicer pa je bil opazno boljši.

3. Implementacija

Pri implementaciji sem v glavnem uporabil tri orodja: MATLAB, MIRToolbox (v sklopu MATLABa) in Weka. MATLAB je plačljivo orodje, MIRToolbox in Weka pa sta odprto-kodni rešitvi in kot taki izdani pod licenco GPL. Za implementacijo vmesne knjižnice, ki povezuje MATLAB in Weko, sem sicer uporabil še Eclipse, vendar ga podrobneje ne bom omenjal. Knjižnica je bila namreč tako preprosta, da bi lahko uporabil katerikoli urejevalnik besedila in prevajalnik Java.

3.1. MATLAB

MATLAB [10] je okolje za numerično matematiko in hkrati programski jezik četrte generacije. Ime izhaja iz besedne zveze "Matrix Laboratory". Omogoča delo z matrikami, izris funkcij in podatkov, implementacijo algoritmov, izdelavo uporabniških vmesnikov in povezavo s programi, pisanimi v drugih programskih jezikih. Uporablja ga več kot milijon razvijalcev.

Njegov razvoj se je začel konec 70. let prejšnjega stoletja. Cleve Moler je želel ponuditi študentom univerze v New Mexicu orodje za uporabo knjižnic LINPACK in EISPACK, ne da bi se morali učiti programski jezik Fortran. Knjižnica LINPACK omogoča izvajanje linearne algebre, EISPACK pa numerično računanje lastnih vrednosti in lastnih vektorjev. Jack Little je leta 1983 spoznal MATLAB, videl njegov tržni potencial in se nato pridružil Clevu Molerju in Steveu Bangertu. Skupaj so MATLAB na novo napisali v programskem jeziku C in leta 1984 ustanovili podjetje MathWorks za nadaljnji razvoj orodja.

Neposredno lahko kliče funkcije, napisane v programskih jezikih C in Fortran. Knjižnice, napisane v programskih jezikih Java, .NET in ActiveX, se lahko neposredno uporabijo v MATLABu. Možna je tudi uporaba MATLABa iz Jave, vendar to ni tako preprosto. Funkcije iz MATLABa je možno izvoziti tudi kot knjižnice za programska jezika Java in .NET.

3.2. MIRToolbox

MIRToolbox [11] razvijajo na University of Jyväskylä na Finskem. To je skupek funkcij za MATLAB, namenjenih pridobivanju glasbenih značilnosti iz zvočnih posnetkov (npr. tonalnost, ritem, itd.). Vsebuje vse pomembnejše MIR-algoritme. Vseh funkcij je prek 50 in jih tukaj ne bom opisoval. Podrobneje pa so opisani algoritmi, ki so bili uporabljeni prek MIRToolboxovih funkcij. Glavni cilj je uporabniku ponuditi čim širši spekter računskih pristopov k problemom s področja MIR. Zasnovan je modularno. Algoritmi so razbiti na stopnje, ki predstavljajo osnovne elemente. Ti osnovni elementi vsebujejo več možnih rešitev, tako s strani razvijalcev MIRToolboxa kot tudi alternativne pristope. Uporabnik ima tako možnost izbrati najprimernejši pristop, hkrati pa ima vsakič ponujeno tudi možnost nastavljanja raznih parametrov. Taka zasnova nadalje omogoča uporabniku poljubno kombiniranje vseh elementarnih elementov, kar mu omogoča preprosto implementacijo rešitve problema.

3.3. Weka

Weka (Waikato Environment for Knowledge Analysis) [12] je med najpopularnejšimi rešitvami za strojno učenje. Razvija jo University of Waikato z Nove Zelandije in je odprto-kodna ter je izdana pod licenco GPL.

Je zbirka vizualizacijskih orodij in algoritmov za podatkovno analizo in grafičnih uporabniških vmesnikov za lažjo uporabo teh orodij. Weka je bila najprej samo maska za (večinoma) modelirne algoritme drugih razvijalcev, ki so bili pisani v različnih programskih jezikih, orodja za predobdelavo podatkov v programskem jeziku C in sistem, zasnovan na Makefile za poganjanje učnih poskusov. V tej obliki je obstajala predvsem za analizo podatkov s kmetijskih področij. Od leta 1997 naprej je Weka (različica 3) v celoti razvita v Javi. Zelo razširila se je tudi njena uporabnost, najbolj pa se jo uporablja v izobraževalne in raziskovalne namene. Podprte metode za podatkovno rudarjenje so predobdelava podatkov, grupiranje, klasifikacija, regresija in vizualizacija.

Vse metode pričakujejo podatke v eni sami datoteki. Primarno je to datoteka tipa ARFF (Attribute-Relation File Format), je pa možen uvoz podatkov iz drugih tipov (npr. CSV, MATLABove datoteke z rezultati in podobno). ARFF-datoteka je razdeljena na dva dela. V prvem delu se nahaja ime relacije in seznam atributov s pripadajočimi tipi, drugi del pa je podatkovni del. Vsak podatek je podan v svoji vrstici in ga sestavljajo prej našteti atributi, katerih število je nespremenljivo. Večinoma se uporablja številske in imenske atribute, podprta pa sta trenutno še tekstovni in datumski tip atributov.

