Universitatea POLITEHNICA din București
Facultatea de Electronică, Telecomunicații și Tehnologia Informației

Sistem automat de transcriere a muzicii folosind rețele neurale adânci

Lucrare de licență

Prezentată ca cerință parțială pentru obținerea
titlului de *Inginer*
în domeniul *Electronică, Telecomunicații și Tehnologia Informației*
programul de studii *Microelectronică, Optoelectronică și Nanotenologii*

Conducători științifici
Prof. Univ. Dr. Ing. Corneliu Burileanu
As. Univ. Drd. Ing. Ana Neacșu

Absolvent
Marian Negru

Anul 2020

Universitatea "Politehnica" din Bucureşti
Facultatea de Electronică, Telecomunicaţii şi Tehnologia Informaţiei
Departamentul **DCAE**

Anexa 1

TEMA PROIECTULUI DE DIPLOMĂ
a studentului **NEGRU P. Marian , 445E-MON.**

1. Titlul temei: Sistem automat de transcriere a muzicii folosind rețele neurale adânci.

2. Descrierea contribuţiei originale a studentului (în afara părţii de documentare) şi specificaţii de proiectare:
Scopul principal al acestei lucrări presupune recunoașterea automată a muzicii folosind metode de învățare automată bazate pe algoritmi de inteligență artificială. Mai exact, se va proiecta și antrena o rețea neurală folosind o bază de date proprie alcătuită din diverse melodii sau exerciții muzicale. Pentru fiecare notă muzicală din baza de date este necesar un 'ground-truth', folosit pentru antrenarea supervizată a rețelei, precum și onset-ul si offset-ul acestei note, pentru a putea determina durata acesteia și momentul în care este cântată. Aceste informații vor fi stocate in format MIDI.

Instrumentul la care se vor interpreta aceste melodii va fi pianul și vor fi considerate doar melodii monofonice (o singură notă la un moment de timp). Semnalul audio va fi digitalizat, iar din el se vor extrage o serie de trăsături utile pentru această aplicație (ex: Transformata Fourier). Se vor testa mai multe metode și trăsături, fie în domeniul timp, fie în domeniul frecvență, în vederea dezvoltării unui sistem cât mai performant. Se vor proiecta mai multe arhitecturi de rețele, variind hiper-parametrii aferenți (numărul de straturi, tipul de activare) în vederea găsirii unui compromis optim între performanță și resurse de calcul necesare.

Validarea se va realiza într-un context real, sistemul recunoscând notele cântate de către un pian.

3. Resurse folosite la dezvoltarea proiectului:
Resurse de calcul (Placă grafică), Python, Tensorflow, Keras

4. Proiectul se bazează pe cunoştinţe dobîndite în principal la următoarele 3-4 discipline:
SS, PDS, TAPDS, PC, SDA

5. Proprietatea intelectuală asupra proiectului aparţine: U.P.B.

6. Data înregistrării temei: 2019-11-28 21:32:13

Conducător(i) lucrare,
Prof. dr. ing. Corneliu BURILEANU

As. univ. dr. ing. Ana Antonia NEACȘU

Student,

Director departament,
Prof. dr. ing Claudius DAN

Decan,
Prof. dr. ing. Mihnea UDREA

Cod Validare: **18bd463fcd**

Declaraţie de onestitate academică

Prin prezenta declar că lucrarea cu titlul *Sistem automat de transcriere a muzicii folosind rețele neurale adânci*, prezentată în cadrul Facultăţii de Electronică, Telecomunicaţii şi Tehnologia Informaţiei a Universităţii "Politehnica" din Bucureşti ca cerinţă parţială pentru obţinerea titlului de *Inginer* în domeniul Inginerie Electronică şi Telecomunicaţii/ Calculatoare şi Tehnologia Informaţiei, programul de studii *Microelectronică, Optoelectronică și Nanotenologii* este scrisă de mine şi nu a mai fost prezentată niciodată la o facultate sau instituţie de învăţământ superior din ţară sau străinătate.

Declar că toate sursele utilizate, inclusiv cele de pe Internet, sunt indicate în lucrare, ca referinţe bibliografice. Fragmentele de text din alte surse, reproduse exact, chiar şi în traducere proprie din altă limbă, sunt scrise între ghilimele şi fac referinţă la sursă. Reformularea în cuvinte proprii a textelor scrise de către alţi autori face referinţă la sursă. Înţeleg că plagiatul constituie infracţiune şi se sancţionează conform legilor în vigoare.

Declar că toate rezultatele simulărilor, experimentelor şi măsurătorilor pe care le prezint ca fiind făcute de mine, precum şi metodele prin care au fost obţinute, sunt reale şi provin din respectivele simulări, experimente şi măsurători. Înţeleg că falsificarea datelor şi rezultatelor constituie fraudă şi se sancţionează conform regulamentelor în vigoare.

Bucureşti, Iunie 2020.
Absolvent: Marian Negru

Negru

Cuprins

Lista figurilor

Lista tabelelor

Lista acronimelor

WAV = Waveform Audio File
MIDI = Musical Instrument Digital Interface
DNN = Deep Neural Networks
AMT = Automatic Music Transcription
MIR = Musical Information Retrieval
STFT = Short Time Fourier Transform
MFCC = Mel-Frequency Cepstral Coefficients
DCT = Discrete Cosine Transform
CQT = Constant-Q Transform
HMM = Hidden Markov Models
ACF = Autocorrelation Function
AMDF = Average Magnitude Difference Function
NCCF = Normalized Cross-Correlation Function
SWIPE = Sawtooth Waveform Inspired Pitch Estimator
ReLU = Rectified Linear Unit
SGD = Stohastic Gradient Descend
Adam = Adaptive Moment Estimation
BPM = Beats Per Minute
SSL = Staționar în Sens Larg
FFT = Fast Fourier Transform
CWT = Continuous Wavelet Transform
DWT = Discrete Wavelet Transform
MSE = Mean Squared Error
CNN = Convolutional Neural Networks
RNN = Recurrent Neural Networks

Introducere

Acest proiect presupune realizarea unui sistem capabil să transcrie o piesă muzicală, în acest caz, monofonică – i.e. o melodie în care notele nu sunt suprapuse în același moment de timp. Pentru proiectarea unui astfel de sistem sunt necesare metode de învățare automată bazate pe algoritmi de inteligență artificială.

De-a lungul timpului, muzica a evoluat continuu, pornind încă din epoca preistorică, unde melodiile reprezentau lovituri repetate între pietre, lemne sau alte obiecte uzuale. Omul a simțit nevoia să creeze un sistem de reprezentare grafică a creațiilor muzicale, numit ulterior portativ, sunetele fiind notate prin semne alese din literele alfabetului. Grecii antici au avut o contribuție majoră în acest aspect, deoarece studiau cu mare interes domeniul acustic și al matematicii. Aceștia au împărțit sunetele în scări ce alcătuiesc diferite tonalități și au introdus noțiunea de ritm muzical. În zilele noastre, piesele pot deveni foarte complexe (interpretări de orchestră), sau pot fi chiar sintetizate în absența intrumentelor muzicale folosind diferite programe speciale.

Motivația acestui proiect constă în observația că multe melodii create de un popor se pierd de-a lungul timpului datorită faptului că nu există o înregistrare și o sistematizare a acestora. În multe cazuri, modalitatea de transmitere a melodiilor se realizează prin cale auditivă, popular spus, „după ureche". Din păcate, în acest mod melodia suferă modificări de la transmiteri succesive între indivizi, până când, după un timp, fie se pierde datorită insuficientelor informații, fie se modifică atât de mult încât devine o melodie complet diferită.

Această lucrare oferă posibilitătea de transcriere automată a melodiilor compuse, în scopul de a fi păstrate în timp și deci transmise corect mai departe, realizând astfel conservarea folclorului autentic.

Proiectul își propune realizarea următoarelor obiective:

- *Clasificarea notelor* – presupune recunoașterea automată a unei note în funcție de anumite proprietăți ale sunetului produs caracteristic notei;
- *Detecția onset / offset* – reprezintă determinarea momentului de apariție a unei note, respectiv momentul opririi acesteia;
- *Estimarea intensității* – semnifică aproximarea volumului unei note produse.

Capitolul 1
Noțiuni teoretice

1.1 Clasificarea semnalelor

Un *semnal* este o cantitate fizică ce reprezintă o funcție de una sau mai multe variabile independente, cum ar fi timpul, distanța, temperatura etc. Dacă un semnal este definit de o funcție de o singură variabilă independentă, este numit *semnal unidimensional* (1-D); dacă acesta este funcție de mai mult de o variabilă, este denumit *multidimensional*(M-D) [1]. Deoarece semnalul muzical este un semnal unidimensional, pentru acest proiect vom discuta doar despre aceste tipuri de semnale 1-D.

Semnalele pot apărea în mod natural, dar pot fi și stimulate sau generate într-un mod artificial.

Semnalele sunt de mai multe feluri: electrice (tensiuni, curenţi), electromagnetice (intensitate câmp electric, inducţie câmp magnetic), mecanice, optice, termice, biologice etc.

Semnalele pot fi considerate:

1. *Utile* – dacă sunt folosite pentru a realiza un anumit scop;
2. *Perturbații* – orice semnal, diferit de cel util, este o perturbaţie. Aceste semnale perturbatoare sunt de obicei numite *zgomote*.

Un semnal este:

(A) *Staționar* – dacă semnalul are parametrii caracteristici constanți în timp;

(B) *Nestaționar* – în cazul în care parametrii statistici ai semnalului variază în funcție de timp.

În funcție de modul de variație în timp a semnalelor, acestea pot fi *deterministe* sau *aleatorii*:

- Dacă evoluţia în timp a semnalului poate fi descrisă printr-o funcţie de timp $s(t)$, astfel încât proprietățile sale pot fi cunoscute la orice moment, semnalul este *determinist*.
- Dacă în cazul unui semnal nu este posibilă descrierea evoluţiei sale în timp, adică predicţia proprietăților sale nu este posibilă, semnalul este *aleator* (întâmplător). În acest caz, se pot face cel mult aprecieri probabilistice asupra caracteristicilor semnalului aleator. De exemplu, se poate determina probabilitatea ca la un moment de timp dat nivelul semnalului să se încadreze între anumite limite.

În comunicații, de obicei, semnalele utile, purtătoare ale informaţiei de transmis, sunt aleatorii; de exemplu, semnalul vocal la ieşirea unui microfon, semnalul de imagine video, semnalul la ieşirea unui scanner etc., deci nu vom putea să prezicem ce se va spune, ce se va întâmpla în câmpul filmat, ce urmează să se scaneze pe pagină. Pe de altă parte, există şi semnale deterministe, cum sunt semnalele de test (de exemplu semnale sinusoidale), semnale de sincronizare (în TV) etc. [2]. Semnalul muzical este de asemenea un semnal aleator, deoarece nu putem să prezicem ce note se vor cânta.

Semnalele mai pot fi clasificate în:

- Semnale periodice: un semnal $x(t)$ este periodic dacă funcția de timp care îl descrie satisface relația:
$$x(t) = x(t \pm kT) \quad , \quad k \in \mathbb{N} \tag{1.1}$$
unde T reprezintă perioada semnalului, definită ca intervalul minim de timp după care semnalul $x(t)$ se repetă identic.

- Semnale neperiodice: sunt considerate un caz limită al semnnalelor periodice, la care perioada tinde spre infinit.

După modul de evoluție în timp a semnalelor utilizate în telecomunicaţii, deterministe sau aleatorii, acestea pot fi: analogice (continue, cu nivel variabil), cuantizate, eşantionate sau eşantionate şi cuantizate, conform figurii 1.1 [2]:

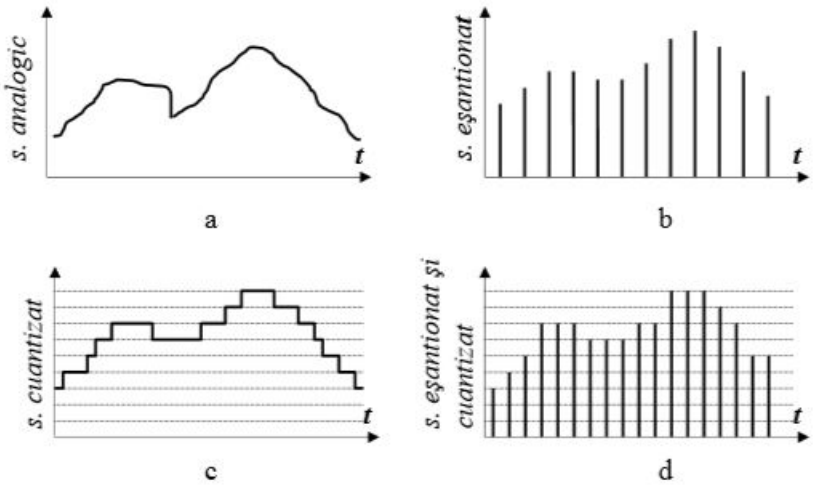


Figura 1.1: Semnal analogic (a), eșantionat (b), cuantizat (c), discret (d)

(a) *Semnalele analogice* au niveluri specificate, existente, într-un număr infinit de puncte dintr-un interval de niveluri şi într-un număr infinit de puncte pe axa timpului şi pot prezenta discontinuităţi;

(b) *Semnalele eşantionate* (discretizate în timp) au niveluri specificate (existente) numai în anumite momente care formează un şir discret;

(c) *Semnale cuantizate* (discretizate în nivel, cu continuitate în timp), cu niveluri existente pe întreaga axă a timpului, dar care formează un şir de valori discrete în intervalul de niveluri limită;

(d) *Semnale eşantionate şi cuantizate* cu niveluri şi momentele de existenţă formând şiruri discrete (numite în general *semnale discrete*).

Un *semnal* conține de obicei informație despre starea sau evoluția unui sistem fizic și este necesară prelucrarea lui astfel încât informația conținută să fie extrasă parțial sau total. La modul general, putem spune că semnalele audio sunt semnale unidimensionale, aleatorii și neperiodice. Din aceste motive, prelucrarea lor este un aspect foarte important, deoarece este mult mai ușor de lucrat cu semnale cvasistaționare (semnale deterministe pe durate scurte de timp) și periodice.

1.2 Reprezentarea semnalelor

Orice semnal $x(t)$ poate fi caracterizat prin două reprezentări [3]:

- Reprezentarea în *domeniul timp*, numită forma de undă a semnalului;
- Reprezentarea în *domeniul frecvență*, numită spectrul de frecvențe al semnalului.

Aceste reprezentări caracterizează în mod univoc semnalul, adică unei reprezentări în timp îi corespunde o singură reprezentare în frecvență și invers, unei reprezentări în *domeniul frecvență* îi corespunde o singură reprezentare în *domeniul timp*.

Legătura între aceste două reprezentări se face cu ajutorul unor funcții matematice numite „transformate". Pentru semnalele periodice această trecere se face cu ajutorul seriilor Fourier. În cazul semnalelor neperiodice tranziția între cele două reprezentări este realizată prin transformata Fourier sau Laplace.

Deoarece se dorește analiza unui semnal complex (semnalul audio), o metodă posibilă este descompunerea acestuia într-o sumă de semnale mai simple și ușor de manipulat (funcții elementare).

Pentru a înțelege mai bine ce reprezintă aceste două domenii, este necesară o interpretare fizică a fenomenelor:

- În *domeniul timp*, un semnal poate fi scris ca o sumă de impulsuri unitate deplasate și scalate:

$$x(n) = \sum_{k=-\infty}^{+\infty} x(k)\delta(n-k) \tag{1.2}$$

  impulsul unitate fiind definit ca:

$$\delta(n) = \begin{cases} 1, & \text{pentru n} = 0 \\ 0, & \text{pentru } n \neq 0 \end{cases} \tag{1.3}$$

- În *domeniul frecvență*, un semnal este scris ca o sumă de semnale sinusoidale:

$$X(e^{j\omega}) = \sum_{-\infty}^{+\infty} x(n)e^{-j\omega n} \tag{1.4}$$

  unde $X(e^{j\omega})$ reprezintă spectrul de frecvențe al semnalului $x(n)$.

Datorită dualității timp-frecvență, fiecare schimbare a semnalului într-un domeniu va avea un efect în reprezentarea celui de-al doilea domeniu (orice modificare în timp va fi observată în frecvență, și invers). Acest efect va trebui luat în considerare în momentul prelucrării semnalului.

Astfel, prin cele două reprezentări, evoluția unui semnal este mai ușor de vizualizat în *domeniul timp*, însă în frecvență evoluția semnalului poate fi „auzită" (deoarece spectrul de frecvențe arată ce frecvențe apar în semnal). Spectrul de frecvențe este o mărime complexă, fiind alcătuită din spectrul de amplitudine $\mid X(e^{j\omega}) \mid$ $(Re(X(e^{j\omega})))$ și spectrul de fază $(Im(X(e^{j\omega}))$. Pentru semnalul audio se va folosi doar spectrul de amplitudine, neglijându-se spectrul de fază, întrucât urechea umană este, în mare măsură, insensibilă la variații ale fazei semnalului.

Pentru a obține un semnal discret este necesară eșantionarea periodică a semnalului analogic, conform *teoremei eșantionării* (Shannon) [3]: orice semnal $x(t)$, ce are o bandă de frecvență limitată (banda nu este infinită), este complet definit (univoc determinat) prin eșantioanele sale $x(nT)$ dacă perioada de eșantionare T îndeplinește condiția:

$$T \leq \frac{1}{2f_M} \tag{1.5}$$

unde f_M este frecvența maximă a spectrului semnalului eșantionat.