Primer ARFF-datoteke, s pomočjo katere se odločamo o ukvarjanju s športom glede na vremensko napoved:

```
@relation vreme

@attribute napoved {sončno, oblačno, deževno}
@attribute temperatura real
@attribute vlažnost real
@attribute veter {da, ne}
@attribute šport {da, ne}

@data
sončno,29,85,ne,ne
sončno,27,90,da,ne
oblačno,28,86,ne,da
deževno,21,96,ne,da
deževno,20,80,ne,da
deževno,18,70,da,ne
oblačno,17,65,da,da
```

3.4. Implementacija

Osnovna ideja rešitve je dokaj enostavna. Za posnetek, ki ga podamo programu, nas zanima njegov glasbeni sestav, kar je tudi rezultat programa. To se zgodi v dveh delih. Najprej se z uporabo MIR-algoritmov iz posnetka pridobi zvočne značilke. Te značilke se nato posreduje algoritmu za strojno učenje, ki glede nanje posnetek klasificira v enega od razredov.

Razredi so v našem primeru vnaprej določeni glasbeni sestavi.

Implementacija rešitve zaradi dobre izbire orodij ni bila zahtevna. Kot glavno orodje sem uporabil MATLAB, kar se je izkazala kot odlična izbira, saj zanj med drugim obstaja tudi dodatek MIRToolbox. V MIRToolboxu so implementirani že vsi najpomembnejši MIR-algoritmi in se tako meni z njihovo implementacijo ni bilo treba ukvarjati. Zadnjih nekaj različic MATLAB-a podpira tudi neposredno izvajanje kode programskega jezika Java. To mi je omogočilo združitev pridobivanja značilk iz posnetkov prek MIRToolboxa in izvajanja strojnega učenja z Weko z enega mesta, brez nepotrebnih vmesnih korakov.

Vhod v program je seznam posnetkov. Za vsakega od posnetkov se nato izvedejo naslednji štirje koraki:

1. **Razdelitev na segmente**: vsak posnetek se najprej razdeli na desetsekundne segmente. Posnetki so namreč večinoma dolgi od nekaj minut do nekaj ur, obenem pa lahko zajemajo tudi več različnih sestavov. Desetsekundni segmenti so bili določeni na podlagi testiranj. Ta so pokazala, da se sestavi na testnih posnetkih v večini primerov spremenijo v desetih ali več sekundah.
2. **Pasovno prepustni filter**: vsak segment se z uporabo pasovno prepustnega filtra razdeli na tri frekvenčne pasove: 0 do 100 Hz, 100 do 1000 Hz in 1000 Hz do polovice frekvence vzorčenja posnetka.
3. **Izračun zvočnih značilk**: za vsak segment se s pomočjo MIRToolboxa izračuna 50 vrednosti značilk. 30 MFC-koeficientov, tempo in frekvenca pojavljanja not niso odvisni od prej uporabljenega filtra in se izračunajo nad celotnim segmentom. Za vsakega od treh frekvenčnih pasov pa se izračuna hitrost prehajanja skozi ničlo, spektralni upad, svetlost zvoka, nesimetričnost spektra, spektralni centroid in spektralna ravnost. Vsaka od teh šestih značilk je tako predstavljena s tremi vrednostmi.
4. **Klasifikacija**: za klasifikacijo sem izbral v Weki implementiran algoritem za strojno učenje LMT. Vhod v ta algoritem je vektor prej izračunanih značilk, na podlagi katerih se nato izvede klasifikacija segmenta v razrede. Pri klasifikaciji se uporabijo vnaprej pripravljeni učni podatki.

Tako dobljeni razredi se za vsakega od segmentov posnetkov izpišejo v konzolo, kar je tudi končni rezultat programa.

Glavni program je napisan v MATLAB-u in ga sestavljajo tri funkcije. Pasovno prepustni filter in izračun značilk sta združena v funkciji na najnižjem nivoju, ki jo kliče funkcija za segmentacijo posnetkov. Na najvišjem nivoju je funkcija za klasifikacijo segmentov. Ta iz prejšnjih dveh funkcij dobi matriko vrednosti značilk (ki vsebuje po 50 vrednosti za vsakega od segmentov) in jo poda algoritmu LMT.

Tu pa se je pokazal manjši problem. Algoritem LMT je implementiran v sklopu orodja Weka, ki je dostopno v obliki Java programske knjižnice. MATLAB za klice metod programskega jezika Java uporablja prilagojeno sintakso, zaradi česar ni bil mogoč klic ene izmed metod Weke, potrebne za izvedbo klasifikacije segmentov. Zaenkrat pri klicu metod Java iz MATLAB-a ni možna uporaba praznih opcijskih parametrov. V ta namen sem izdelal vmesno knjižico Java, ki je to omejitev zaobšla.

4. Rezultati

4.1. Učni podatki

Izvor vseh podatkov v sklopu tega diplomskega dela so zvočni posnetki. Zaradi uporabe strojnega učenja je treba pridobiti dve množici posnetkov oziroma podatkov, in sicer učno in testno. S podatki iz učne množice se, kot pove že ime, uči klasifikacijski model. Tako naučen model nato uporabimo za klasifikacijo testnih podatkov.

Obstaja več tipov strojnega učenja. V diplomskem delu sem uporabil algoritem LMT (opisan v drugem poglavju), ki spada med algoritme nadzorovanega strojnega učenja. Nadzorovano učenje pomeni, da se vsakemu od vhodnih podatkov iz učne množice določi tudi želena izhodna vrednost. V našem primeru to pomeni, da za vsak posnetek vnaprej povemo, v katero kategorijo spada. Iz tako podanih podatkov algoritem nato zgradi klasifikacijski model, ki naj bi (vsaj v idealnih okoliščinah) za vsak veljaven vhodni podatek pravilno napovedal, v katero od kategorij spada.

Iz zgoraj napisanega lahko torej sklepamo, da priprava učne množice ni povsem trivialna naloga. Zagotoviti je treba dovolj podatkov za vsako od kategorij. Poleg zadostne količine pa je treba izbrati tudi prave podatke, torej take, iz katerih bo algoritem lahko zgradil zadovoljiv klasifikacijski model. Vsaka kategorija mora biti čim bolj pokrita z učnimi podatki. S tem je mišljeno, da mora biti v določeni kategoriji zajetih čim več različnih možnih pojavitev primerov podatkov.

V grobem bi se lahko reklo, da je pri pripravi učne množice treba upoštevati tudi poznejše testne podatke. Na primeru zvoka bi to pomenilo, da mora biti recimo v kategoriji trobente zajetih čim več različnih tonov in stilov igranja trobente. Če bi bila celotna učna množica za trobento sestavljena samo iz enega ali dveh tonov, bi to zelo verjetno pomenilo, da bo iz teh podatkov zgrajen slab klasifikacijski model. Ko bi temu modelu nato dali v klasifikacijo testne podatke, ki bi vsebovali posnetke trobente z drugimi toni, bi nam vrnil precej nepredvidljive rezultate. Prav tako je treba upoštevati morebiten šum. Če je v testnih podatkih predviden šum, je včasih dobro, da ga zajamemo tudi v učnih podatkih.

Pri pripravi učnih podatkov za svoj program sem imel na voljo večinoma samo terenske posnetke raznih dogodkov. Posnetki so bili dolgi tudi uro in pol ali več, njihova vsebina pa je bila zelo mešana (glasba, intervjuji, petje, itd.). To je pomenilo veliko ročnega urejanja. Vse posnetke je bilo treba poslušati in iz njih izrezati uporabne dele. Iz tako urejenih posnetkov so se nato lahko določile kategorije, ki bi me zanimale pri klasifikaciji. Že med razrezom posnetkov se zapisal, kateri od glasbil so prisotni na vsakem izseku oziroma kateri regiji pripadajo. Če je posnetek pripadal regiji, se je kot kategorija določila regija, sicer pa je bila kategorija posamezno glasbilo ali skupina le-teh. Izbor se je nato še zožil glede na število posnetkov v posamezni kategoriji. Če je bilo v kategoriji manj kot 10 posnetkov, se je kategorija odstranila vključno s posnetki v njej.

Kategorija	Št. posnetkov v kategoriji
Rezija - petje	286
Rezija - glasba	267
Harmonika - solo	196
Prekmurje	60
Harmonika s petjem	30
Bela krajina - glasba	28
Bela krajina - glasba s petjem	20
Harmonika in violina	19
Harmonika in tuba	18
Harmonika in bas	14

Tabela 1: Kategorije, ki so bile izbrane kot možne za uporabo.

Učni posnetki se nadalje uvrščajo še v tri podmnožice, ki se razlikujejo v dolžini posnetka. Iz vsakega od posnetkov so bili izrezani trije odseki različne dolžine. Če je bil posnetek slučajno krajši od predvidenega izseka, je iz 5- in 15-sekundne podmnožice izpadel, v 10-sekundni podmnožici pa so lahko tudi krajši posnetki. V prvi skupini so bili izseki med 5. in 10. sekundo posnetka, torej dolgi 5 sekund. V drugi skupini so bili izseki od začetka posnetka do 10. sekunde, v tretji pa med začetkom in 15. sekundo posnetka.

4.2. Testiranje in rezultati

Testnih podatkov v nasprotju z učnimi ni treba posebej pripravljati. Pravzaprav je tudi mišljeno, da program prejme neobdelan terenski posnetek kot vhod in vrne kategorije za vsak njegov odsek. Vseeno pa sem za boljši test obdelane posnetke vsake kategorije ločil na učno in testno množico v razmerju 2:1.

Uspešnost klasifikacije je bila preverjana na tri načine: najprej s prečnim preverjanjem s pregibanjem, nato z obdelanimi testnimi podatki, ki sem jih ločil od učne množice, in nazadnje še s terenskimi posnetki. Pri vsakem od treh načinov preverjanja so bile uporabljene tudi različne kombinacije učnih podatkov in nastavitev programa.

Kategorije iz tabele 1 so bile nato uporabljene v dveh testnih skupinah. Kategorije iz prve testne skupine so bile v grobem izbrane že med zbiranjem posnetkov, pri drugi testni skupini pa me je zanimalo, ali je mogoče uspešno uporabiti tudi razširjen nabor kategorij.

Kategorija	Št. učnih posnetkov	Št. testnih posnetkov
Bela krajina	32	16
Harmonika – solo	132	64
Prekmurje	40 (30)	20 (12)
Rezija – glasba	176	84
Rezija – petje	166 (5s) 192 (10s) 105 (15s)	82 (5s) 94 (10s) 52 (15s)

Tabela 2: Prva testna skupina. Obe kategoriji za Belo krajino iz tabele 1 sta združeni v eno kategorijo. Število posnetkov za kategorijo Rezija – petje je podano glede na dolžino zaradi kriterija opisanega na koncu prejšnjega poglavja. Pri kategoriji Prekmurje je v oklepaju navedeno število posnetkov po izločitvi napačnih.