Această condiție este numită *condiția lui Nyquist* și se poate rescrie sub forma:

$$\frac{f_e}{2} = f_N \geq f_M \tag{1.6}$$

unde f_e reprezintă frecvența de eșantionare iar f_N este frecvența Nyquist.

Astfel, această condiție sugerează că pentru o frecvență de eșantionare mai mică decât dublul frecvenței maxime din spectrul semnalului continuu, componentele periodice ale spectrului $X(e^{j\omega})$ se vor suprapune parțial, iar în acest mod se va produce eroarea de spectru suprapus (aliere spectrală) – i.e. vor apărea frecvențe false, nedorite în spectru, datorită fenomenului de oglindire spectrale.

1.3 Semnalul audio

1.3.1 Caracteristici generale

Semnalele cu spectrul în intervalul [20Hz – 20kHz] sunt considerate semnale de audiofrecvenţă (audio, AF), deoarece sunt percepute de urechea umană când sunt sub formă de variaţii ale presiunii aerului (sunet).

Semnalul audio poate fi *vocal* sau *muzical*.

Pentru *semnalul vocal* s-a constatat că cea mai mare parte a energiei spectrale este concentrată într-un interval mic de frecvențe, în jurul benzii de (300Hz – 2kHz). Timbrul pe de altă parte (cel care identifică vorbirea), este determinat de frecvențele mai mari, până la 3 – 4kHz. Din acest motiv, se consideră acceptabilă banda (300Hz – 3.4kHz).

Există mai multe clase de semnale audio în funcție de lărgimea de bandă, gama dinamică și calitatea oferită [4]:

- semnale vocale de *bandă îngustă* (300Hz – 3.4kHz) – folosite în special pentru aplicații de transmisie pe canal telefonic;
- semnale vocale de *bandă largă* (50Hz – 7kHz) – utilizate în aplicații de recunoaștere a vorbirii sau a vorbitorului, precum și în sinteză a vorbirii cu integibilitate și naturalețe foarte bune;
- semnale audio de *bandă intermediară* (pentru transmisii radio AM și FM, cu un maxim al benzii de frecvențe de aproximativ 10kHz, respectiv 15kHz;

- semnale audio de *bandă largă și înaltă fidelitate* (banda 20Hz – 20kHz) - folosite pentru stocare pe CD, DAT, DVD, BD, etc.

Sunetul este caracterizat de patru atribute:

1. *Înălțime* – aceasta este reprezentată de frecvența fundamentală a semnalului audio, uzual numită „pitch" în literatură;
2. *Durată* – reprezintă intervalul de timp în care este menținut sunetul;
3. *Intensitate* – este senzația produsă de un sunet, numită și tărie sau volum sonor;
4. *Timbru* – semnifică proprietatea sunetului prin care se identifică sursa sonoră.

Procesul prin care se atribuie nume de note diferitelor înălțimi (frecvențe) se numește acordare [5]. În muzica contemporană, acordajul se face astfel încât frecvenței de 440Hz să îi corespundă nota `La` (`A`). Diferența în numărul de vibrații pe secundă dintre două sunete cu înălțimi diferite se numește interval, octava fiind definită ca intervalul în care frecvența unui sunet se dublează. Aceasta a fost împărțită într-o scară de 7 note muzicale, notate alfabetic: `A` = `La`, `B` = `Si`, `C` = `Do`, `D` = `Re`, `E` = `Mi`, `F` = `Fa`, `G` = `Sol`. Johann Sebastian Bach a fost cel care a introdus scara uniform temperată în muzică. În această scară există 12 semitonuri în interiorul fiecărei octave, frecvențele notelor succesive fiind separate printr-un interval de un semiton, formând un raport constant, egal cu $\sqrt[12]{2} \approx 1.0594631$. Tabelul 1.1 prezintă notele muzicale cuprinse în intervalul de aproximativ 261Hz și 523Hz:

Denumire științifică	Frecvența [Hz]
C5 (**C tenor**)	523.251
B4	493.883
A♯4 sau B♭4	466.164
A4 (**A440**)	**440.000**
G♯4 sau A♭4	415.305
G4	391.995
F♯4 sau G♭4	369.994
F4	349.228
E4	329.628
D♯4 sau E♭4	311.127
D4	293.665
C♯4 sau D♭4	277.183
C4 (**C mijlociu**)	261.626

Tabela 1.1: Octava C4 – C5

Durata reprezintă caracteristica sunetului de a fi mai lung sau mai scurt în timp. Aceasta se calculează din momentul impactului până la dispariția ultimei vibrații sonore percepute. Duratele notelor muzicale și ale pauzelor nu sunt definite absolut, ci relativ, fiind însă proporționale între ele. Prin definiție, pătrimea este considerată ca lungime de referință. Adăugarea unui punct asupra duratei unei note va prelungi durata notei cu jumătate din valoarea sa inițială. Tabelul 1.2 prezintă duratele notelor uzuale:

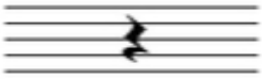
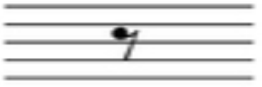
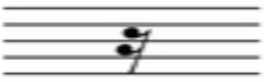
Nume	Semnul notei	Semnul pauzei	Durata
notă întreagă			4 timpi
doime			2 timpi
pătrime			1 timp
optime			$\frac{1}{2}$ timp
șaisprezecime			$\frac{1}{4}$ timp

Tabela 1.2: Duratele uzuale ale notelor

Prin intensitate sonoră se înțelege senzația pe care o produce asupra organului nostru auditiv amplitudinea unei vibrații sonore, sau altfel spus, volumul vibrației. Cu cât amplitudinea vibrațiilor este mai mare, cu atât crește și intensitatea sunetului rezultat, și invers. Intensitatea este definită ca puterea vibrației produse pe unitatea de arie. Spre exemplu, pentru o sferă de rază r, intensitatea este:

$$I_S = \frac{P}{4\pi r^2} \tag{1.7}$$

Se definește nivelul de intensitate sonoră N_S prin formula:

$$N_S(dB) = 10lg\frac{I_S}{I_{0S,min}} \tag{1.8}$$

unde $I_{0S,min}$ este intensitatea sonoră minimă ce poate fi percepută de urechea umană pentru sunetul de referință, cu frecvența $\upsilon_0 = 1$kHz, numit și sunet normal. Domeniul percepției auditive umane este prezentat în figura 1.2 [6]:

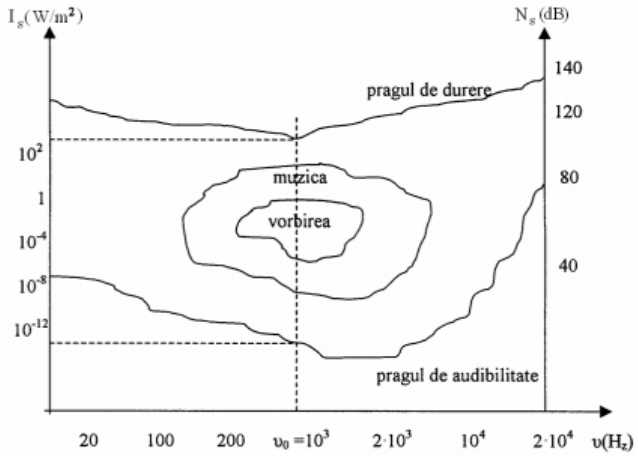


Figura 1.2: Domeniul de percepție auditivă accesibil urechii umane

Această senzație se mai numește tărie și depinde de frecvență: la aceeași intensitate (presiune sonoră), senzația produsă este mai mare la frecvențe medii (aproximativ 1000Hz) decât la frecvențele cele mai înalte sau mai joase [2]. Deci, pentru a obține aceeași senzație este nevoie de o intensitate sonoră mai mare pentru frecvențele joase sau înalte față de frecvențele medii. Cea mai mică intensitate sonoră pentru care se produce o senzație auditivă se numește prag de audibilitate și diferă în funcție de persoană, condiții de măsură și frecvență. Astfel, zgomotele sonore de joasă și înaltă frecvență se aud mai slab decât cele de frecvențe medii.

Timbrul reprezintă mulțimea de proprietăți ale semnalului ce permit diferențierea unei surse sonore, precum vocea umană sau instrumentele muzicale.

Sonoritatea unui instrument depinde de următoarele aspecte:

- *modul de execuție* – diferit în funcție de tipul de instrument (ex: instrumente cu coarde, de suflat, etc.);
- *analiza spectrului* – fiecare notă are în general o frecvență fundamentală caracteristică, dominantă ca amplitudine în spectru. Totuși, instrumentele muzicale se pot diferenția prin multiplii ai acestei frecvențe fundamentale, numite armonici, ce au valori ale amplitudinilor diferite în funcție de tipul de instrument, deci anvelope spectrale diferite (uneori aceste armonici pot avea valori chiar mai mari decât frecvența fundamentală, ca de exemplu vioara);
- *analiza atacului* – perioada de atac reprezintă începutul unui sunet muzical, adică intervalul scurt de timp necesar declanșării mecanismelor producerii sunetului. Acest factor va fi diferit între intrumente (instrumentele de percuție vor avea alt atac față de instrumentele cu coarde, etc.).

1.3.2 Percepția sunetelor

Până acum am discutat despre ce înseamnă un semnal audio, dar este important să precizem și cum este acesta perceput de organul auditiv al omului, urechea.

Deoarece fiecare sunet se manifestă printr-o vibrație a aerului, toate undele sonore provenite din diverse surse sunt unice. Astfel, fiecare persoană sau lucru se va auzi diferit și vor avea intensități diferite. Urechea umană are rolul de a capta undele sonore și de a le transforma în mesaje pe care creierul le poate înțelege. Definim în acest mod trei părți importante ale urechii unei persoane [7]:

- *Urechea externă* – are rolul de a capta sunetul și de a-l direcționa către urechea medie. Este formată din pavilion, canal auditiv și timpan. Prin elasticitatea și forma sa, pavilionul urechii este esențial în detectarea direcției din care vin sunetele și favorizează anumite frecvențe în detrimentul altora (urechea are un maxim de sensibilitate pentru sunete cu frecvența de aproximativ 3500Hz);
- *Urechea medie* – transformă undele sonore în unde de presiune mecanică, pe care le transferă la lichidele din urechea internă. Prezintă trei oscioare: ciocanul, nicovala și scărița. Urechea medie are rolul de a asigura corespondența impedanței între aer și apă, precum și reducerea transmisiei sunetelor, în special a celor de frecvență joasă, atunci când acestea au o intensitate foarte mare (reflex acustic);
- *Urechea internă* – transformă undele de presiune în semnale pe care creierul le poate înțelege. În interiorul urechii interne există cohleea, un organ ce are o formă de spirală, iar de-a lungul ei se află membrana Reissner (vestibulară) și membrana bazilară.

Procesarea semnalului sonor de către ureche prezintă următoarele etape, ilustrate în figura 1.3 [8]:

1. Sunetul pătrunde în canalul urechii – Undele sonore parcurg canalul auditiv și lovesc timpanul.

2. Timpanul și oasele aferente vibrează – Aceste unde sonore fac ca timpanul și cele trei oase subțiri (ciocanul, nicovala și scărița) din urechea medie să vibreze.

3. Lichidele se deplasează prin urechea internă – Vibrațiile create de urechea medie sunt transmise prin lichidul din cohlee de-a lungul membranei bazilare, membrană fibroasă ce prezintă 20000 – 30000 fibre bazilare sau corzi. Această membrană rezonează cu sunetele de frecvențe înalte la bază (corzi groase și scurte) și cu sunetele de frecvențe joase la vârf (corzi subțiri și lungi). Aceste vibrații se convertesc apoi în semnale chimice pentru nervul acustiv.

4. Nervii acustici comunică cu creierul – Nervul acustic trimite în final informația la creier prin impulsuri electrice, unde sunt interpretate ca sunet.

Figura 1.3: Percepția sunetului prin urechea umană

Astfel, urechea umană este un sistem foarte complex, fiind capabilă chiar să identifice semnalul audio în lipsa frecvenței fundamentale, pe baza armonicelor. Aceasta, ca și restul senzorilor biologici (ochii, simțul olfactiv, etc.), are un răspuns logaritmic la stimul.

1.3.3 Standardul MIDI

Un semnal audio poate fi reprezentat în două moduri:

- Colecție de eșantioane (ex: formatul `WAV`)

- `MIDI` (Musical Instrument Digital Interface) – memorează „evenimente de sunet”. Este utilizat un limbaj scripting pentru a specifica mesajele prin care se indică notele, duratele acestora și intrumentele folosite. Fiecare mesaj descrie un „eveniment” (cum ar fi schimbarea notei, a cheii, a tempo-ului, etc.). Concret, acest format reține orice legat de semnalul audio, în afară de semnalul audio propriu-zis.

Avantaje și dezavantaje ale standardului MIDI:

- *Dezavantaj*: mesajele MIDI generează sunetul corespunzător prin intermediul unui sintetizator (se caută fiecare sunet într-o memorie de sunete sau se compune sunetul pe baza unui calcul matematic), proces numit sinteză FM. Din acest motiv, sunetul produs este inferior celui redat de un fișier ce reține eșantioanele semnalului (sunetul dat de MIDI poate fi artificial sau mecanic, în timp ce un fișier de eșantioane poate memora interpretarea și subtilitățile muzicianului).

- *Avantaj*: se pot utiliza instrumente sau echipamente cu clape MIDI, conectate direct la un calculator prin cablu USB, pentru a înregistra o piesă muzicală. Fișierul obținut poate fi ulterior editat (se poate modifica instrumentul folosit, înălțimea și durata notelor, intensitatea acestora, cheia utilizată, etc.). Întrucât MIDI este un standard, se poate face transferul de fișiere MIDI direct între echipamentele MIDI sau între calculatoare.

Dispozitivele hardware ce generează mesajele MIDI se numesc controllere MIDI (ex: keyboard pian electronic), iar cele ce interpretează mesaje MIDI și generează sunete sunt sintetizatoarele MIDI (unele dispozitive pot realiza ambele operații). Un secvențiator MIDI reprezintă un dispozitiv hardware sau aplicație software ce permite recepția, editarea și memorarea datelor MIDI.

Programele realizate pentru prelucrarea semnalelor audio de eșantioane pot citi și fișiere MIDI și se poate realiza conversia într-un format de eșantioane (WAV), întrucât se cunoaște toată informația despre semnal. Transformarea unui fișier de eșantioane într-un fișier MIDI în schimb este un lucru mult mai greu de realizat.

Este necesară o conversie a proprietăților semnalului audio într-un număr cât mai mic de biți pentru a stoca informația cu o dimensiune redusă și pentru a o transmite cât mai rapid. Caracteristicile semnalelor audio prezentate anterior sunt astfel diferite în formatul MIDI:

1. *Frecvența* – va fi reprezentată folosind un număr MIDI, numit MIDI „pitch", în intervalul [0 – 127] (deci 7 biți), formula pentru calculul frecvenței semnalului fiind următoarea:

$$f = 440 \times 2^{\frac{(m-69)}{12}} Hz \tag{1.9}$$

   unde m reprezintă numărul MIDI caracteristic frecvenței f.

2. *Durata* – aceasta nu există într-un fișier MIDI. În schimb, se definește momentul în care nota începe a fi cântată, numit „onset", respectiv momentul în care nota se termină, numit „offset", durata notei fiind calculată cu ajutorul formulei:

$$\text{durată}[s] = \text{offset} - \text{onset} \tag{1.10}$$

3. *Intensitatea* – în formatul MIDI este definită ca forța măsurată de senzorul clapei MIDI aplicată pentru a produce nota respectivă. Aceasta va fi reprezentată tot printr-un număr în intervalul [0 – 127], unde numărul 0 semnifică absența unei note (deci teoretic numărul 1 va fi cea mai mică intensitate posibilă a unei note, practic inauzibilă), iar numărul 127 este intensitatea maximă pe care o notă o poate avea.

4. *Timbrul* – acesta este ales dintr-un tabel de instrumente din secvențiatoarele MIDI și se poate modifica în funcție de cerințele utilizatorului.

1.4 Rețele neurale

În general, metodele tradiționale de prelucrare a semnalelor discrete în timp sunt bazate pe anumite presupuneri (sisteme de analiză liniare, semnale staționare cel puțin pe o durată scurtă de timp, zgomotele suprapuse peste semnalul util sunt considerate semnale aleatorii staționare, etc.). Toate aceste metode parametrice oferă performanțele dorite doar în anumite condiții particulare și sunt complexe din punct de vedere matematic. Deoarece în multe situații reale aceste presupuneri nu sunt adevărate, este necesară punerea problemei din alte puncte de vedere [4].

Un sistem „inteligent" se definește ca un sistem ce se poate adapta continuu la condițiile date, prin învățarea din experiență. Aceste sisteme pot astfel generaliza, utilizând noțiunile învățate în afara domeniului de experiență. Adaptarea (învățarea) reprezintă modificarea parametrilor interni ai sistemului astfel încât spațiului mărimilor de la intrare să i se asocieze un spațiu al mărimilor de la ieșire – i.e. realizarea unei modelări a funcției f:

$$f : X \to Y \tag{1.11}$$

unde X este spațiul intrărilor, iar Y este spațiul ieșirilor.

Rețelele neurale (sau sisteme neurale artificiale) reprezintă astfel de sisteme „inteligente", ele fiind capabile să estimeze funcții ce fac corespondența între perechea de date intrare–ieșire, sau extrag anumiți parametrii caracteristici din spațiul datelor de intrare. Aceste rețele funcționează precum unor sisteme de calcul, formate dintr-un număr de unități de prelucrare a informației (neuroni artificiali), interconenctate și capabile să învețe, în scopul rezolvării unei sarcini.

Sistemele neurale artificiale nu sunt programate, ci antrenate pentru a executa anumite sarcini, procesarea informației fiind distribuită în întreaga structură spre deosebire de sistemele obișnuite de calcul. Astfel, execuția nu este de tip secvențial, rețeaua fiind capabilă să exploreze simultan mai multe ipoteze datorită paralelismului și a interconectării unităților de prelucrare a informației, legate prin intermediul unor funcții ponderi, ce se modifică în timpul adaptării rețelei. Putem spune astfel că proiectarea unei rețele neurale constă în definirea caracteristicilor nodurilor (alegerea funcției de activare potrivită), alegerea arhitecturii rețelei (numărul de straturi, noduri și tipul de conexiune) și specificarea algoritmului de optimizare. Acesta din urmă reprezintă un proces iterativ, ce stă la baza mecanismului de antrenare și se bazează pe minimizarea unei funcții de cost.

Aceste rețele sunt privite ca niște „cutii negre" (*eng. black boxes*), ce primesc „intrări" și produc „ieșiri", realizând astfel diferite tipuri de operații [4]:

1. *Clasificare* – semnifică repartizarea într-un set de clase (predefinite la ieșirea rețelei) a vectorilor de intrare în rețea. Această operație poate fi privită ca o asociere de modele, fiind o grupare a mărimilor de la intrare în funcție de mărimile de la ieșire. Clasificarea este folosită în rețele antrenate în mod supervizat și presupune domeniu de ieșire discret;

2. *Autoasociere* (*eng. clustering*) – rețeaua caută asemănări între mărimile de la intrare și le grupează în funcție de trăsăturile similare pe care le au în comun. Se mai numește și „clasificare nesupervizată", rețeaua fiind antrenată în mod nesupervizat;

3. *Predicție* – rețeaua funcționează similar unui filtru adaptiv în această configurație, astfel încercând să estimeze la ieșire eșantionul următor pe baza secvenței de eșantioane în timp de la intrare;

4. *Aproximare funcțională* (regresie) – este estimată o funcție necunoscută din perechile de date intrare–ieșire și presupune un domeniu de ieșire continuu.

5. *Scoatere de sub zgomot* – similar sistemelor tradiționale, se încearcă obținerea la ieșire a unui semnal cu un zgomot redus dintr-un semnal afectat de zgomot dat la intrare;

6. *Control* – este creat un model de referință ce semnifică starea curentă iar rețeaua transformă răspunsul dorit al sistemului de control într-o secvență de comandă pentru o evoluție corectă a sistemului;

7. *Optimizare* – este produs un set de valori (ponderi) pentru care diferența dintre ieșirea dorită și ieșirea rețelei este minimă, rezolvând astfel tipul de problemă ce constă în minimizarea sau maximizarea unei funcții obiectiv.

În mod general, fiecare element de prelucrare calculează un produs scalar al tuturor valorilor conexiunile sale de intrare și produce o singură valoare la ieșire. Această combinație liniară a valorilor de intrare x_i cu ponderile w_{ij} este urmată de o funcție de obicei neliniară f, numită funcție de activare:

$$y_j = f(\sum_{i=0}^{N} x_i w_{ij}) = f(x w_j^T + b) \tag{1.12}$$

unde $x = (1; x_1, x_2, ..., x_N)$ este vector linie, $w_j^T = (w_{0j}, w_{1j}, ..., w_{Nj}; 1)^T$ este vector coloană, iar N este numărul de intrări asociate neuronului j. De multe ori este introdusă și o polarizare (*eng. bias*) b, ce va reprezenta componenta constantă la intrarea neuronului.

Datorită faptului că un neuron poate fi reprezentat de un sumator (partea liniară) și o funcție de activare (transformare neliniară), o rețea neurală va deveni o compunere de funcții. Totuși, performanțele acesteia depind foarte mult de tipul de funcții folosite.

Alegerea funcției de activare potrivită depinde de cerințele problemei. Aceste funcții reprezintă practic filtre ce procesează informația transmisă și ele au rolul de a mărgini rezultatele într-un gamă controlabilă de valori. Din acest motiv, ne dorim un set de funcții cu proprietăți specifice, deoarece de ele depinde modificarea ponderilor pentru învățarea rețelei neurale.

Funcția cea mai simplă de activare este funcția liniară, în care nu se aplică nicio transformare. Rețelele astfel obținute sunt foarte ușor de antrenat, dar nu sunt capabile să modeleze funcții complexe. În general, aceste funcții liniare pot fi folosite în rețele ce predic o cantitate (probleme de regresie).

Funcțiile neliniare populare sunt *sigmoid* și *tangenta hiperbolică*. Funcția *sigmoid* (numită și funcție logistică) este o funcție continuă, monoton crescătoare și diferențiabilă, care tinde asimptotic către valorile sale de minim și maxim, de obicei [0, 1]. Intrările cu valori foarte mari vor fi convertite în valoarea 1, iar valorile negative vor fi egalate cu 0. *Tangenta hiperbolică* (pe scurt *tanh*), are o formă similară de `S` precum funcția *sigmoid*, dar mărginită în intervalul [-1, 1]. Astfel, valorile negative vor fi mapate spre -1 iar valorile nule vor fi puse în vecinătatea valorii 0. Folosind funcția *tanh* rețeaua se antrenează mai ușor și se pot obține performanțe mai bune, întrucât *sigmoida* poate determina blocarea unei rețele la un anumit set de ponderi în timpul antrenării.

O problemă generală a funcțiilor *sigmoid* și *tanh* este faptul că aceste funcții se saturează, adică valorile mari sunt trecute ca 1 iar valorile mici sunt mapate ca -1 sau 0. Mai mult, aceste funcții sunt cu adevărat sensibile la schimbări în jurul punctului lor mijloc, precum 0.5 pentru *sigmoid* și 0 pentru *tanh*. Datorită acestor lucruri, devine dificil pentru algoritmul de învățare să continue adaptarea ponderilor în scopul îmbunătățirii performanței rețelei neurale. Din aceste motive, cea mai folosită funcție de activare în acest moment este funcția *ReLU* (Rectified Linear Unit), cu o gamă de valori posibile [0, $+\infty$]. Deși s-a rezolvat problema saturației, un dezavantaj al acestei funcții este faptul că toate valorile negative de la intrare vor fi mapate în valoarea 0, ceea ce scade abilitatea de învățare a rețelei neurale.

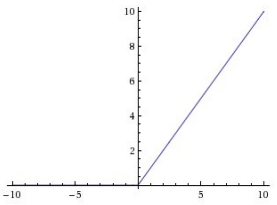
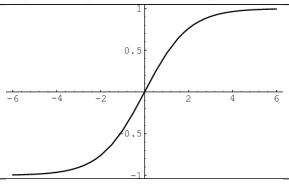
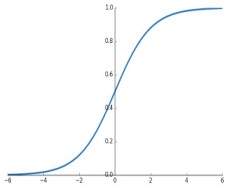
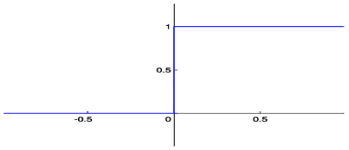
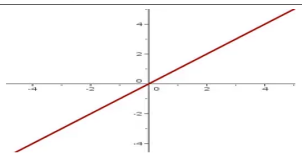
Denumire	Formulă	Formă
ReLU	$f(x) = max(0, x)$	
Tangentă hiperbolică	$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$	
Sigmoid	$f(x) = \frac{1}{1+e^{-x}}$	
Treaptă	$f(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}$	
Liniară	$f(x) = \lambda x$	

Tabela 1.3: Funcții de activare uzuale

Rolul acestor funcții prezentate în tabelul 1.3 este de a introduce o neliniaritate (cu excepția funcției liniare) în funcționalitatea rețelei, condiție necesară pentru a putea modela funcții complexe. Elementele din interiorul aceluiași strat se comportă identic (utilizează aceeași funcție de activare). În cazul unei rețele multistrat, se pot realiza conexiuni care „sar" un număr de straturi.

Avantajul acestor funcții este faptul că sunt derivabile, ceea ce face posibilă utilizarea unor tehnici de învățare bazate pe micșorarea gradientului (*eng. gradient descend*) în rețele cu mai multe straturi. Derivata erorii este propagată prin rețea pentru modificarea ponderilor și descrește cu fiecare nou strat pe care îl parcurge datorită derivatei funcției de activare folosite (*eng. vanishing gradient problem*), ceea ce îngreunează învățarea eficientă în rețelele multistrat.

Tehnica bazată pe micșorarea gradientului funcției cost este cea mai comună metodă de optimizare. În micșorarea gradientului, un lot (*eng. batch*) reprezintă numărul de exemple folosite simultan pentru învățarea rețelei. Un lot prea mare cu exemple luate la întâmplare conține date redundante. Mai exact, redundanța crește în funcție de dimensiunea lotului. Unele redundanțe sunt utile pentru a elimina gradienți zgomotoși, dar în general dimensiunea lotului nu trebuie să fie foarte mare. Optimizatorul *SGD* (Stohastic Gradient Descend) folosește un singur exemplu ales la întâmplare din lot-ul curent pentru a estima gradientul pe întregul lot, micșorând astfel redundanța și timpul de calcul necesar.

Deoarece acest *SGD* are probleme în găsirea minimului global pentru funcții cost ce au o curbă mult mai abruptă într-o dimensiune decât în altele, caz în care optimizatorul va oscila pe panta respectivă, este nevoie de un mod de a accelera optimizatorul în direcția potrivită și de a amortiza oscilațiile, numit „impuls" (*eng. momentum*) și se bazează pe introducerea

valorilor sale anterioare în calculul parametrului ce trebuie actualizat. S-au realizat mai multe versiuni de astfel de optimizatori bazați pe impuls, cel mai utilizat fiind *Adam* (Adaptive Moment Estimation), ce folosește atât valori anterioare ale parametrului respectiv, cât și valori anterioare ale gradientului calculat.

S-a demonstrat că se poate obține orice tip de transformare neliniară folosind o rețea cu un singur strat intermerdiar, cu un număr suficient de mare de neuroni.

Datorită numeroaselor modalități de conectare a neuronilor într-o rețea, în funcție de tipul de arhitectură, rețelele se împart în două mari categorii:

(A) *Rețele unidirecționale* (statice) – reprezintă sisteme fără memorie, în care nodurile sunt aranjate în straturi, fiecare nod primind semnale de la nodurile din stratul anterior, sau din spațiul de intrare. Acest tip de rețele calculează un răspuns la ieșire pentru un anumit set de date la intrare, fiind foarte util în rezolvarea problemelor de clasificare, aproximări de funcții, recunoașterea de date, predicție și control. Un exemplu de astfel de rețea (*eng. fully connected*) este prezentat în figura 1.4:

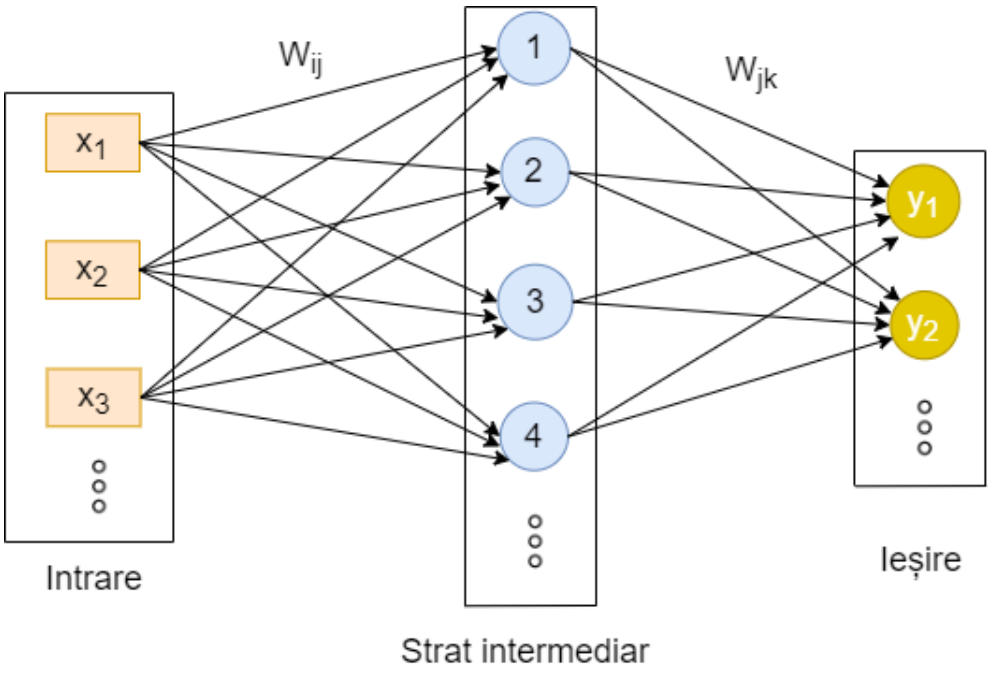


Figura 1.4: Rețea unidirecțională total conectată

Identificăm astfel faptul că o rețea are cel puțin 2 straturi: stratul de intrare (numit de obicei „pasiv" pentru că nu participă la procesul de învățare) și stratul de ieșire. Între aceste două straturi pot exista straturi intermediare, numite straturi „ascunse" (*eng. hidden layers*) și împreună cu stratul de ieșire acestea sunt straturi „active", fiind formate din neuroni. Se adoptă convenția ca nodurile de intrare să nu fie numărate ca un strat efectiv. Mărimile w_{ij} și w_{jk} reprezintă ponderile conexiunilor între straturi, primul indice reprezentând neuronul de la care „pleacă" ponderea, iar cel de-al doilea indice este neuronul spre care „vine" ponderea. Aceste ponderi sunt ajustate în momentul învățării, conexiunile cu valori pozitive fiind numite „excitatorii", iar cele cu valori negative „inhibitorii". Informația este prelucrată local de fiecare unitate de prelucrare, fară a se cunoaște starea celorlalte elemente. Numărul de conexiuni (deci straturi) poate fi foarte mare, rețelele fiind astfel numite „adânci" (*eng. Deep Neural Networks*, pe scurt, *DNN*), mărind complexitatea calculelor, dar în acest mod se pot rezolva probleme mai avansate.

(B) *Rețele recurente* (cu reacție) – reprezintă sisteme dinamice, sunt cele mai complexe, iar pentru fiecare stare de intrare se caută starea de echilibru. Proprietățile lor dinamice sunt descrise de sisteme neliniare diferențiale și sunt utilizate de obicei în modelarea sistemelor, predicția neliniară sau în probleme de control și optimizare. Un exemplu de astfel de rețea este în figura 1.5:

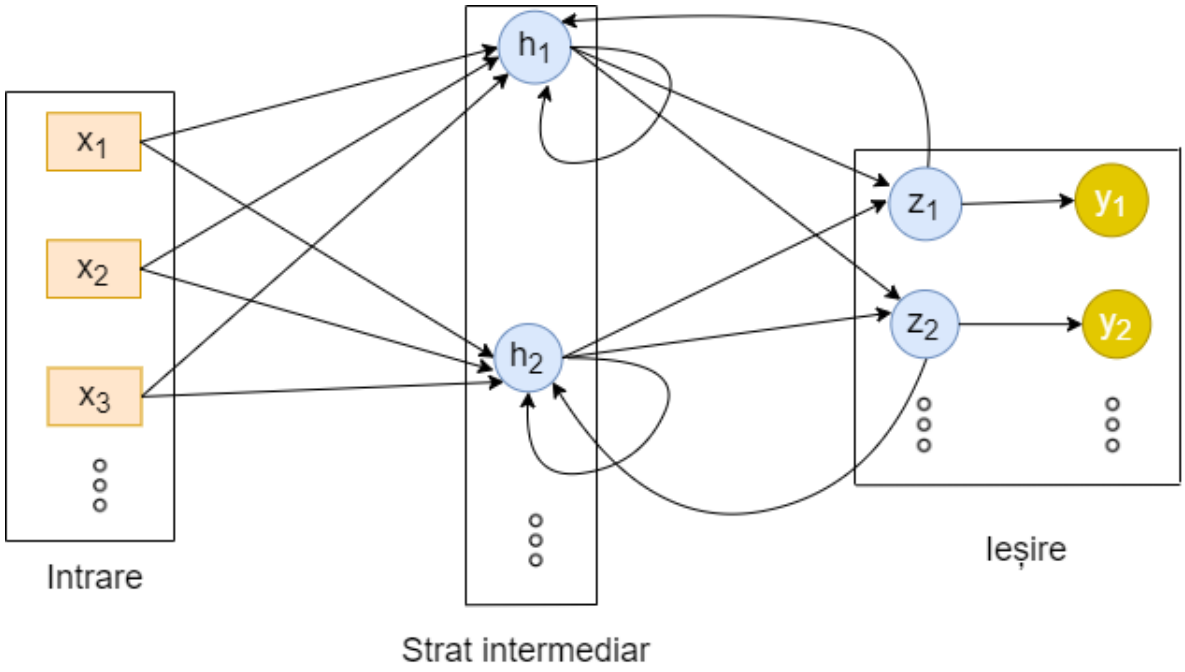


Figura 1.5: Rețea recurentă

Din punct de vedere al modului de învățare al rețelelor, distingem trei categorii:

1. *Învățare supervizată* (*eng. supervised learning*) – presupune existența unor valori dorite („etichete") pentru fiecare neuron din stratul de ieșire al rețelei. Astfel, sistemului îi este furnizat un set de perechi intrare–ieșire cu ajutorul căruia se calculează eroarea în funcție de rezultatul real obținut și cel dorit (învățare cu corectarea erorii). Se minimizează această eroare prin ajustarea valorilor ponderilor și a polarizării introduse.