Kategorija	Št. učnih posnetkov	Št. testnih posnetkov
Bela krajina – glasba	18	10
Bela krajina – glasba in petje	12	8
Harmonika – solo	132	64
Harmonika in bas	10	4
Harmonika in petje	20	10
Harmonika in tuba	12	6
Harmonika in violina	13	6
Prekmurje	40	20
Rezija – glasba	176	86
Rezija – petje	166 (5s) 192 (10s) 105 (15s)	82 (5s) 94 (10s) 52 (15s)

Tabela 3: Druga testna skupina. Število posnetkov za kategorijo Rezija – petje je podano glede na dolžino, ker je nekaj posnetkov zaradi dolžine izpadlo iz 5 in 15 sekundne množice.

Uporabljen je bil tudi pasovno prepustni filter, ki je posnetek ločil na izbrane frekvenčne pasove. Vsak od frekvenčnih pasov se je nato pri računanju značilk obravnaval kot samostojen posnetek.

	Meje frekvenčnih pasov v Hz
Filter 1 (3 pasovi)	min, 100, 1000, max
Filter 2 (15 pasov)	min, 100, 200, 300, 400, 500, 600, 800, 1000, 2000, 3000, 4000, 5000, 10000, 15000, max
Filter 3 (5 pasov)	min, 100, 500, 1000, 5000, max

Tabela 4: Meje frekvenčnih pasov pasovno prepustnega filtra. Frekvence posameznega pasu določata dve sosednji vrednosti. Vrednosti min in max predstavljata minimalno in maksimalno možno frekvenco posnetka.

4.3. Prečno preverjanje s pregibanjem

Pri prečnem preverjanju s K-pregibi se učna množica razdeli naključno na K podmnožic. Ena od tako pridobljenih podmnožic se uporabi za testne podatke, drugih K-1 podmnožic pa za učne podatke. Prečno preverjanje se nato ponovi K-krat, tako da je pri vsaki ponovitvi (oziroma pregibu) vsaka od K podmnožic za testne podatke uporabljena natančno enkrat. Med bolj uporabljanimi je prečno preverjanje z 10 pregibi, tako da sem to možnost izbral tudi pri svojih testih.

Testiranje s prečnim preverjanjem sem izvajal neposredno v okolju Weka na sedmih učnih množicah. Najprej sem jih pripravil šest, ki so razdeljene v dve skupini zgoraj opisanih kategorij. V vsaki od skupin so bili posnetki dolžin 5, 10 in 15 sekund. Pri 5- in 10-sekundnih množicah posnetkov sem uporabil še pasovno prepustni filter z dvema različnima nastavitvama, in sicer v tabeli 4 opisana filter 1 in filter 3.

Množica 15-sekundnih posnetkov se je že pri testiranju brez filtra izkazala za časovno preveč zahtevno. Tudi rezultati so bili zelo podobni kot pri 10-sekundni množici, tako da sem jo iz nadaljnjega testiranja izločil. Filter 3 se je prav tako hitro izkazal za časovno preveč potratnega, hkrati pa ni dal opaznega prispevka h klasifikacijski natančnosti. Po nekaj testih sem ga zato umaknil iz nadaljnjega testiranja in ga nisem vključil med rezultate.

V teh šestih množicah učnih podatkov so zajete samo značilke, ki opisujejo barvo zvoka in splošno obliko spektra. Glede na to, da so si določeni posnetki iz različnih kategorij lahko glede na te značilke dokaj podobni, sem se odločil za test še ene učne množice. Tu sem poleg značilk, uporabljenih v drugih učnih množicah, dodal še dve značilki, ki opisujeta ritem posnetka. Pri poslušanju posnetkov sem namreč opazil, da tudi če sta posnetka iz različnih kategorij tonsko precej enaka, se velikokrat razlikujeta v ritmu.

Uspešnost klasifikacije je pri prečnem preverjanju po navadi višja kot pri testiranju na ločenih testnih podatkih. Vseeno pa so rezultati prečnega preverjanja dobra osnova za izbiro klasifikacijskega modela. Če bi bili rezultati slabi že pri prečnem preverjanju, bi bili zelo verjetno slabši tudi na dejanskih testnih podatkih. Slabi rezultati prečnega preverjanja bi namreč pomenili, da klasifikacijski model ni sposoben uspešno razlikovati niti med podatki, s katerimi je bil zgrajen.

Vendar pa se na rezultate prečnega preverjanja ne gre povsem zanašati. Iz tabele 5 je razvidno, da ima skupina 2 sicer res nekoliko manjši odstotek pravilnih klasifikacij, a ima

obenem dvakrat toliko kategorij kot skupina 1. Z upoštevanjem obojega bi bilo seveda smiselno zgubiti nekoliko klasifikacijske točnosti na račun precejšnjega povečanja števila kategorij. So pa hitri testi s testnimi podatki kmalu pokazali, da tudi če je klasifikacijska točnost pri prečnem preverjanju dokaj visoka, to ne drži tudi za točnost pri teh testih. Osnovnih pet kategorij je program še vedno prepoznaval precej uspešno, dodatnih pet pa je v približno dveh tretjinah primerov klasificiral med osnovne. Izkazalo se je, da so imele dodatne kategorije premalo učnih posnetkov. Skupino 2 sem tako iz nadaljnjih testiranj izločil.

	Brez filtra	**Filter 1**	**Filter 3**
Skupina 1 – 5s	85%	88%	89%
Skupina 1 – 10s	87%	90%	91%
Skupina 1 – 15s	88%	/	/
Skupina 1 z značilkama za ritem – 10s	88% (90%)	***92% (93%)***	93% (94%)
Skupina 2 – 5s	78%	84%	83%
Skupina 2 – 10s	83%	87%	88%
Skupina 2 – 15s	82%	/	/

Tabela 5: Odstotek pravilnih klasifikacij prečnega preverjanja z 10 pregibi na različnih kombinacijah posnetkov in filtrov. Odebeljeno je označen odstotek pri kombinaciji, ki sem jo izbral za nadaljnja testiranja. V oklepajih je podan odstotek pravilnih klasifikacij po odpravi napake v učnih podatkih.

Iz tabele 5 je glede na odstotke pravilno klasificiranih primerov prečnega preverjanja videti, da je uporabljen klasifikacijski model precej uspešen. Vendar so ti rezultati, kot sem že omenil, precej optimistični, kar bo razvidno tudi pri rezultatih preostalih dveh testov.

Rezultate prečnega preverjanja sem tako uporabil kot osnovo pri izbiri najboljše učne množice za nadaljnja testiranja. Pri podatkih v tabeli 5 je najbolj opazen preskok med dolžinama posnetkov 5 in 10 sekund ter med uporabo ali neuporabo filtra. Učne množice s 15-sekundnimi posnetki so bile izločene že kmalu po začetku testiranja. Glede na opazen preskok v uspešnosti klasifikacij sem tako izločil še vse učne množice s 5-sekundnimi posnetki.

Pri preostalih učnih množicah z 10-sekundnimi posnetki sem nato preverjal še uspešnost klasifikacije glede na filter. Tudi množice brez uporabe filtra imajo opazno slabše rezultate in so bile izločene. Glede na uspešnost klasifikacije so po rezultatih prečnega preverjanja boljše učne množice, pri katerih je bil uporabljen filter 3. Po drugi strani pa razlika v natančnosti ni velika, tako da sem upošteval tudi porabljen čas izračuna značilk, ki je pri uporabi filtra 3 opazno večji.

Z upoštevanjem točnosti klasifikacije kot tudi časa izračuna značilk sem se odločil za uporabo učne množice z 10-sekundnimi posnetki, uporabo filtra 1 in dodanima značilkama za ritem.

	BelaKrajina	Harmonika	Prekmurje	RezijaGlasba	RezijaPetje
BelaKrajina	**25**	4	1	1	1
Harmonika	1	**119**	2	3	7
Prekmurje	1	2	**24**	2	1
RezijaGlasba	0	4	2	**170**	0
RezijaPetje	0	8	0	0	**184**

Tabela 6: Klasifikacijska matrika napak za izbrano učno množico. Vrstice predstavljajo dejanske razrede, stolpci pa klasificirane razrede. Številke predstavljajo koliko posnetkov je klasifikacijski model klasificiral v katerega od dejanskih razredov. Označena diagonala predstavlja pravilno klasificirane posnetke.

4.4. Test z obdelanimi testnimi posnetki

Drugi del testiranj je bil izveden s tretjino posnetkov, ki so bili vzeti iz učne množice. Ti posnetki izhajajo iz istega sklopa kot posnetki iz učne množice in so vmesna stopnja med učnimi in terenskimi posnetki. Razdeljeni so že na posamezne kategorije, od učnih posnetkov pa so bili ločeni še pred krajšanjem le-teh na 5-, 10- in 15-sekundne izseke.

V testni množici je bilo 272 posnetkov iz vseh petih kategorij. Dolžina posnetkov je bila med 7 sekundami in 33 minutami, v povprečju pa so bili dolgi približno minuto in pol. Pri testiranju je bila uporabljena učna množica, ki je bila izbrana v prejšnjem poglavju. Posnetki so bili razdeljeni na 10-sekundne segmente.

Klasifikacijska točnost je v nadaljevanju določena z enačbo

$$točnost = \frac{št.\ pravilno\ klasificiranih\ posnetkov}{št.\ vseh\ testnih\ posnetkov} * 100\,\% \qquad (10)$$

Za vsakega od testnih posnetkov v tem delu testiranj sem vnaprej vedel, kateri kategoriji pripada. Najprej sem tako za vsakega od posnetkov neposredno primerjal znano kategorijo s kategorijo, v katero je posnetek klasificiral program. Rezultati tega testa so bili pričakovano nižji od rezultatov prečnega preverjanja.

Pri prvem testu je program na ta način dosegel 67% točnost klasifikacije. Rezultate sem nato natančneje preveril še s ponovnim poslušanjem testnih posnetkov. Pri tem sem ugotovil, da je v nekaterih posnetkih, uvrščenih v kategorijo rezijske glasbe, zajeto tudi njihovo petje, kar je program večinoma pravilno prepoznal. Prav tako je bilo v posnetkih Bele krajine zajeto petje, ki pa ni bilo vključeno med učne podatke. Tudi to je program večinoma prepoznal kot rezijsko petje, občasno pa tudi kot harmoniko. Ko sem rezijsko petje upošteval kot pravilno, petje Bele krajine pa sem izločil iz rezultatov, se je točnost dvignila na 69 %.