2. *Învățare nesupervizată* (*eng. unsupervised learning*) – rețeaua extrage singură anumite caracteristici importante ale datelor de la intrare și formează reprezentări interne ale acestora. În acest caz, rețelele nu beneficiază de date de ieșire dorite pentru a evalua performanța, deci nu au în timpul antrenării informații despre ce înseamnă un răspuns corect sau greșit. În schimb, rețelele utilizează o „competiție" între neuronii elementari (învățare competitivă) pentru a modifica ponderile aferente neuronului care „câștigă" lupta, restul conexiunilor fiind neafectate.

3. *Învățare mixtă* (*eng. reinforcememt learning*) – combină învățarea supervizată și nesupervizată: o parte din ponderi sunt determinate prin intermediul unei învățări supervizate, iar restul sunt obținute pe baza unei învățări nesupervizate. În această situație, rețeaua nu beneficiază de semnalul dorit, ci de un semnal ce oferă o informație calitativă asupra funcționării sistemului (informație binară, de tipul răspuns corect / greșit), astfel sistemul este încurajat să producă acțiunea care duce la un rezultat corect.

După învățare, scopul final al unei rețele este să generalizeze ceea ce a învățat pe un set extins de date, cu anumite caracteristici comune ale setului de antrenare. Trebuie evitat fenomenul de „supraînvățare" (*eng. overfitting*), adică procesul prin care o rețea găsește legături specifice setului de antrenare dar nu poate generaliza deloc pe un set extins de date.

1.5 Starea artei (State of the art)

Transcrierea automată a muzicii (*eng. Automatic Music Transcription*, pe scurt, *AMT*), se referă la procesul automat prin care sunt identificate evenimente muzicale într-un semnal audio și sunt convertite în notații muzicale, fie într-un „piano-roll" (evoluția înălțimilor și duratelor notelor este prezentată în timp), fie într-un portativ (*eng. staff*) [9].

Pentru a putea realiza transcrierea automată a notelor muzicale, este necesară identificarea proprietăților muzicale precum frecvența notelor (pitch), durata și intensitatea acestora, timbrul instrumentului, armonicile spectrului, măsura în care este cântată piesa, cheia acesteia, etc. Astfel, *AMT* se ocupă cu identificarea diferitelor atribute ale semnalului audio dat, față de generarea semnalului în funcție de condiții date despre aceste atribute. Formatul `MIDI` este potrivit pentru codarea acestor proprietăți și poate fi decodat de programele uzuale folosite în muzică, deci acesta va fi formatul de ieșire folosit pentru un sistem *AMT*.

În funcție de modul în care sunt cântate notele într-o piesă, putem face următoarea clasificare:

- *Melodii monofonice* – reprezintă melodiile în care este folosit un singur instrument și este cântată o singură notă la un moment de timp, notele fiind deci separate. Acesta este tipul de melodii folosite în acest proiect.
- *Melodii polifonice* – semnifică melodiile uzuale, fiind formate din mai multe instrumente iar notele sunt simultan cântate, deci suprapuse fie de la același instrument, fie de la instrumente diferite. Aceste melodii sunt cele mai dificile de transcris în domeniul *AMT* și încă nu există o soluție generală sau suficient de bună în comparație cu acuratețea umană.

Datorită complexității și dificultății realizării unui sistem de transcriere *End-to-End* doar din semnalul audio [10], multe abordări presupun împărțirea în diferite subsarcini, precum clasificarea notelor, extragerea de onset-uri și offset-uri, estimarea intensității, recunoașterea timbruluui și a instrumentelor. Fiecare subsarcină saparată poate avea interesante aplicații în afara transcrierii muzicii End-to-End, ele fiind clasificate ca subprobleme în domeniul recuperării informațiilor muzicale (*eng. Musical Information Retrieval*, prescurtat *MIR*).

Deoarece dimensiunea unui semnal audio $x(n)$ este destul de mare, fiind uzual folosită o frecvență de eșantionare de 44.1kHz, se dorește reducerea dimensiunii prin extragerea unor trăsături caracteristice. Din acest motiv, se lucrează în mod general cu ferestre de semnal $w(n)$ de 10 – 50ms, eventual suprapuse cu un anumit procent, din care sunt extrase trăsături precum transformata Fourier (short-time Fourier transform *STFT*), sau spectrograma, folosind formulele:

$$\mathbf{STFT}\{x(n)\}(m,\omega) = \sum_{n=-\infty}^{+\infty} x(n)w(m-n)e^{-j\omega n} \tag{1.13}$$

$$\mathbf{Spectrogram}\{x(n)\}(m,\omega) = \mid \mathbf{STFT}\{x(n)\}(m,\omega) \mid^2 \tag{1.14}$$

Aceste funcții oferă foarte multe informații în frecvență, dar datorită dimensiunii încă mari se vor extrage din ele anumiți parametrii, precum coeficienții *MFCC* (Mel-Frequency Cepstral Coefficients) [11], inspirați din domeniul vocal, prin aplicarea unor bancuri de filtre Mel și selectarea primelor componente *DCT* (Discrete Cosine Transform) ce conțin factori independenți ce descriu forma spectrală.

O altă transformată, *CQT* (Constant-Q Transform) [12], folosește bancuri de filtre în care frecvențele centrale ale acestor filtre au un factor Q constant, care este raportul dintre frecvența

centrală și lățimea de bandă de 3dB a unui filtru. Prin modificarea acestei transformate pentru a produce 12 filtre per octavă, se pot obține coeficienți ce corespund fiecărui ton muzical și se obține o reprezentare numită „cromagramă" [13].

Pentru extragerea informațiilor referitoare la măsură sau tempo, respectiv apariția unei note, se aplică o funcție precum diferența de ordinul întâi a funcției energetice logaritmice în domeniul timp sau a fluxului spectral ce măsoară variația de energie a spectrului de frecvență. Se obține astfel o curbă de noutate (*eng. novelty curve*) ce arată vârfuri de energie, apărute în general în momentul apariției unei note [14]. Aceste momente de apariție a notelor pot fi folosite pentru a obține informații legate de tempo, folosind o reprezentare numită „tempogramă" [15], [16].

Aceste tehnici de extragere a trăsăturilor caracteristice semnalului audio sunt ilustrate în figura 1.6 [9]:

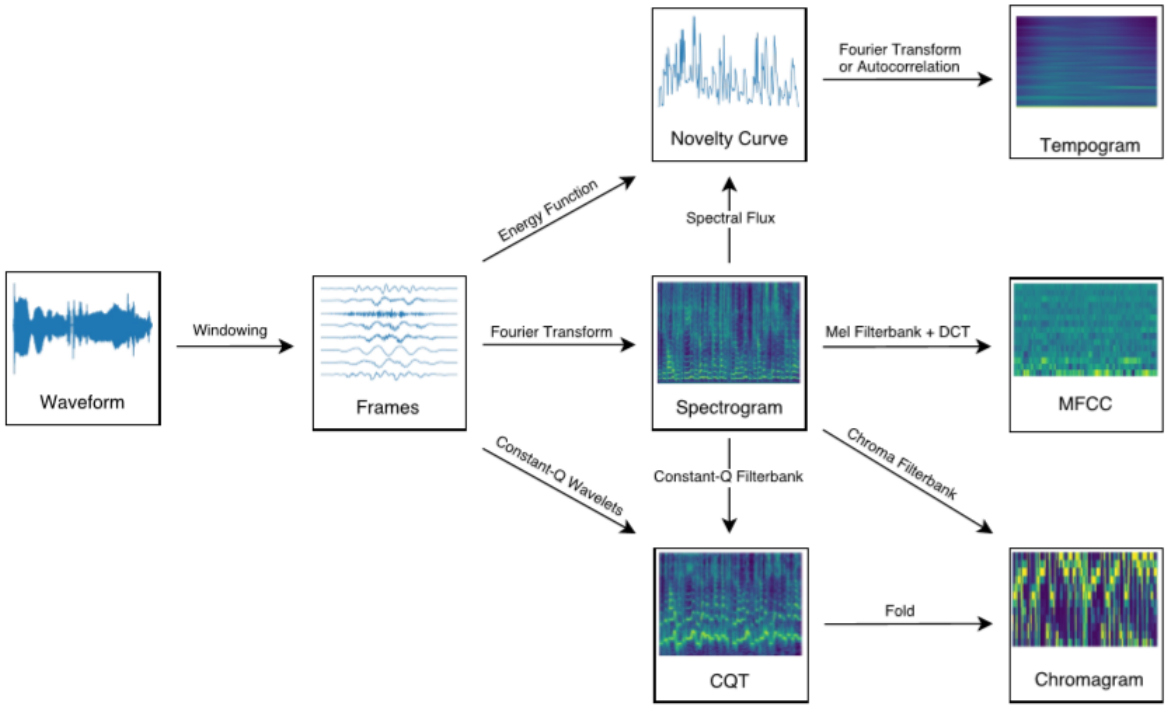


Figura 1.6: Prelucrarea generală a semnalului audio pentru extragerea trăsăturilor muzicale

Recent, au fost înlocuite câteva din aceste transformări în domeniul *MIR*, fiind folosite modele bazate direct pe spectrogramă sau semnalul audio. Aplicații de învățare profundă (*eng. deep learning*) sunt folosite pentru rezolvarea acestor sarcini, inclusiv clasificarea notelor melodiei [17], identificarea tempo-ului [18], clasificarea genului de muzică [19], oferind performanțe superioare în comparație cu abordări bazate pe extragerea de trăsături. În afară de câteva astfel de abordări, majoritatea modelelor ce folosesc învățarea adâncă se bazează încă pe transformate precum *STFT* sau *CQT*, datorită faptului că aceste trăsături oferă informații despre armonicitatea semnalului audio și este mai ușor pentru o rețea neurală să învețe aceste concepte fără a se produce fenomenul de supraînvățare. Totuși, extragerea de trăsături are ca efect o pierdere de informație, din acest motiv modelul cu cele mai bune performanțe ar beneficia cel mai mult de semnalul audio nemodificat, dacă sunt suficiente date de antrenare și există un sistem hardware ce le poate procesa [20].

Pentru a îmbunătăți performanțele rețelelor neurale, diverse cunoștințe muzicale sunt aplicate. Se pot face astfel presupuneri, precum faptul că schimbările bruște în muzică sunt rare și

majoritatea se petrec gradual. Se poate aplica deci o filtrare mediană [21] pentru a suprima modificările bruște, iar modelele ascunse Markov (*eng. Hidden Markov Models*, pe scurt, *HMM*) sunt folosite pe scară largă pentru a modela secvențe de date precum acordurile [22], dar și pentru a netezi secvențele de ieșire ca o etapă de postprocesare [23]. Pentru durata unei note, există abordări ce detectează onset-ul și offset-ul notei pentru a o putea transcrie [24], sau se modelează atacul și degradarea unei note [25], precum și generalități ale evoluției temporale a notei [26]. În domeniul frecvență, principiul netezirii spectrale estimează faptul că anvelopa spectrală a unor sunete reale variază ușor în funcție de frecvență [27]. Acest principiu a fost implementat în diverse modalități, precum un filtru mediu în mișcare pentru estimarea și separarea iterativă a surselor [27], o funcție de scor pentru candidatul frecvenței fundamentale F0 [28], precum și în modelarea autoregresivă de ordin redus a tonurilor suprapuse [29].

Pentru melodiile monofonice, au fost aplicate diverse metode pentru extragerea frecvenței fundamentale. Astfel, pentru estimarea pitch-ului, au fost introduse funcții precum *cepstrul* [30], funcția de autocorelație (*ACF*) [31], funcția diferenței de amplitudini medie (*AMDF*) [32], funcția de corelație încrucișată normalizată (*NCCF*), propusă în [33], sau [34], precum și funcția de diferență medie normalizată cumulată (metoda *YIN*), propusă în [35]. Aplicații mai recente includ metoda *SWIPE* (Sawtooth Waveform Inspired Pitch Estimator) [36], care realizează o aproximarea a frecvenței fundamentale folosind spectrul unui forme de undă „dinte de ferăstrău", sau metoda *pYIN* [37], ce reprezintă o variantă probabilistică pentru metoda *YIN*, utilizând modele *HMM* pentru a decoda cea mai probabilă secvență de valori ale notelor. Potrivit unor studii, în cazul melodiilor monofonice rezultatele cele mai bune sunt obținute folosind metodele *YIN* [38], [39], metoda *pYIN* având cele mai bune perfomanțe. Deoarece majoritatea metodelor monofonice de extragere a frecvenței fundamentale se bazează pe faptul că o singură notă este cântată la un moment de timp, acestea nu pot fi aplicate direct pentru studiul melodiilor polifonice, fiind necesare alte abordări pentru estimarea frecvențelor multiple.

Capitolul 2
Setup experimental

2.1 Baze de date

În domeniul transcrierii automate a muzicii, pentru o bază de date sunt necesare minim următoarele fișiere:

- Un fișier în format `MIDI`, ce va conține detalii despre notele cântate și modul în care au fost înregistrate;
- Un fișier `WAV`, ce reprezintă semnalul audio propriu-zis;
- Un fișier text, în care vor fi reprezentate toate notele cântate și timpul la care fiecare notă va începe, respectiv se va termina.

2.1.1 Baza de date inițială

Inițial a fost folosită o bază de date publică [40], citată de mai mulți autori în articole legate de transcrierea automată a notelor muzicale. Această bază de date conține înregistrări la pian, fiind creată pentru reconstituirea notelor muzicale și evaluarea algoritmilor de estimare pentru melodii monofonice și polifonice. Baza de date oferă o cantitate mare de sunete obținute în diverse condiții de înregistrare. Înregistrările au fost eșantionate la o frecvență $f_e = 44.1kHz$.

Pentru acest proiect a fost folosită doar o parte din baza de date prezentată, mai exact doar înregistrările ce conțineau melodii monofonice, deci note izolate (secțiunea *IZOL*). Astfel, sunt utilizate 3909 de înregistrări (împărțite în 2645 de piese pentru antrenare, 625 pentru validare și 639 pentru testare), obținând aproximativ 5h și 14 min de semnal audio. Sunt utilizate 88 de note, în intervalul `A0` – `C8`, obținându-se astfel gama de frecvențe [27.5Hz – 4186Hz]. O notă poate dura până la 2.21s, sau pot fi cântate până la 32 de note pe secundă.

Durata minimă a unei note într-o bază de date are o semnificație specială. Astfel, deoarece cunoaștem faptul că:

$$T = \frac{1}{f} \tag{2.1}$$

putem spune că perioada reprezintă intervalul minim de timp după care semnalul se repetă identic. Dacă ar fi să calculăm perioada necesară pentru cea mai mică frecvență a primei baze de date, vom obține:

$$T_1 = \frac{1}{f_1} = \frac{1}{27.5Hz} = 36.36ms \tag{2.2}$$

Dar, deoarece în baza de date există note cu durate mai mici decât această valoare (durata minimă gasită este de 14ms), este foarte greu de aproximat frecvența fundamentală pentru aceste note cu durata mai mică decât perioada. Aceasta valoare este mult prea mică și în realitate nu sunt niciodată cântate note cu o astfel de durată.

În urma experimentelor, deoarece pentru clasificarea notelor nu s-a ajuns la o acuratețe similară cu ce există deja prezentat în starea artei (peste 95%), datorită diverselor condiții ale înregistrărilor audio, precum și a faptului că există note de o durată mai mică (14ms) decât durata minimă necesară pentru determinarea notei, s-a decis crearea unei noi baze de date, proprie, fiecare înregistrare fiind făcută sub anumite restricții.

2.1.2 A doua bază de date

Această bază de date, disponibilă online [1] a fost concepută special pentru acest proiect de diplomă, cu scopul de a arăta că transcrierea automată a muzicii este posibilă prin utilizarea unui instrument real, anume orga. Astfel, înregistrările au fost făcute în format `MIDI`, prin intermediul unei orgi Roland, cu 5 octave (gama de note `C2` – `C7`), conectată direct la un laptop. Baza de date constă în 100 de exerciții muzicale și melodii scurte, toate înregistrările fiind monofonice, adică o singură notă este cântată la un moment de timp.

În continuare vor fi prezentate restricțiile acestei baze de date:

- Gama de note posibile este `C2` – `C7`, deci 61 de note, față de cele 88 de note `A0` – `C8` pentru setul prezentat anterior. Această limitare nu a fost facută intenționat, ci s-a datorat faptului că modelul de orgă utilizat dispunea de doar 5 octave. Acest lucru reprezintă un avantaj, deoarece se va restrânge numărul de clase posibile la 61, aceste note fiind oricum în realitate cele mai utilizate, domeniul extins de 88 de note fiind mai rar utilizat;

- Gama de frecvențe este [65.4Hz – 2093Hz], în comparație cu [27.5Hz – 4186Hz] a primei baze de date. Aceasta este o consecință a faptului că gama de note este mai restrânsă;

- Durata minimă a unei note este de 93ms, iar durata maxima de 3s. La prima bază de date s-a constatat că durata minimă este de 14ms, iar cea maximă de 2.21s.

Valoare duratei minime nu a fost aleasă arbitrat. Astfel, dacă considerăm un tempo normal de 120 BPM obținem:

$$1beat = \frac{60s}{120} = 0.5s \tag{2.3}$$

Deci, pentru un ritm normal de 4 / 4, o pătrime va fi echivalentul unei măsuri (beat):

$$1\text{pătrime} = 1\text{beat} = 0.5s \tag{2.4}$$

Astfel, dacă considerăm faptul că pentru o melodie normală durata minimă ar fi șaisprezecimea (lucru normal în melodiile uzuale), vom obține o valoare minimă de:

$$1\text{șaisprezecime} = \frac{1\text{pătrime}}{4} = \frac{0.5s}{4} = 125ms \tag{2.5}$$

Când a fost creată noua bază de date, au fost înregistrate note cu valori mai mici decât limita minimă de 125ms, pentru a acoperi eventuale diferențe de durată din cauza interpretării sau

[1] `https://speed.pub.ro/downloads/`

întârzieri date de cablul USB în momentul înregistrării în format `MIDI`. Astfel, s-a obținut valoarea minimă de 93ms.

Deoarece această bază de date a fost creată special pentru învățarea pe baza rețelelor neurale, unele melodii au fost repetate în diferite octave, pentru a obține un număr minim de apariții pentru toate notele, lucru important pentru identificarea numărului de clase aferent rețelei.

Astfel, au fost obținute 100 de înregistrări, formând un total de aproximativ 1h și 17 min de semnal audio.

Baza de date a fost împărțită în următoarele seturi:

- *setul de antrenare*: reprezentat de 90 de fișiere, având 1h și 7 min de semnal audio, dimensiunea înregistrărilor variind între 6s și 150s;
- *setul de evaluare*: constă în 10 fișiere, reprezentând aproximativ 10 min de semnal audio, cu înregistrări de dimensiunea între 12s și 120s.

Pentru testare, a fost înregistrată o melodie suplimentară de 40s.

Toate înregistrările au fost făcute în condiții normale, fără zgomot, folosind o frecvență de eșantionare de $f_e = 44.1kHz$. Piesele au fost înregistrate în format `MIDI`, folosind programul `MidiEditor`[2] în următoarea manieră: s-a activat modul de înregistrare din `MidiEditor` înainte ca melodia să fie cântată, urmând ca înregistrarea să fie oprită dupa ce piesa s-a terminat (deci va fi un moment de liniște la începutul și la sfârșitul înregistrărilor). Ca orice bază de date muzicală, fiecărui fișier `MIDI` îi trebuie asociat un fișier `.wav` și un fișier `.txt`. Fișierul `.wav` a fost realizat prin conversia fișierului `MIDI`, folosind un script în python ce utilizează `FluidSynth` prin librăria `midi2audio`[3]. Fișierul text a fost creat prin citirea fișierului `MIDI` în limbajul `Python` utilizând librăria `midi`[4] și extragerea informațiilor în legătură cu apariția notelor, precum și înălțimea acestora. Acest fișier text conține deci pentru fiecare notă momentul în timp în care aceasta a fost apăsată (onset) și momentul în care aceasta s-a oprit (offset), iar fiecare notă este reprezentată printr-un număr `MIDI`. Formatul acestui fișier text este următorul:

```
Onset [s]     Offset [s]     Notă [număr MIDI]
```

2.2 Preprocesarea semnalului

Prelucrarea digitală a semnalelor constă în reprezentarea semnalelor ca secvențe ordonate de numere și prelucrarea acestora în scopul estimării anumitor parametrii caracteristici (deci extragerii unei anumite informații), eliminării sau reducerii unor componente nedorite, sau în scopul transformării unui semnal într-o formă care să fie mai semnificativă din anumite puncte de vedere [1].

Deoarece bazele de date sunt deja în format digital, nu trebuie să ne punem probleme legate de conversia analog-digital.

Întrucât ambele baze de date conțin semnale audio stereo (cu 2 canale), pentru a lucra mai ușor, primul pas de preprocesare este convertirea semnalului într-un semnal mono (cu un singur canal), prin medierea valorilor celor două canale, eșantion cu eșantion.

---

[2] `https://www.midieditor.org/`

[3] `https://pypi.org/project/midi2audio/`

[4] `https://github.com/louisabraham/python3-midi/`

O altă metodă de preprocesare folosită este normarea semnalului pentru a obține aceeași gamă de valori ale amplitudinilor, în intervalul [0, 1]. Acest lucru se realizează prin împărțirea fiecărui eșantion la maximul de amplitudine al semnalului audio. Această etapă este importantă deoarece o rețea neurală va învăța mai ușor pentru un set de date cu aceeași gamă de valori, fiind astfel capabilă să diferențieze mai rapid semnalele.

Pentru o dimensiune mai mică a semnalului audio, deci un număr mai mic de eșantioane care să-l reprezinte, este de dorit ca un semnal să nu fie supraeșantionat. Se cunoaște faptul că frecvența de eșantionare trebuie să fie minim egală cu dublul frecvenței maxime a semnalului audio. Deoarece avem o frecvență de eșantionare $f_e = 44.1kHz$ pentru ambele baze de date, iar frecvențele maxime sunt: $f_{max1} = 4186Hz$ (pentru prima bază de date), respectiv $f_{max2} = 2093Hz$ (pentru a doua bază de date), teoretic, reducerea frecvenței de eșantionare a semnalelor, este posibilă. Dar, deoarece armonicele notelor sunt foarte importante pentru diferențierea între note, practic, reeșantionarea este posibilă doar pentru cea de-a doua bază de date, la o nouă frecvență de eșantionare $f_e^{'} = 16kHz$.

Semnalele muzicale reale sunt, de obicei, aperiodice, de durată mult mai mare decât intervalele de analiză uzuale din aplicațiile inginerești. Din acest motiv, pentu a putea aplica legile statistice cunoscute pentru semnale staționare și periodice se va analiza semnalul pe un interval finit de timp.

Analiza semnalului pe un interval finit de timp

A extrage (selecta) dintr-un semnal de durată mare un segment de lungime finită reprezintă de fapt multiplicarea semnalului cu o secvență dreptunghiulară alcătuită din N eșantioane-unitate (ce poartă denumirea de fereastră dreptunghiulară), notată cu $w_R(n)$ și ilustrată în figura 2.1 [41]. Astfel, se obține un semnal de forma:

$$x_N(n) = \begin{cases} x(n), & \text{pentru } n = 0, 1, \ldots, N-1 \\ 0, & \text{în rest} \end{cases} \tag{2.6}$$

unde $x(n)$ este semnalul original, presupus aperiodic și de durată nelimitată.

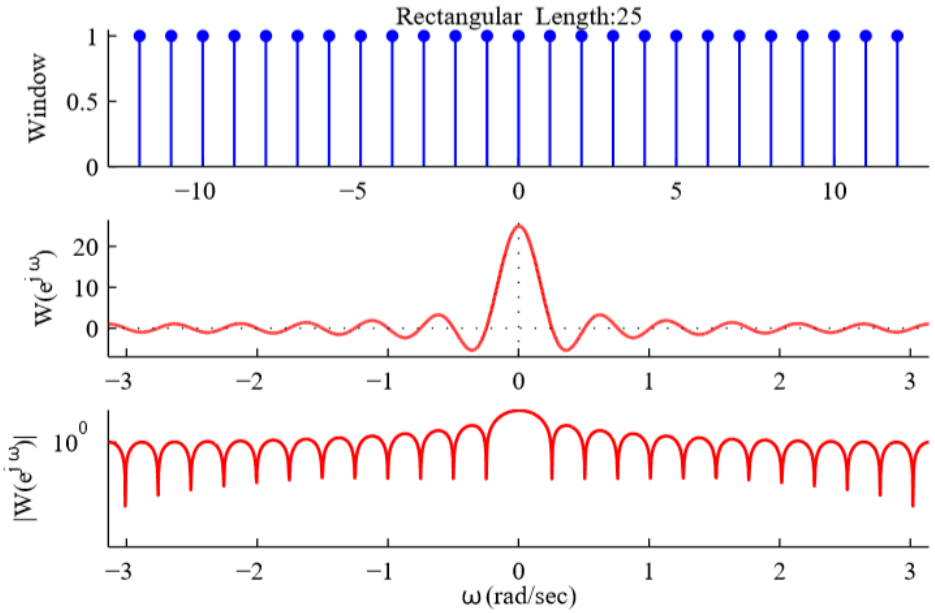


Figura 2.1: Fereastră dreptunghiulară

Aceasta se numește „tehnica ferestruirii" și este folosită uzual în majoritatea aplicațiilor, cu scopul de a obține semnale cvasistaționare de lungime finită, dintr-un semnal complex.

Multiplicarea în timp a două semnale discrete se va manifesta în domeniul frecvență printr-o convoluție:

$$x(n)w_R(n) \leftrightarrow X(e^{j\omega}) * W_R(e^{j\omega}) = X_N(e^{j\omega}) \tag{2.7}$$

unde $X(e^{j\omega})$, $W_R(e^{j\omega})$ și $X_N(e^{j\omega})$ sunt transformatele Fourier ale semnalului x(n), secvenței dreptunghiulare $w_R(n)$ și respectiv secvenței $x_N(n)$. Funcția $W_R(e^{j\omega})$ este numită și fereastră spectrală. Pentru o secvență dreptunghiulară aceasta este de forma unei funcții `sinc`, având forma unui lob central (cu lărgimea de bază $2 \times 2\pi/N$), înconjurat de lobi laterali mici, care vor descrește progresiv în amplitudine către extremitățile intervalului unei perioade.

Deci, spectrul real $X_N(e^{j\omega})$ va fi o convoluție între spectrul ideal $X(e^{j\omega})$ și o funcție de tipul `sinc`. Acest lucru determină o deformare a spectrului ideal, însoțită de apariția unor ondulații în spectrul semnalului $x_N(n)$. Acest fenomen se numește dispersie, sau împrăștiere a spectrului (*eng. leakage*).

Deoarece spectrul real $X_N(e^{j\omega})$ va fi „eșantionat" în frecvență pentru a obține coeficienții Fourier $X_N(k)$, vor apărea erori în forma acestui spectru discret și în consecință în forma semnalului refăcut din eșantioane. Aceste erori sunt modificări ale amplitudinilor coeficienților spectrali $X(k)$, numite erori de amplitudine (*eng. peaked-fence*), apariția unor frecvențe false (din cauza lobilor laterali din spectrul ferestrei dreptunghiulare), sau pierderea unei informații de frecvență (nu mai pot fi puse in evidență componente de frecvențe foarte apropiate din spectrul unui semnal complex).

Înmulțirea semnalului cu o fereastră dreptunghiulară este în esență o trunchiere „abruptă" a semnalului și are ca principale efecte în domeniul frecvență, o dispersie a componentelor spectrale ale semnalului și o modificare a amplitudinilor acestor componente. Aceste efecte pot fi interpretate în domeniul timp (datorită periodicității transformatei Fourier discrete inverse) precum apariția unor discontinuități la capetele intervalului de analiză.

Pentru a diminua toate aceste efecte, soluția constă în utilizarea unor funcții fereastră, cu o curbă mai „netedă" și cu un spectru care să permită micșorarea acestor efecte. Aceste funcții poartă denumirea de ferestre de ponderare (*eng. window weighting functions*). Astfel, se va realiza o trunchiere mai puțin abruptă, semnalul fiind adus către zero la capetele intervalului de N eșantioane. Spectrele acestor funcții vor fi formate dintr-un lob central, ce va conține cea mai mare parte din energia semnalului și din lobi laterali cu amplitudine descrescătoare către capetele intervalului.

Astfel, aplicarea unei funcții fereastră (alta decât cea dreptunghiulară) asupra unui semnal determină pierderea unei anumite cantități de informație în timp spre capetele intervalului și duce la obținerea unui spectru cu vârfuri mai largi și de amplitudine mai mică decât în cazul ferestrei dreptunghiulare, dar se atenuează mult lobii laterali și deci se reduce semnificativ fenomenul de dispersie a spectrului.

Pentru acest proiect, se va folosi pentru analiza în domeniul timp, fereastra dreptunghiulară (pentru a păstra semnalul nemodificat astfel încât să nu existe pierderi de informație), iar în domeniul frecvență, pentru transformata Fourier, se va folosi fereastra de ponderare de tip *Hamming* (pentru a reduce fenomenul de dispersie spectrală și erorile de amplitudine). Fereastra *Hamming* este prezentă în figura 2.2 [41] și are următoarea formulă:

$$w_{HAM}(n) = \begin{cases} \alpha - (1-\alpha)\cos\left(\frac{2\pi n}{N-1}\right), & \text{pentru} \quad 0 \le n \le N-1 \\ 0, & \text{în rest} \end{cases} \tag{2.8}$$

$$(\text{uzual, } \alpha = 0.54)$$

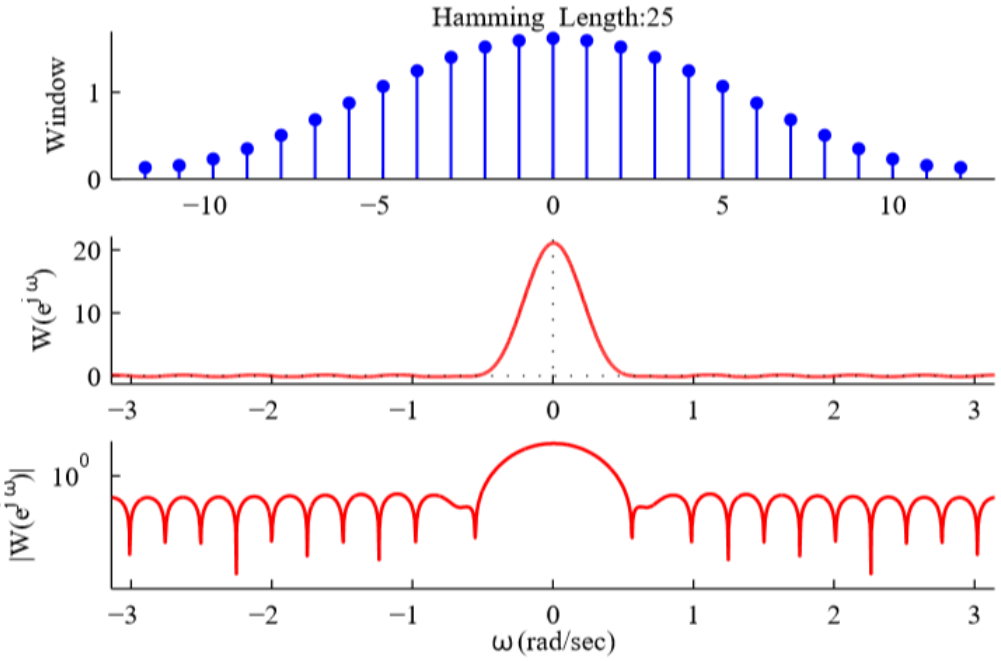


Figura 2.2: Fereastra Hamming

Această fereastră prezintă următoarele proprietăți:

- primul lob lateral este mult atenuat;
- lobii laterali următori au o descreștere de -6dB / octavă

2.3 Clasificarea notelor

Schema din figura 2.3 reprezintă modul în care clasificarea notelor este rezolvată.

Astfel, pentru clasificarea notelor, sunt parcurse următoarele etape:

1. Semnalul audio normat este împărțit în ferestre de semnal, utilizând tipul de fereastră potrivit;
2. Din fiecare fereastră de semnal sunt extrase trăsăturile dorite, fie în domeniul timp, fie în domeniul frecvență;
3. Se formează un vector de trăsături corespunzător tuturor semnalelor audio din setul de antrenare al bazei de date (se procedează similar și pentru setul de evaluare);
4. Pentru fiecare vector de trăsături (antrenare, respectiv evaluare), se va crea un vector de clase pentru a putea identifica notele cântate, numite „etichete” (*eng. ground-truth*);
5. Vectorul de trăsături pentru setul de antrenare va reprezenta vectorul de intrare în rețeaua neurală, iar vectorul de clase corespunzător va fi vectorul de ieșire dorit din această rețea;
6. Rețeaua se va antrena pe parcursul mai multor epoci, modificând valorile ponderilor astfel încât funcția cost folosită să atingă un minim global. Astfel, rețeaua va încerca să găsească o legătură cât mai bună între vectorul de intrare (trăsăturile) și vectorul de ieșire (nota dorită);
7. După antrenare, rețeaua va prezice un vector de clase ce va reprezenta notele cântate în piesa dorită.