Obenem sem med tem natančnejšim preverjanjem odkril napako v učnih podatkih. Med rezultati so bili namreč določeni deli kategorije Prekmurje klasificirani kot Rezija – glasba in obratno. Po kontroli učnih podatkov sem med učnimi posnetki Prekmurja našel tudi posnetke rezijske glasbe. Iz učne množice za Prekmurje je tako izpadlo 10 posnetkov. Po popravku učne množice sem najprej ponovil prečno preverjanje in pri tem dobil boljše rezultate, kar je vidno v tabeli 5. S popravljeno učno množico sem ponovil še drugi test in tudi tu so bili rezultati precej boljši. Točnost klasifikacije se je dvignila na 83 %.

Točnost klasifikacije sem nato preverjal še za posamezne kategorije. Tu so bili upoštevani samo deli posnetkov, ki so dejansko spadali v njihovo kategorijo. Posnetki Bele krajine so vključevali tudi dele s harmoniko ali petjem. Teh delov nisem upošteval pri izračunu točnosti.

	Točnost klasifikacije	**% učnih posnetkov glede na vse kategorije**
Bela krajina	70%	6%
Harmonika	91%	24%
Prekmurje	20%	5%
Rezija – glasba	88%	31%
Rezija – petje	82%	34%

Tabela 7: Točnost klasifikacije za vsako od kategorij in odstotek učnih posnetkov glede na ostale kategorije.

Med rezultati zelo izstopa kategorija Prekmurje. Slabe rezultate za to kategorijo sem najprej pripisal premajhnemu številu učnih posnetkov, vendar se to ne sklada z dokaj visoko točnostjo Bele krajine, ki ima prav tako nizek odstotek vseh učnih posnetkov. Izvedel sem tudi ponovno prečno preverjanje učnih posnetkov med paroma kategorij Prekmurje in Bela krajina ter Prekmurje in Harmonika. Prekmurje je bilo največkrat napačno klasificirano kot ena izmed teh dveh kategorij. V obeh primerih je bilo pravilno klasificiranih okrog 95 % posnetkov. Rezultat prečnega preverjanja kaže na to, da so učni podatki verjetno dobri. Po primerjanju učnih in testnih posnetkov sem ugotovil, da sem posnetke slabo razdelil v obe množici. Vsi posnetki sicer spadajo v kategorijo Prekmurje, vendar je način igranja različen. Zaradi majhnega števila posnetkov v obeh množicah pa je slaba delitev zelo opazna.

	BelaKrajina	**Harmonika**	**Prekmurje**	**RezijaGlasba**	**RezijaPetje**
BelaKrajina	**180**	32	18	59	56
Harmonika	43	**593**	5	4	7
Prekmurje	47	44	**26**	5	7
RezijaGlasba	49	42	18	**1028**	39
RezijaPetje	40	24	7	1	**320**

Tabela 8: Klasifikacijska matrika napak za testno množico.

4.5. Test z obdelanimi testnimi posnetki brez pasovno prepustnega filtra

Za primerjavo razlik v klasifikacijski točnosti sem izvedel še test brez uporabe pasovno prepustnega filtra. Uporabljeni so bili isti testni posnetki in izračunane iste zvočne značilke kot pri testu iz prejšnjega poglavja. Kot se je pokazalo že pri prečnem preverjanju s pregibanjem, so rezultati slabši tudi pri tem testu. Algoritem je dosegel 73% točnost

klasifikacije, torej kar 10 % slabšo kot z uporabo pasovno prepustnega filtra pri prejšnjem testu. Opazno so se zmanjšale tudi točnosti klasifikacije po posameznih kategorijah.

Iz primerjave rezultatov obeh testov je razvidno, da uporaba pasovno prepustnega filtra klasifikacijsko točnost opazno poveča.

	Točnost klasifikacije	**% učnih posnetkov glede na vse kategorije**
Bela krajina	57%	6%
Harmonika	83%	24%
Prekmurje	13%	5%
Rezija – glasba	84%	31%
Rezija – petje	58%	34%

Tabela 9: Točnost klasifikacije brez uporabe pasovno prepustnega filtra za vsako od kategorij in odstotek učnih posnetkov glede na ostale kategorije.

	BelaKrajina	**Harmonika**	**Prekmurje**	**RezijaGlasba**	**RezijaPetje**
BelaKrajina	**150**	48	6	75	66
Harmonika	57	**539**	15	20	21
Prekmurje	41	35	**17**	34	2
RezijaGlasba	107	48	8	**986**	27
RezijaPetje	77	39	12	37	**227**

Tabela 10: Klasifikacijska matrika napak za testno množico brez uporabe pasovno prepustnega filtra.

4.6. Test s terenskimi posnetki

V zadnjem sklopu testov so bili uporabljeni posnetki neposredno iz arhiva ljudske glasbe. Posnetki so bili povsem neobdelani, tako da so bili pričakovani tudi opazno slabši rezultati.

Testna množica je vsebovala 23 daljših terenskih posnetkov in dva krajša posnetka, pridobljena s kompaktnih plošč. Dolžina posnetkov se je gibala med 67 sekundami in 96 minutami, povprečna dolžina pa je bila okrog 25 minut. Posnetki so zajemali vseh pet kategorij in so bili tako kot prej razdeljeni na 10-sekundne segmente.

Preverjanje rezultatov teh testov je bilo za razliko od prejšnjih precej težje. Pri nobenem od posnetkov namreč nisem vnaprej vedel, v katero kategorijo spada. Poleg tega pa je v večini primerov v enem posnetku zajetih več kategorij. Zaradi tega je bilo treba vse posnetke poslušati in ročno določiti kategorijo za vsakega od 10-sekundnih segmentov.