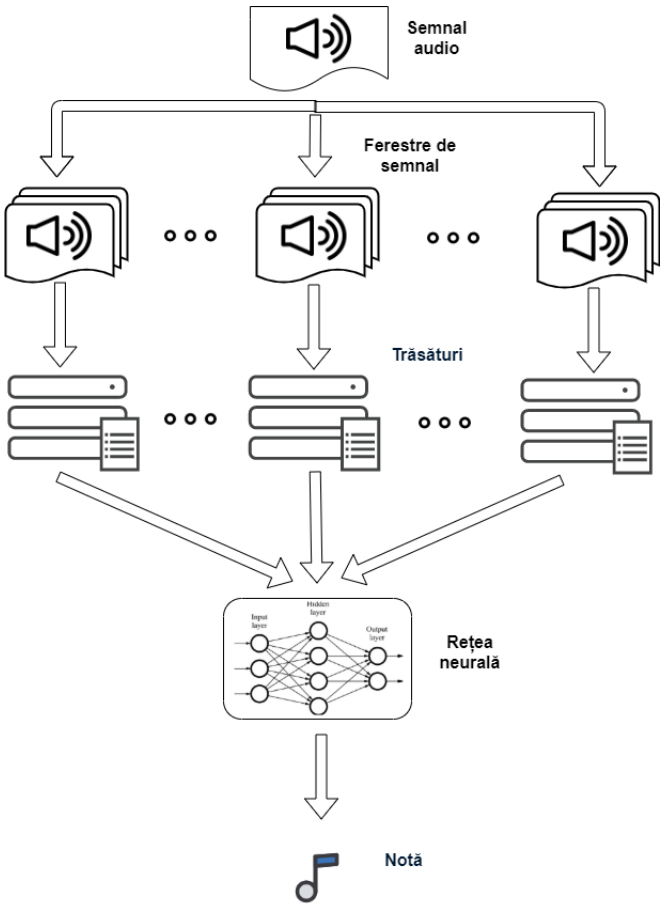


Figura 2.3: Clasificarea notelor

Trăsături utilizate

Domeniul timp

1. *Semnalul pur*

   Recent, este de dorit ca un semnal, indiferent de tipul acestuia, să fie introdus într-o rețea neurală fără a extrage parametrii care să-l reprezinte, urmând ca rețeaua să-și extragă singură ceea ce este necesar pentru a putea caracteriza semnalul dorit (rețele *End-to-End*). Din acest motiv și datorită simplității preprocesării, precum și a unui timp redus de calcul, prima și cea mai simplă „trăsătură" în timp a unui semnal folosită pentru acest proiect va fi chiar semnalul în sine, numit „semnal pur".

2. *Funcția de autocorelație*

   Deoarece un semnal muzical este un semnal aleator, acesta este definit la fiecare moment de timp t printr-o lege de probabilitate a amplitudinii sale. Această lege se poate exprima printr-o densitate de probabilitate $p(x,t)$:

$$p(x,t) = \lim_{\Delta x \to 0} \frac{Prob[x \leq s(t) \leq x + \Delta x]}{\Delta x} \tag{2.9}$$

   Astfel, pentru un semnal aleator $s(t)$ se lucrează cu metode statistice, deci cu o prelucrare statistică. Se definesc 3 mărimi de interes special [4]:

- *Valoarea medie* (moment de ordin 1):

$$m_1 = \mu_s = E[s(t)] = \int_{-\infty}^{+\infty} xp(x,t)dx \tag{2.10}$$

  unde $E[s(t)]$ reprezintă speranța matematică.

- *Momentul de ordin 2*:

$$m_2(t1,t2) = E[s(t_1)s(t_2)] = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} x_1x_2p(x_1,x_2;t_1,t_2)dx_1dx_2 \tag{2.11}$$

  Această mărime este o reprezentare probabilistică a corelației între perechi de valori pentru semnalul $s(t)$. În general, se mai numește funcție de corelație, dar în acest caz definește practic *funcția de autocorelație*, deoarece se analizează un singur semnal, $s(t)$.

- *Varianța* (moment centrat de ordin 2):

$$var(s(t)) = \sigma_s^2 = E[(s-\mu_s)^2] \tag{2.12}$$

  Dacă media semnalului este 0, varianța este practic identică cu puterea semnalului.

Un semnal aleator se numește *Staționar în Sens Larg* (*SSL*) dacă proprietățile statistice până la ordinul 2 inclusiv sunt independente de timp. Acest lucru implică faptul că valoarea medie a semnalului este constantă (nu depinde de timp), iar funcția de autocorelație depinde doar de 2 momente de timp $\tau = t_2 - t_1$. Deoarece vom lucra pe ferestre mici de semnal astfel încât semnalul devine cvasistaționar, îl vom considera semnal *SSL*.

Astfel, mărimile statistice devin:

$$m_1 = E[s(t)] = ct \tag{2.13}$$

$$m_2 = E[s(t)s(t+\tau)] = R_{ss}(\tau) \tag{2.14}$$

Funcția de autocorelație are următoarele proprietăți:

- Este o funcție reală și pară:

$$R_{ss}(\tau) = R_{ss}(-\tau); \tag{2.15}$$

- Maximul ei este în origine și corespunde cu puterea medie a semnalului:

$$R_{ss}(\tau) \leq R_{ss}(0) = E[s^2(t)] = P \tag{2.16}$$

- Autocorelația sumei a două semnale complet necorelate este suma autocorelațiilor pentru cele două semnale;
- Autocorelația unui semnal periodic este o funcție periodică cu aceeași perioadă ca cea a semnalului. Astfel, funcția de autocorelație prezintă maxime distincte descrescătoare în amplitudine la invervale de timp cu durata de $\frac{1}{f_0}$.

Ultimele două proprietăți sugerează posibilitatea evidențierii unor periodicități (deci frecvențe) sau cel puțin a unor evenimente repetitive într-un semnal aleator. Aceste proprietăți sunt motivul pentru care această funcție este aleasă ca o trăsătura ce poate caracteriza semnalul audio, deoarece se poate extrage frecvența fundamentală caracteristică fiecărei note pe baza periodicității semnalului. Lungimea ferestrei aleasă trebuie să

fie mică datorită variabilității semnalului audio, dar suficient de mare pentru a cuprinde cel puțin două perioade ale formei de undă, astfel încât să se poată obține informația de periodicitate.

Un semnal *SSL* este și ergodic dacă valorile medii statistice corespund cu cele temporale. Această ergodicitate a semnalelor are consecințe practice foarte importante deoarece oferă un mijloc de a avea acces la proprietățile statistice ale unui semnal aleator pornind de la observarea sa în decursul timpului.

Deoarece în realitate vom lucra cu mărimi temporale, redefinim mărimile prezentate mai sus:

$$m_x = \mu_x = \frac{1}{N}\sum_{n=0}^{N-1} x(n) \tag{2.17}$$

$$R_{xx}(l) = \frac{1}{N}\sum_{n=0}^{N-1} x(n)x(n+l) \tag{2.18}$$

unde N reprezintă lungimea ferestrei, iar ℓ întârzierea între eșantioane.

În realitate, funcția de autocorelație va fi calculată pe baza teoremei Wiener-Khinchin:

$$R_{xx}(l) = \frac{1}{2\pi}\int_{-\pi}^{+\pi} S_{xx}(e^{j\omega})e^{j\omega}d\omega \tag{2.19}$$

unde $S_{xx}(e^{j\omega})$ reprezintă densitatea spectrală de energie pentru un semnal de energie finită și arată cum e distribuită energia (puterea) în frecvențe. Aceasta are formula:

$$S_{xx}(e^{j\omega}) =\mid X(e^{j\omega}) \mid^2 \tag{2.20}$$

unde $X_N(e^{j\omega})$ este transformata Fourier pentru semnalul $x(n)$.

Astfel, funcția de autocorelație va fi calculată folosind transformata Fourier inversă a densității spectrale de energie pentru un semnal de energie finită.

Domeniul frecvență

1. *Transformata Fourier*

   Analiza în frecvență consideră semnalul ca o superpoziție de sinusoide și permite în acest mod extragerea unor parametrii greu de pus în evidență în domeniul temporal. Calculul spectrului de frecvențe al semnalului înseamnă de fapt examinarea directă a informației codificate în frecvența, amplitudinea și faza componentelor sinusoidale ale semnalului [1].

   Astfel, pentru un semnal aperiodic, de durată nelimitată, transformata Fourier se definește astfel:

$$X(e^{j\omega}) = \sum_{-\infty}^{+\infty} x(n)e^{-j\omega n} \tag{2.21}$$

   iar transformata Fourier inversă:

$$x(n) = \frac{1}{2\pi}\int_{-\pi}^{+\pi} X(e^{j\omega})e^{j\omega n}d\omega \tag{2.22}$$

   Deoarece pentru evaluarea numerică nu se poate efectua practic o sumă infinită și nu se poate introduce variabila continuă ω într-un sistem de prelucrare numerică, este necesar

ca semnalul să fie limitat la o durată finită, iar variabila continuă ω trebuie înlocuită cu o variabilă discretă.

Astfel, se va înlocui variabila continuă ω cu variabila discretă ω_k, prin notația:

$$\omega = k\Delta\omega, \qquad k \in \mathbb{Z} \tag{2.23}$$

unde $\Delta\omega$ este pasul utilizat pe axa frecvențelor. Aceste frecvențe discrete ω_k se mai numesc și frecvențe armonice.

Deoarece $X(e^{j\omega})$ este periodică în ω de perioadă 2π este suficientă utilizarea acestei înlocuiri pe o perioadă. Pentru un număr de eșantioane N pe o perioadă, obținem:

$$\Delta\omega = \frac{2\pi}{N} \tag{2.24}$$

Deci se poate face schimbarea de variabilă:

$$\omega \to \omega_k = \frac{2\pi}{N}k, \qquad k \in [-\frac{N}{2}, \frac{N}{2} - 1] \tag{2.25}$$

Astfel, spectrul continuu $X(e^{j\omega})$ va deveni un spectru discret, $X(k)$, fiind calculat în N puncte.

Se obțin astfel relațiile care definesc transformata Fourier discretă, respectiv inversă, pentru semnale de durată finită:

$$X(k) = \sum_{n=0}^{N-1} x(n)e^{-j\frac{2\pi}{N}nk}, \qquad \forall n \in \{0, 1, \ldots, N-1\} \tag{2.26}$$

$$x(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(k)e^{j\frac{2\pi}{N}kn}, \qquad \forall n \in \{0, 1, \ldots, N-1\} \tag{2.27}$$

Se va folosi transformata Fourier rapidă (*eng. Fast Fourier Transform*, pe scurt, `FFT`), ce reprezintă de fapt un algoritm de calcul rapid pentru transformata Fourier discretă (cu aceleași limitări de precizie). Principiul de calcul rapid se bazează pe descompunerea unei secvențe de lungime N în secvențe de lungime mai mică și pe proprietățile de simetrie și periodicitate ale lui $W_N = e^{-j\frac{2\pi}{N}}$. Din acest motiv, este de preferat ca numărul de puncte N_{fft} în care va fi calculată transformata Fourier să fie un multiplu de puteri ale lui 2. Se reduce astfel numărul de operații dramatic, ceea ce înseamnă o micșorare a timpului de calcul de câteva sute de ori.

Spectrul $X(e^{j\omega})$ este un spectru complex, fiind format din spectrul de amplitudine (partea reală) și spectrul de fază (partea imaginară). În general, pentru reprezentările frecvențiale se folosește doar spectrul de amplitudine $\mid X(e^{j\omega}) \mid$, iar spectrul de fază se neglijează. Acest lucru se datorează faptului că urechea este, în mare măsură, insensibilă la fază. Pentru acest proiect se va utiliza tot doar partea reală a spectrului, fiind de interes doar modificările în amplitudinile frecvențelor.

Transformata Fourier este utilă deoarece ne arată cum sunt distribuite frecvențele unui semnal audio. Deoarece vom lucra pe ferestre suficient de mici, iar melodiile sunt monofonice, vom avea o singură notă într-o fereastră de semnal, deci teoretic va exista o frecvență fundamentală (un vârf spectral) în spectrul de amplitudine pentru acea fereastră de semnal, iar această frecvență va caracteriza nota respectivă.

2. *Cepstru*

De-a lungul anilor, studiul semnalului vocal a prezentat un interes major în domeniul cercetării, încercând să se găsească diferite metode pentru a putea determina parametrii semnalului vocal utili, folosiți în procesul de recunoaștere a vorbirii. Astfel, s-a găsit o metodă, ce derivă din analiza Fourier, numită analiză cepstrală, bazată pe principiul detaliat în figura 2.4:

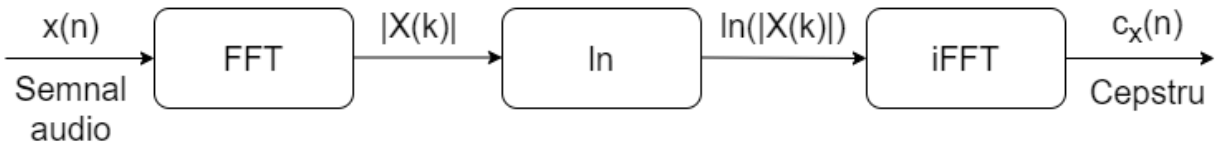


Figura 2.4: Modul de calcul al cepstrului

Astfel:

- Se extrage spectrul de amplitudine $\mid X(e^{j\omega} \mid)$ al semnalului audio folosind transformata Fourier;
- Se logaritmează acest spectru: $ln(\mid X(e^{j\omega} \mid))$. Acest lucru este foarte util în scopul de a simula modul în care urechea umană percepe semnalele audio, tot pe o scară logaritmică;
- Se aplică transformata Fourier inversă (sau alte transformate), pentru a calcula cepstrul semnalului.

Se poate deduce formula cepstrului real:

$$c_x(n) = \frac{1}{N} \sum_{k=0}^{N-1} ln(\mid X(k) \mid) e^{j\frac{2\pi}{N}kn}, \qquad \forall n \in \{0, 1, \ldots, N-1\} \tag{2.28}$$

Deoarece în acest caz s-a ales transformata Fourier inversă, rezultatul va fi într-un domeniu temporal. Dar acest domeniu nu este identic cu domeniul temporal inițial deoarece avem parametrii caracteristici de natură spectrală într-un domeniu temporal. Din acest motiv, noul domeniu va avea denumirea „domeniu cepstral". Din punct de vedere lingvistic, termenul de „cepstru" din limba română provine din „cepstrum" (în limba engleză), care reprezintă anagrama cuvântului „spectrum" (spectru în limba română).

În practică, pentru semnalul vocal, cepstrul se folosește în procesul de separare (deconvoluție) a două semnale ce intervin într-o operație de convoluție. Mai exact, semnalul vocal este convoluția în timp a semnalului excitație produs de vibrația coardelor vocale și răspunsul în timp al filtrului reprezentat de tractul vocal.

Pentru acest proiect, cepstrul a fost ales pentru clasificarea notelor datorită următoarelor proprietăți:

- *Analiza în domeniul frecvență* se face folosind scara logaritmică, precum cea a sensibilității urechii umane;
- *Analiza cepstrală* permite estimarea frecvenței fundamentale a semnalului, întrucât cepstrul reflectă periodicitatea acestui semnal. Astfel, maximele în amplitudine se află la distanța egală cu valoarea perioadei fundamentale.

3. *Transformata Wavelet*

Precizia transformatei Fourier este limitată de principiul de incertitudine determinat de Heisenberg, care se rezumă astfel: produsul dintre duratele în timp și frecvență ale unui semnal este limitat inferior de o valoare nenulă. Astfel, este necesar un compromis între rezoluția folosită pentru a caracteriza un semnal în domeniul timp și în domeniul frecvență [42].

În cazul analizei de tip Fourier pe termen scurt ($STFT$), fereastra de analiza are o durată fixă, indiferent de frecvența corespunzătoare semnalului studiat.

Transformata Wavelet este un mod de reprezentare a unui semnal cu ajutorul wavelet-urilor. Aceste wavelet-uri reprezintă copii scalate și translatate ale mai multor unde oscilante de lungime finită, numite „wavelet-uri mamă".

Wavelet-ul reprezintă un tip de funcție folosit pentru a împărți un semnal în diferite componente de timp-frecvență. În acest mod semnalele pot fi studiate la o rezoluție corespunzătoare scalei tipului de wavelet.

Un prim avantaj al transformatei Wavelet este faptul că oferă ferestre de durate variabile, ce depind de banda de frecvențe a semnalului: la frecvențe joase sunt folosite ferestre de durată lungă, iar la frecvențe înalte sunt folosite ferestre de durată scurtă. Astfel, se obține o localizare precisă a momentelor de timp în care apar componente de frecvențe înalte în semnal (dar nu se poate prezica exact valoarea acestor frevențe), respectiv se va determina cu mare precizie valoarea componentelor de frecvențe joase existente, fără a localiza strict momentul în care acestea apar. Un alt avantaj este dat de capacitatea de a analiza și sintetiza semnale neperiodice sau nestaționare.

Similar transformatei Fourier, există transformări Wavelet discrete (DWT) și transformări Wavelet continue (CWT). Transformările Wavelet continue pot acționa asupra oricărei scalări sau translații (situație ideală), iar transformările discrete folosesc o submulțime specifică de valori pentru aceste operații (situație reală). Pentru acest proiect se vor folosi transformatele discrete DWT (deoarece nu pot folosi un număr infinit de valori pentru parametrii).

Wavelel-ul reprezintă funcția ψ ce are următoarele proprietăți:

- Este o funcție de medie nulă:

$$\int_{-\infty}^{+\infty} \psi(t)dt = 0; \tag{2.29}$$

- Este o funcție normată:

$$|| \psi ||= 1; \tag{2.30}$$

- Este o funcție centrată în $t = 0$.

Prin scalarea acestei funcții ψ cu a și translatarea ei cu b, unde a și b reprezintă coeficienți reali, se obține o familie de semnale de timp-frecvență:

$$\psi_{a,b}(t) = \frac{1}{\sqrt{\mid a \mid}}\psi(\frac{t-b}{a}) \tag{2.31}$$

Transformata Wavelet pentru o funcție $f(t)$ este definită ca:

$$W(a,b) = \int_{-\infty}^{+\infty} f(t)\frac{1}{\sqrt{\mid a \mid}}\psi^*(\frac{t-b}{a})dt \tag{2.32}$$

Transformata Wavelet folosește analiza multirezoluție pentru a reprezenta semnalul. Aceasta constă în modelarea semnalului printr-o succesiune de aproximări ce conțin din ce în ce mai multă informație. Astfel, fiecare nivel de aproximație va conține întreaga parte de informație disponibilă la nivelul precedent, la care se adaugă o componentă suplimentară pentru detaliu (privesc pădurea și copacii din interior), proces ilustrat în figura 2.5:

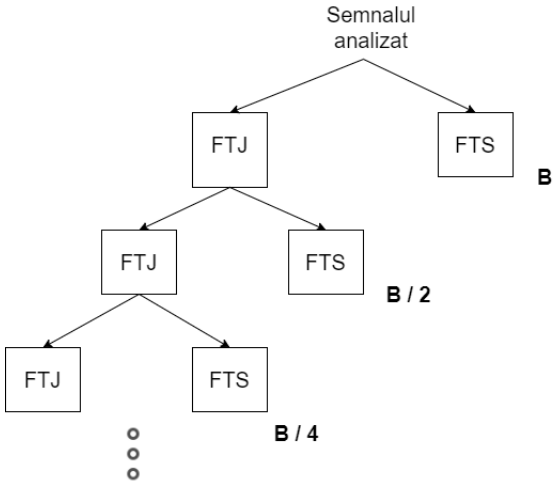


Figura 2.5: Aplicarea recursivă a bancurilor de filtre

Pentru un semnal unidimensional (1D), filtrul de analiză descompune semnalul în două subbenzi, o subbandă pentru frecvențe joase (partea grosieră) și o subbandă de frecvențe înalte (partea pentru detaliu). Transformata Wavelet este obținută prin descompunerea recursivă a subbenzii de frecvențe joase în alte două subbenzi mai mici. În acest mod, transformata Wavelet discretă produce o reprezentare multiscală a semnalului.

Pentru acest proiect se va folosi transformata Wavelet discretă utilizând familia de wavelet-uri *Daubechies*, mai exact wavelet-ul *Daubechies 1* (*db1*).

În urma experimentelor pentru clasificarea notelor folosind cele două baze de date, s-a constatat faptul că cea de-a doua bază de date oferă rezultate mult superioare față de prima bază de date. Din acest motiv, restul sarcinilor necesare pentru acest realizarea acestui proiect au fost efectuate doar pe cea de-a doua bază de date, prima bază fiind folosită doar pentru comparația rezultatelor obținute pentru cele două baze de date în clasificarea notelor.

2.4 Detecția onset / offset

Similar clasificării notelor, se va face în continuarea normarea semnalului la maximul său de amplitudine și se va realiza reeșantionarea la $f_e^{'} = 16kHz$, numărul de eșantioane corespunzător pentru o fereastră de 92ms fiind de $N = 1472$. Se va păstra suprapunerea ferestrelor dreptunghiulare de 90% și se va lucra doar pe semnalul pur deoarece se dorește atribuirea unor valori tot în domeniul temporal, fără legătură cu periodicitatea semnalului.

2.4.1 Prima metodă folosită

Pentru a detecta momentul în care apare / se termină o notă (onset / offset), s-au constituit 5 clase ce reprezintă tranzițiile ce se pot găsi într-o melodie:

1. Tranziția *pauză – pauză*. Aceasta reprezintă în general partea de început și partea de final a unei piese, acestea fiind momentele de liniște;

2. Tranziția *pauză – notă*. Acestea sunt momentele în care nota începe să fie cântată;

3. Tranziția *notă – pauză*. Sunt micile momente de pauză după încheierea unei note;

4. Tranziția *notă – o nouă notă*. Semnifică iterația de la o notă la alta, fără a exista pauze între aceste note;

5. Tranziția *notă – aceeași notă*. Reprezintă menținerea aceleiași note pentru o durată de timp.

După o mică analiză, se remarcă faptul că aceste tranziții, numite în continuare clase, nu au aceeași frecvență de apariție într-o înregistrare muzicală. Astfel, deoarece o melodie este concepută cu scopul de a induce o anumită senzație (fericire, tristețe, melancolie, etc.) ascultătorului, clasele ce arată momentele de pauză (tranzițiile 1, 2, respectiv 3) sunt destul de rare. De asemenea, trecerea de la o notă la alta (clasa 4) se face pentru un moment foarte scurt de timp, iar apariția acestei clase ține strict de numărul de note cântate într-o piesă, fără pauze între note. În concluzie, clasa dominantă a fiecărei melodii normale va fi clasa 5, ce reprezintă menținerea unei note. Această clasă depinde de lungimea ferestrei aleasă și de durata fiecărei note din înregistrarea audio. Deoarece am stabilit ca vom lucra doar cu cea de-a doua bază în continuare, vom avea o durată minimă de 93ms pentru o notă (durata maximă fiind de 3s) și vom folosi o fereastră dreptunghiulară de 92ms, cu suprapunere de aproximativ 90% (9.19ms) între ferestre, deci 147 de eșantioane, pentru a surprinde cât mai multe apariții ale acestor tranziții.

2.4.2 A doua metodă utilizată

Prima metodă introdusă oferă rezultate satisfăcătoare, dar prezintă marea problemă a faptului că tranzițiile prezentate într-o melodie normală apar rar, în afară de menținerea notei (clasa 5). Din acest motiv, există posibilitatea ca pentru o melodie să obținem un număr mai mic sau mai mare de intervale decât cele reale, mai exact un număr diferit de note, din cauza faptului că rețeaua nu poate prezice perfect tranzițiile dorite. Totodată, duratele notelor pot fi diferite de cele reale, uneori mai mici (în cazul apariției unei tranziții între note) sau mai mari (lipsa unei tranziții).

Totuși, datorită acestei metode inițiale, s-a remarcat faptul că cele mai evidente schimbări sunt în momentul în care se produce o notă (onset), datorită creșterii bruște de energie. Astfel, cea de-a doua metodă a fost realizată. În locul observării a 5 tranziții pe parcursul unei melodii, am decis observarea doar a apariției unei note. Definim astfel 2 clase:

1. *Clasa 0* – reprezintă momentele în care nu se produce nicio schimbare. Făcând analogia cu prima metodă, această clasă ar reprezenta tranzițiile pauză - pauză, respectiv notă - notă, deci putem spune că este menținerea fie a unei note, fie a unei pauze în timp, sau reformulat, nicio modificare față de momentul anterior.

2. *Clasa 1* – semnifică apariția unei note. În comparație cu prima metodă, această clasă ar fi combinarea tranzițiilor pauză – notă și notă – o nouă notă și reprezintă practic schimbările într-o melodie.

Este evident faptul că nici aceste clase nu sunt echilibrate din punct de vedere al aparițiilor, clasa 0 fiind mai comună în general. Acest lucru depinde în totalitate de numărul de note într-o melodie, deci pot fi piese ce vor avea note multe și scurte, adică mai multe schimbări (clasa 1), sau piese cu note foarte lungi și puține, cu multe nemodificări (clasa 0). Pentru baza proprie folosită, clasa 0 este de aproximativ 5 ori mai comună decât clasa 1, dar totuși acest raport

este normal și sunt suficiente date pentru antrenare. Folosind această clasificare binară pentru onset, s-au obținut rezultate mai bune decât prima metodă folosită, reducându-se numărul de tranziții de la 5 la 2 și fiind mult mai sesizabile schimbările.

Apare întrebarea: *Dacă pentru onset clasificarea binară funcționează, oare merge și pentru offset?* Răspunsul, pe scurt, este nu, sau mai bine zis, nu atât de bine, din câteva motive fundamentale ce țin de muzică, mai exact de modul de producere a unei note. Astfel, la emiterea unei note muzicale distingem următoarele etape ce formează anvelopa de amplitudine a notei:

1. *Atac* (*eng. attack*) - reprezină momentul de timp în care începe emiterea notei până când aceasta atinge amplitudinea sa maximă;
2. *Decădere* (*eng. decay*) - semnifică scăderea în amplitudine a notei după ce s-a atins maximul în amplitudine;
3. *Susținere* (*eng. sustain*) - este intervalul de timp în care nota este mentinută;
4. *Eliberare* (*eng. release*) - reprezintă momentul de timp în care sunetul este oprit înainte de stingerea sa naturală, numită reverberație sau ecou (pentru un interval mai mare).

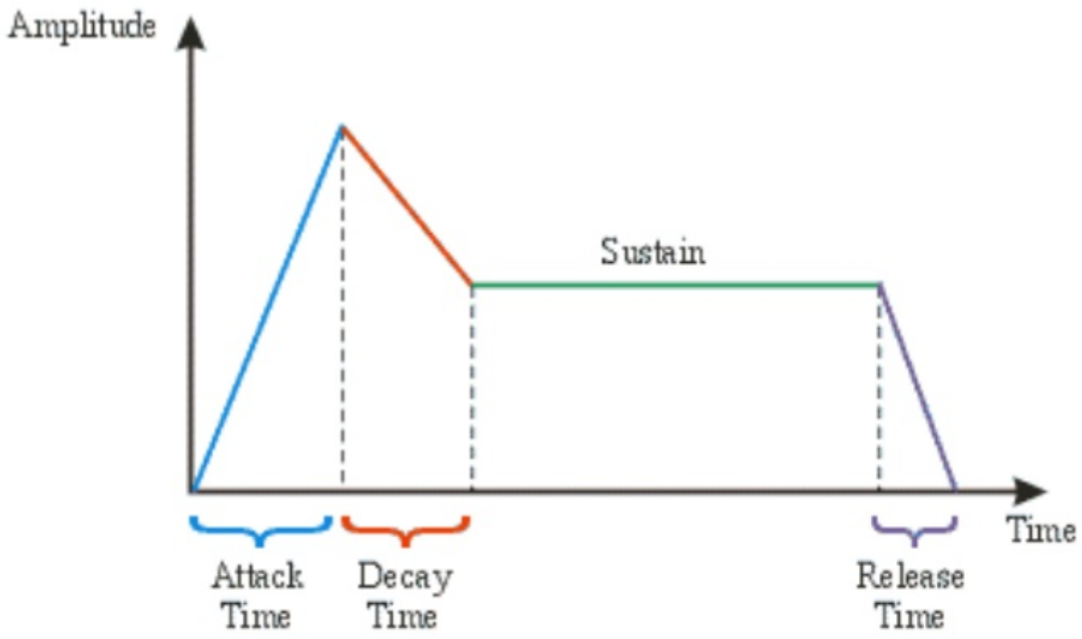


Figura 2.6: Anvelopa de amplitudine a unei note

Anvelopa de amplitudine din figura 2.6 [43] reprezintă variația de amplitudine a unei note în timp și este o caracteristică a timbrului muzical, deci fiecare instrument va avea această anvelopă (numită și anvelopă *ADSR*) diferită. De exemplu, instrumentele de percuție vor avea o perioadă de atac mai mică și o susținere scurtă față de restul instrumentelor. De asemenea, notele de frecvențe înalte au o degradare mai rapidă în timp față de notele de frecvențe joase, deoarece energia frecvențelor înalte se disipă mai rapid decât energia frecvențelor joase.

Întrucât instrumentul studiat în acest proiect este pianul, trebuie să precizăm că perioadă de atac este momentul de lovire a coardei metalice de către ciocănel și depinde de modul de apăsare a clapei (fie brusc, fie lent). De asemenea, pianul prezintă o decădere rapidă, iar perioada de susținere produce și ea o scădere treptată a amplitudinii notei, conform figurii 2.7 [43]:

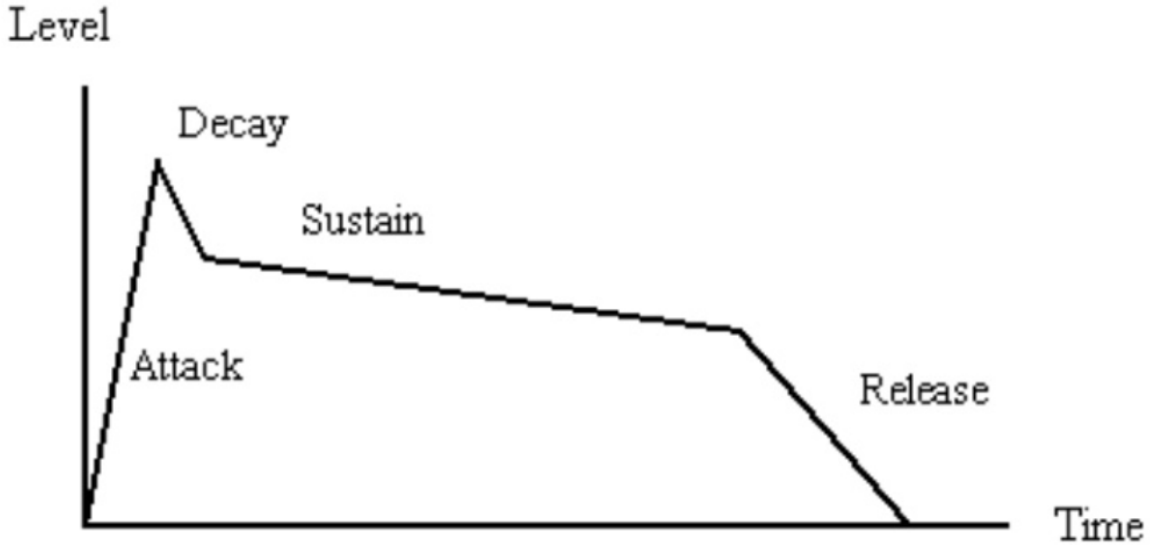


Figura 2.7: Anvelopa de amplitudine a unei note cântate la un pian

Astfel, putem acum observa de ce clasificarea binară a onset-ului functionează atât de bine. Deoarece introducem ferestre de 92ms ca intrare în rețea, vom surprinde de fapt perioada de atac a notei respective, care este cea mai sesizabilă perioadă a unei note, fiind o creștere bruscă în amplitudine și deci în energie. În schimb, dacă analizăm perioada în care nota este oprită, vom observa o scădere treptată a amplitudinii, din perioada de susținere, dar este greu să estimăm exact momentul opririi. Mai mult decât atât, în muzică, notele sunt cântate una după alta, de obicei cu perioade foarte mici între ele (uneori chiar nule). Din acest motiv, apariția unei noi note, deci o nouă perioadă de atac, va eclipsa perioada de eliberare a notei anterioare, facând astfel și mai greu de identificat momentul opririi acesteia fără a cunoaște informații despre noua notă. În plus, datorită interpretării muzicale (legăturii dintre note, momentele de tranziție între clape, etc.) și a înregistrării `MIDI`, pot exista cazuri reale în care offset-ul unei note anterioare apare după onset-ul notei viitoare, caz teoretic imposibil în melodiile monofonice, dar în realitate prezent pentru câteva zeci de ms. În concluzie, datorită acestor noi aspecte, putem preciza că offset-ul unei note este mult mai greu de interpretat folosind o clasificare binară similară onset-ului.

Totuși, încercând să analizăm de ce metoda aplicată onset-ului nu merge la fel de bine pentru offset, am sesizat o rezolvare pentru determinarea offset-ului. Deoarece am identificat deja toate onset-urile prezente într-o melodie, putem parcurge două onset-uri consecutive (numit cadru de notă) și pe parcursul acestui segment vom încerca să depistăm acea scădere în amplitudine din perioada de eliberare a notei. Deoarece o notă reprezintă un semnal cu o anumită periodicitate (datorită frecvenței fundamentale și a armonicelor sale) dar totuși departe de un semnal sinusoidal (pot exista diferite oscilații cu diferite amplitudini, în general mici, de-a lungul notei), pentru a pune în evidență forma anvelopei de amplitudine prezentată în figura 2.7, ne vom folosi de o mărime ajutătoare, numită flux spectral:

$$SF_{(i,i-1)} = \sum_{k=0}^{N-1} (\mid X_i(k) \mid - \mid X_{i-1}(k) \mid)^2 \tag{2.33}$$

unde $\mid X_i(k) \mid$ reprezintă coeficientul cu numărul k al spectrului de amplitudine $\mid X_i(e^{j\omega}) \mid$ pentru fereastra i a semnalului $x(n)$, formată din N eșantioane, iar $\mid X_{i-1}(k) \mid$ este coeficientul spectral cu numărul k pentru fereastra anterioară cu un eșantion, $i-1$, a semnalului $x(n)$.