Enako kot pri prejšnjem testu si je tudi tukaj možno rezultate razlagati na več načinov:

1. Klasifikacijsko točnost se izračuna z upoštevanjem vseh segmentov posnetka. Na ta način dobljeni rezultati so najslabši.
2. Pri izračunu klasifikacijske točnosti se izloči segmente, ki v kategorije učnih podatkov

ne spadajo (npr. petje, ki ni iz Rezije, govor, šum, itd.). Tako dobljena klasifikacijska točnost je vsaj enaka kot pri prvem načinu, po navadi pa opazno boljša.

3. 3. Pri izračunu klasifikacijske točnosti se enako kot pri drugem načinu izloči neveljavne segmente, le petje iz drugih pokrajin se upošteva med kategorijo Rezija – petje. Klasifikacijska točnost se na ta način lahko tudi zmanjša, prav tako pa ne more biti manjša kot pri prvem načinu.

Glede na delovanje programa se mi zdi najbolj smiselno upoštevati rezultat, dobljen na drugi način. Program bo namreč dani posnetek vedno klasificiral v eno izmed petih njemu znanih kategorij, tudi če posnetek ne pripada nobeni. Če bi programu dali v klasifikacijo posnetek, ki bi vseboval samo govor, bi bila po prvem način izračunana klasifikacijska točnost enaka 0.

Po drugi strani pa je pomembna tudi uspešnost klasifikacije nad celotnim posnetkom, saj smo takega tudi podali programu.

Tretji način sem upošteval predvsem zato, ker sem pri preverjanju rezultatov opazil, da v večini primerov program katerokoli petje klasificira v kategorijo Rezija – petje. V nadaljevanju so rezultati podani na vse tri opisane načine.

	Način 1	**Način 2**	**Način 3**
Test 1	25%	50%	45%
Test 2	4%	/	/
Test 3	42%	86%	79%

Tabela 11: Klasifikacijska točnost treh testov izračunana na tri zgoraj opisane načine.

Prvi test je bil izveden, še preden je bila ugotovljena napaka v učni množici. Ta je tako v kategoriji Prekmurje še vsebovala tudi posnetke rezijske glasbe. Rezultati so bili precej slabši od pričakovanih, kar je kazalo na napako v programu. Kmalu po pregledu rezultatov tega testa sem našel tudi napako v učni množici in sem slabe rezultate pripisal njej.

Drugi test je bil tako izveden s popravljeno učno množico, a so bili rezultati skoraj nični. Tokrat so bili pravilno klasificiranih samo 4 % segmentov. 97 % vseh segmentov je bilo klasificiranih v kategorijo Bela krajina, čeprav jih vanjo dejansko spada samo 2 %. Izračun rezultatov na druga dva načina zaradi tega ni bil smiseln. Tako slabi rezultati so kazali na še eno napako v programu ali podatkih. Ob pregledu delovanja programa sem opazil zelo veliko število javljenih opozoril o zaokroževanju vrednosti pri izračunu značilk terenskih posnetkov. Izkazalo se je, da so imeli posnetki frekvenco vzorčenja 48000 Hz, drugi posnetki v učni in testni množici pa 44100 Hz.

Za izvedbo zadnjega testa sem terenskim posnetkom spremenil frekvenco vzorčenja na 44100 Hz. S tem popravkom se je odstotek pravilno klasificiranih segmentov drastično dvignil. Točnost prepoznave je bila tako celo boljša od pričakovane. Klasifikacijska točnost, dobljena po prvem načinu, je sicer nizka, vendar je to posledica velikega števila segmentov posnetkov, ki niso spadali v nobeno od kategorij. V posnetkih je bilo namreč veliko intervjujev, tišine in posnetkov publike.

5. Zaključek

Implementacija se je glede na rezultate izkazala kot presenetljivo dobra, čeprav klasifikacijska točnost tudi pri 80 % in več ni vrhunska. Predvsem mislim, da bi jo dodatni testi na drugih posnetkih še zmanjšali. Vseeno pa točnost ne zaostaja veliko za podobnimi rešitvami. Primerjave z drugimi so dokaj težke. Njihova točnost je sicer podobna moji rešitvi, pri nekaterih še celo nekoliko nižja, vendar je precejšnja razlika v številu kategorij, ki jih je program sposoben razpoznati. Pri rugih je namreč razpoznavanje 10 kategorij dokaj običajno. Z višanjem števila kategorij se uspešnost prepoznave pričakovano niža.

Primerjavo algoritmov med seboj otežujejo tudi nestandardizirani testi, kar opažajo tudi drugi raziskovalci. V mojem primeru je prepoznavanje omejeno samo na pet kategorij slovenske ljudske glasbe. Primerjanje moje rešitve z neko bolj univerzalno rešitvijo tako verjetno ni ravno na mestu.

Prostora za izboljšave je še veliko, med drugim tudi glede hitrosti. Algoritem namreč ni ravno hiter. Na Core2Duo 2,2GHz sistemu s 4 GB pomnilnika za obdelavo posnetka potrebuje približno toliko, kolikor je posnetek dolg. Sam se z optimizacijo hitrosti nisem ukvarjal in me tudi ni preveč zanimala. Predvidevam, da bi s pravilno nastavitvijo parametrov pri algoritmih za izračun značilk iz posnetkov in z boljšim zaporedjem izvajanja teh algoritmov hitrost delovanja programa lahko opazno povečali.

Pri rezultatih je sicer pomemben tudi algoritem strojnega učenja, a ima večjo vlogo pravilno določanje značilk. Verjetno bi kakšen drug algoritem dal še boljše rezultate, vendar je je LMT dovolj uspešen. Najprej bi se bilo tako treba osredotočiti na izboljšavo izračuna značilk. Mislim, da so algoritmi, implementirani v MIRToolboxu, še vedno prava izbira, le uporabiti bi jih bilo treba na pametnejši način. Prek parametrov se ne spreminja samo nastavitev posameznega algoritma v MIRToolboxu. Precej algoritmov je implementiranih tako, da lahko z nastavitvami parametrov uporabnik za izračun značilke izbere različne pristope različnih raziskovalcev. Možnosti je veliko že za izračun ene same značilke, potrebna pa je pravilna kombinacija nastavitev izračunavanja več značilk. Pri tem so meni največji problem predstavljali zamudni testi vsake spremembe in sem 80–86% klasifikacijsko točnost sprejel kot dovolj dobro. Prepričan sem, da bi jo bilo možno še opazno dvigniti. Po izboljšanju izračuna značilk bi bilo verjetno še vedno smiselno testirati tudi druge algoritme strojnega učenja, vendar to rezultatov ne bi pomembno izboljšalo.

Zanimivo bi bilo tudi videti, kako se program obnaša pri večjem številu kategorij. To sem imel v načrtu, vendar so bili rezultati zaradi premajhnega števila posnetkov v dodatnih kategorijah preslabi. Tako bi moral še precej časa posvetiti zbiranju uporabnega števila posnetkov teh kategorij. Z boljšo učno množico mogoče popravki programa za dodatne kategorije niti ne bi bili potrebni, vendar gre samo za ugibanja.

Še ena od uporabnih izboljšav bi bila uporaba vnaprej zgrajenih klasifikacijskih modelov. V trenutni različici programa se mora ob vsakem zagonu klasifikacijski model iz učnih podatkov znova zgraditi. To sicer na prej omenjenem sistemu traja približno minuto in pol in ni časovno najzahtevnejši del izvajanja. Vendar so učni podatki večinoma vsakič enaki in je tako vsakokratna gradnja modela povsem nepotrebna. Z uporabo vnaprej zgrajenih modelov pa bi bil čas klasifikacije praktično zanemarljiv.

Največji problem vidim pri klasifikaciji posnetkov, ki ne spadajo v nobeno od petih programu znanih kategorij. Program bo katerikoli posnetek vedno klasificiral v eno izmed teh kategorij, tudi če gre za npr. tišino, šum ali čist ton, ki zelo očitno ne spadajo v nobeno. V trenutni obliki je program smiselno uporabljati samo na posnetkih, ki v te kategorije spadajo v

celoti. Le tako bodo dobljeni rezultati uporabni. Trenutna rešitev je preveč omejujoča za kaj več kot testiranje. Program bi bilo treba dopolniti še s kategorijo Neznano, kamor bi bili klasificirani posnetki ali njihovi segmenti, ki ne spadajo v pet kategorij. S tem bi postal primeren za splošno uporabo.

Od vsega naštetega bi bilo najprej treba dodati kategorijo Neznano in nove kategorije za prepoznavo. Če bi bilo to možno brez prevelikega poslabšanja rezultatov, bi se program mogoče že lahko kosal tudi z drugimi rešitvami. Z izvedenimi vsemi naštetimi popravki pa bi ga bilo smiselno tudi prijaviti na katero od tekmovanj, kot je npr. MIREX.

Seznam slik

Seznam tabel

Literatura

[1] J. C. Brown., „Computer identification of musical instruments using pattern recognition with cepstral coefficients as features”, 1999

[2] P. Hamel, S. Wood, D. Eck, „Automatic Identification of Instrument Classes in Polyphonic and Poly-Instrument Audio“, 2009, dostopno na http://ismir2009.ismir.net/proceedings/PS3-2.pdf

[3] W. M. Hartmann, „Signals, sound, and sensation“, American Inst. of Physics, 1997, str. 283

[4] T. Heittola, A. Klapuri and T. Virtanen, „Musical Instrument Recognition in Polyphonic Audio Using Source-Filter Model for Sound Separation“, 2009 dostopno na http://ismir2009.ismir.net/proceedings/OS3-2.pdf

[5] K. Jensen, „Timbre Models of Musical Sounds“, 1999, pogl. 7.2.5., dostopno na http://vbn.aau.dk/files/46619185/TMoMS.pdf

[6] N. Landwehr, M. Hall, E. Frank, „Logistic Model Trees“, 2003, dostopno na http://www.cs.waikato.ac.nz/~ml/publications/2003/landwehr-etal.pdf

[7] K. D. Martin, „Sound-Source Recognition: A Theory and Computational Model”, 1999, dostopno na http://sound.media.mit.edu/Papers/kdm-phdthesis.pdf

[8] W. A. Sethares, „Tuning, timbre, spectrum, scale“, Springer, 1999, str. 29

[9] Uradna spletna stran ISMIR: http://www.ismir.net/

[10] Uradna spletna stran orodja MATLAB: http://www.mathworks.com/products/matlab/

[11] Uradna spletna stran orodja MIRToolbox: https://www.jyu.fi/hum/laitokset/musiikki/en/research/coe/materials/mirtoolbox

[12] Uradna spletna stran orodja Weka: http://www.cs.waikato.ac.nz/ml/weka/

[13] Uradna spletna stran tekmovanja MIREX: http://www.music-ir.org/mirex/wiki/MIREX_HOME