Dar, după cum se observă, această formulă se bazează pe spectrul semnalului, deci este necesară transpunerea în domeniul frecvență folosind transformata Fourier. Deoarece nu dorim informații legate de periodicitate, deci frecvență, ci vrem să determinăm o scădere a amplitudinii, în timp, vom rescrie această formulă în domeniul temporal:

$$TF_{(i,i-1)} = \sum_{k=0}^{N-1} (x_i(k) - x_{i-1}(k))^2 \tag{2.34}$$

Această formulă va fi folosită pentru a calcula offset-ul unei note. Mai exact, pentru fiecare interval dintre două onset-uri succesive, se va calcula această mărime pe ferestre de 92ms și se va estima apariția offset-ului când este atinsă o valoare de prag minimă a acestei mărimi. În cazul în care pragul nu este atins, se va considera că tranziția dintre note se face imediat și offset-ul notei va fi considerat ca același moment de timp în care nota următoare începe. Mai bine zis, utilizând această tehnică se încearcă să se găsească pauza dintre note, sesizabilă prin diferența pătratică de amplitudini a două ferestre de semnal consecutive.

2.5 Estimarea intensității

Standardul `MIDI` permite 128 (intervalul 0 – 127) de intensități diferite pentru fiecare notă. Pentru a clasifica intensitatea unei note folosind rețele neurale este indicat ca fiecare valoare a intensității să apară de un număr minim de ori. Acest lucru nu a fost luat in considerare când a fost creată cea de-a doua bază de date, piesele fiind interpretate în mod natural, fără a varia într-un mod specific intensitatea notelor. Astfel, nu se vor găsi toate valorile de intensitate și numărul lor de apariții diferă în funcție de interpretare. Deoarece se dorește estimarea intensității, semnalul nu se mai normează la vârful maxim de amplitudine, dar reeșantionarea la $f_e^{'} = 16kHz$ este în continuare posibilă.

2.5.1 Metode deterministe

Metoda puterii maxime în perioade scurte de timp

Prima metodă folosită pentru estimarea intensității se bazează pe valorile amplitudinilor pentru o fereastră mică în timp a semnalului [44]. Astfel, pentru o fereastră dreptunghiulară de 25ms, suprapusă 90%, aplicată semnalului $x(n)$, se va calcula radicalul puterii acestei ferestre de semnal, după formula:

$$P_{i,j,rms} = \sqrt{\frac{1}{N} \sum_{n=0}^{N-1} x(n)^2} \tag{2.35}$$

unde $N = 400$ reprezintă numărul de eșantioane necesar unei ferestre de 25ms la o frecvență de eșantioanare $f_e^{'} = 16kHz$. Indicele i este numărul `MIDI` reprezentativ pentru valoarea intensității, iar indicele j semnifică fereastra cu numărul j aplicată semnalului corespunzător (notei) de intensitate i.

Fiecare intensitate i din baza de date are un număr k de apariții. Astfel, se va calcula pentru fiecare apariție k a intensității i maximul de putere din toate ferestrele j ale notei corespunzătoare:

$$P_{i,k} = \max P_{i,j,rms} \tag{2.36}$$

În final, pentru fiecare intensitate i din baza de date se va calcula o putere corespunzătoare, prin realizarea mediei pentru toate cazurile de apariție k obținute:

$$P_i = \frac{1}{k} \sum_{l=0}^{k-1} P_{i,l} \tag{2.37}$$

În acest mod se va obține o corespondență a puterii pentru fiecare valoare a intensității. Estimarea intensității notelor pentru o nouă melodie se va face prin aproximarea valorilor pentru puterile extrase din melodie cu puterile obținute din baza de date, iar apoi se va face corespondența cu valorile intensităților estimate.

Metoda amplitudinii maxime

Această metodă se bazează, în mod similar metodei pentru detecția offset-ului prezentată, pe caracteristicile muzicale ale notei. Deoarece majoritatea aparatelor de studio muzicale folosesc ca volum o reprezentare a intensității în decibeli, s-a decis studierea pe o scară logaritmică a amplitudinii notelor. Mai exact, dacă analizăm din nou anvelopa de amplitudine a unei note din figura 2.6, observăm că fiecare notă atinge un maxim de amplitudine pe durata ei. Astfel, în loc să parcurgem ferestre mici de semnal conform metodei puterii, ne vom folosi de această caracteristică muzicală și vom căuta pe durata unei note direct maximul ei de amplitudine, pe care îl vom înregistra în decibeli:

$$A_{max}[dB] = 20log(A_{max}) \tag{2.38}$$

Deoarece intensitățiile apar de mai multe ori în baza de date, este în continuare necesară o mediere a tuturor valorilor înregistrate pentru o intensitate i:

$$A_{max}(i) = \frac{1}{k} \sum_{l=0}^{k-1} A_{max}(i,l)[dB] \tag{2.39}$$

unde k este numărul de apariții pentru frecvența i.

Se va realiza astfel o mapare a intensităților în funcție de amplitudinile lor maxime.

2.5.2 Metoda regresivă

Deoarece scopul acestui proiect este de a folosi rețele neurale pentru a estima caracteristicle semnalului muzical, este necesar să prezentăm și o metodă folosind rețele pentru estimarea intensității. Această metodă se bazează tot pe anvelopa *ADSR* din figura 2.6, mai exact se folosește perioada de atac a unei note. Astfel, se va alege o fereastră suficient de mare pentru a cuprinde toată perioada de atac a unei note și se va folosi ca intrare într-o rețea de regresie ce va estima un număr cât mai apropiat de valoarea reală a intensității.

În general, perioada de atac a notelor este sub 20ms, dar pot exista cazuri în care o notă este apăsată foarte lent iar acest interval poate crește chiar la 40ms, posibil chiar puțin mai mult. Pentru a fi complet în siguranță, s-a decis utilizarea ferestrei de 92ms, apropiată de durată minimă a notelor din cea de-a doua bază de date, pentru a surprinde absolut complet o perioadă de atac pentru orice notă și pentru a oferi un context în privința descreșterii în amplitudine după maximul atins. Deoarece obținem doar o fereastră de semnal pentru fiecare notă cântată, pentru a mări numărul de date de intrare în rețea astfel încât să aibă suficiente informații pentru generalizare, s-a decis introducerea primelor 100 ferestre de semnal de 92ms la un eșantion distanță, pentru fiecare notă.

Capitolul 3
Implementarea sistemului

3.1 Realizarea sistemului

Sistemul complet va fi compunerea celor trei mari sarcini ce trebuie rezolvate, prezentat în figura 3.1. Etapele parcurse sunt următoarele:

1. *Detecția onset / offset* – este necesară pentru a determina momentul în timp de apariție a unei note, respectiv momentul în care aceasta se încheie. Se realizează întâi această sarcină pentru a putea surprinde intervalele de timp în care sunt cântate notele, utile pentru clasificarea notelor. În această etapă este necesară o postprocesare în care se aplică o filtrare mediană asupra predicțiilor rețelei pentru metoda tranzițiilor, iar suplimentar în cazul metodei clasificării binare se calculează offset-ul din intervalul a două onset-uri consecutive. Pentru a realiza această etapă, semnalul audio va fi împărțit in ferestre dreptunghiulare de semnal, de lungimea 92ms, suprapuse 90%, ce vor fi introduse apoi la intrarea rețelei neurale destinată rezolvării acestei sarcini. Durata fiecărei note va fi obținută ca diferența dintre offset-ul și onset-ul acesteia.

2. *Clasificarea notelor* – folosind intervalele în timp determinate în prima etapă, acestea vor fi împărțite în ferestre de semnal cu lungimi de 32ms, 46ms, respectiv 92ms (suprapuse 67%, 75%, 90%) și se vor extrage anumite trăsături ce vor fi introduse în rețeaua neurală pentru clasificarea de note. Se vor compara rezultatele obținute pentru fiecare fereastră și se va alege lungimea optimă a ferestrei folosite și tipul de trăsătură cel mai performant. Deoarece există o suprapunere mare, nota finală va fi cea mai frecventă notă apărută într-un interval determinat în prima etapă.

3. *Estimarea intensității* – pentru această etapă se va folosi semnalul audio fară a fi normat la valoarea maximă de amplitudine, în scopul de a avea valori cât mai distincte ale amplitudinii semnalului într-un interval cât mai larg. Se vor utiliza din nou intervalele obținute din *detecția onset / offset* și se vor aplica cele 3 metode propuse, în vederea comparării acestora.

După ce toate aceste etape au fost realizate cu succes, este necesară convertirea informațiilor obținute într-un fișier `MIDI`, tipic pentru transcrierea automată a muzicii. Vor fi deci introduse pentru fiecare notă pitch-ul (reprezentat printr-un număr `MIDI`), durata (mai exact onset-ul și offset-ul notei), precum și intensitatea acesteia (scrisă ca un număr `MIDI` în intervalul 0 – 127). Se va folosi frecvența de eșantionare clasică de 44.1 kHz și un tempo uzual de 120 BPM.

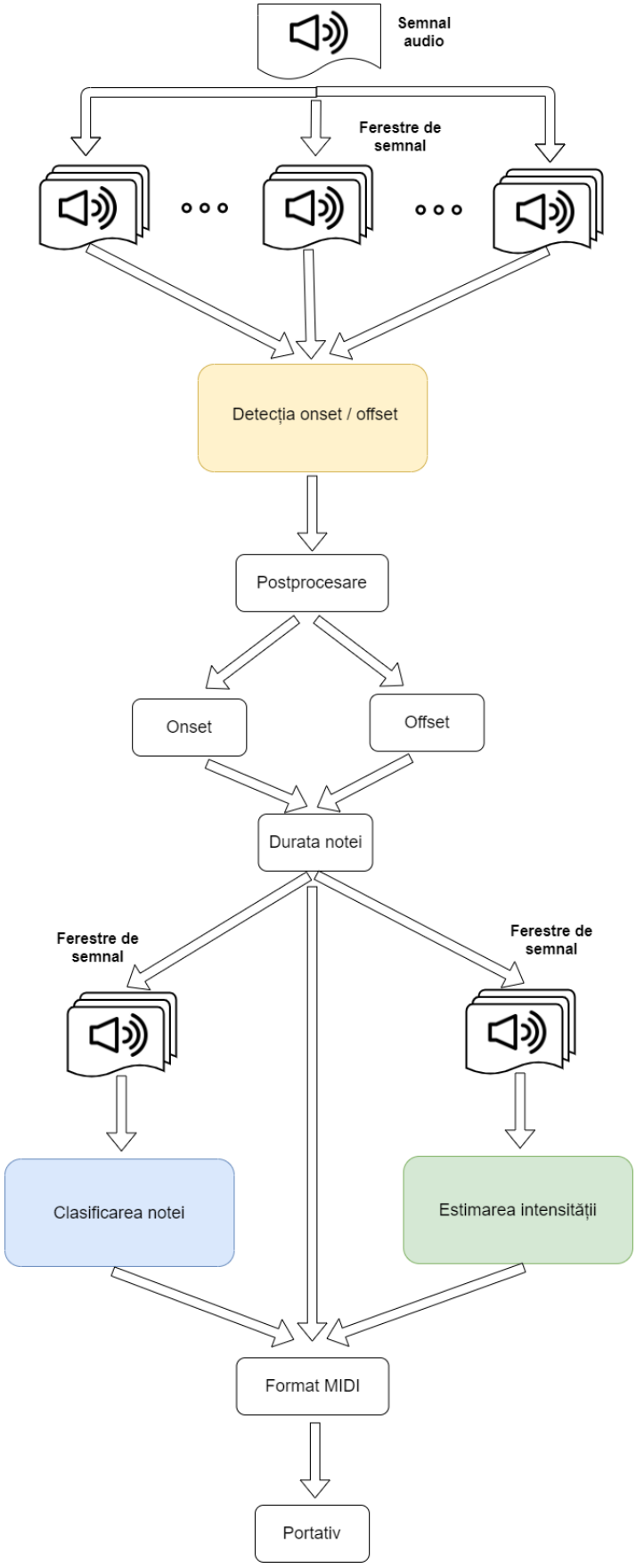


Figura 3.1: Reprezentarea sistemului propus

3.2 Arhitecturi folosite

3.2.1 În clasificarea notelor

Pentru a obține un semnal cvasistaționar, vom lucra cu lungimi ale ferestrelor de 32ms, 46ms, respectiv 92ms:

- Lungimea de 32ms a fost aleasă pentru a obține detalii ale semnalului audio pentru o durată cât mai mică în timp (similar ferestrei de 25ms folosită pentru semnalul vocal). Deși prima bază de date conține semnale mai mici de această durată (durata minimă de 14ms), s-a ales acestă valoare ca un compromis pentru a cuprinde cât mai multe note cântate, chiar dacă perioada necesară pentru frecvența minimă ($f_{min1} = 27.5Hz$) este de 36.6ms. Pentru a doua bază de date nu există astfel de probleme, perioada necesară pentru frecvența minimă ($f_{min2} = 65.4Hz$) fiind de 15.3ms.

- Durata de 46ms a fost aleasă ca o valoare intermediară între ferestrele de 32ms și 92ms. Practic, această durată cuprinde cel puțin o perioadă necesară pentru identificarea oricărei note, în detrimentul neglijării notelor cu durate mai mici de 46ms pentru prima bază de date.

- Fereastra de 92ms este folosită pentru a cuprinde minim două perioade ale oricărei note (lucru util pentru cepstru și funcția de autocorelație, ce oferă informații bazate pe această periodicitate a semnalelor). Dezavantajul acestei lungimi este faptul că vor fi neglijate notele din prima bază de date cu durată mai mică decât această fereastră, dar această lungime a ferestrei este optimă pentru cea de-a doua bază de date, deoarece notele înregistrate au o durată minimă de 93ms.

Întrucât a doua bază de date este de aproximativ 5 ori mai mică decât prima bază de date, pentru preprocesarea înregistrărilor audio din a doua bază de date se va face o suprapunere de aproximativ 67% / 75% / 90% (10.6ms, 11.5ms, 9.19ms) ale acestor ferestre (32ms, 46ms, respectiv 92ms), în scopul de a obține un număr cât mai mare de trăsături ce vor fi transmise la intrarea rețelei neurale.

Pe fiecare dintre aceste ferestre se vor extrage cele 5 trăsături prezentate anterior. Pentru a reduce fenomenul de dispersie a spectrului și a erorilor de amplitudine, pentru transformata Fourier se va folosi fereastra de pondere *Hamming*, urmând ca pentru restul de trăsături să fie utilizată fereastra dreptunghiulară, pentru a nu modifica semnalul audio. În cazul trăsăturilor bazate pe transformata Fourier, se va extrage doar prima jumătate (informația utilă) a spectrului [0 – (N_{fft}/2 - 1)], cealaltă jumătate reprezentând de fapt oglindirea spectrală a primei jumătăți.

Deoarece avem două baze de date, arhitectura rețelei neurale va fi diferită pentru fiecare bază.

1. *Arhitectura rețelei neurale pentru clasificarea notelor folosind prima bază de date*

   Deoarece înregistrările audio ale acestei baze sunt făcute la o frecvență de eșantionare de $f_e = 44.1kHz$, pentru a calcula numărul de eșantioane N necesar pentru fiecare fereastră se aplică următoarea formulă:

$$\Delta f = \frac{f_e}{N} => N = \frac{f_e}{\Delta f} => N = int(T_w \times f_e), \qquad N \in \mathbb{Z} \tag{3.1}$$

   unde $T_w = \frac{1}{\Delta f}$ reprezintă lungimea ferestrei dorite.

   Astfel, pentru cele 3 lungimi de ferestre, obținem:

- $N1 = 1411$, pentru $T_{w1} = 32ms$;
- $N2 = 2028$, pentru $T_{w2} = 46ms$;
- $N3 = 4057$, pentru $T_{w3} = 92ms$.

Se observă faptul că $N2$ și $N3$ sunt foarte apropiate de multiplii de puteri ale lui 2 (2048, respectiv 4096), deci va fi nevoie de un număr minim de eșantioane cu valoarea 0 introduse suplimentar într-o fereastră de semnal pentru a obține un număr de eșantioane multiplu de puteri ale lui 2 (N_{fft}), necesar pentru performanța transformatei Fourier.

Astfel, se constată faptul că dimensiunea minimă a fiecărui vector de trăsături pe o fereastră va fi de 1024 ($Nfft_{min} = 2048 => Nfft_{min}/2 = 1024$), iar cea maximă de 2048 ($Nfft_{max} = 4096 => Nfft_{max}/2 = 2048$).

Deoarece scopul rețelei neurale este de a găsi o legătură între vectorul de intrare (trăsăturile) și cel de ieșire (notele), se dorește o scădere treptată a dimensiunii parametrilor caracteristici extrași de rețea.

Astfel, este necesar ca numărul de neuroni din primul strat ascuns al rețelei neurale să fie apropiat de dimensiunea vectorului de trăsături de la intrare. Întrucât se preferă un număr de neuroni multiplu al puterilor lui 2, se va alege cel mai apropiat număr după dimensiunea minimă a vectorului de trăsături (1024), care va fi 512. Cel de-al doilea strat ascuns va avea dimensiunea de 256 de neuroni, iar ultimul strat ascuns va fi de 128 neuroni și va fi un strat de legătură pentru stratul de ieșire, format din cele 88 de clase (note). În urma experimentelor, s-a constatat că trei straturi ascunse este numărul optim pentru rețeaua neurală folosită în clasificarea notelor pentru prima bază de date, prezentată în figura 3.2:

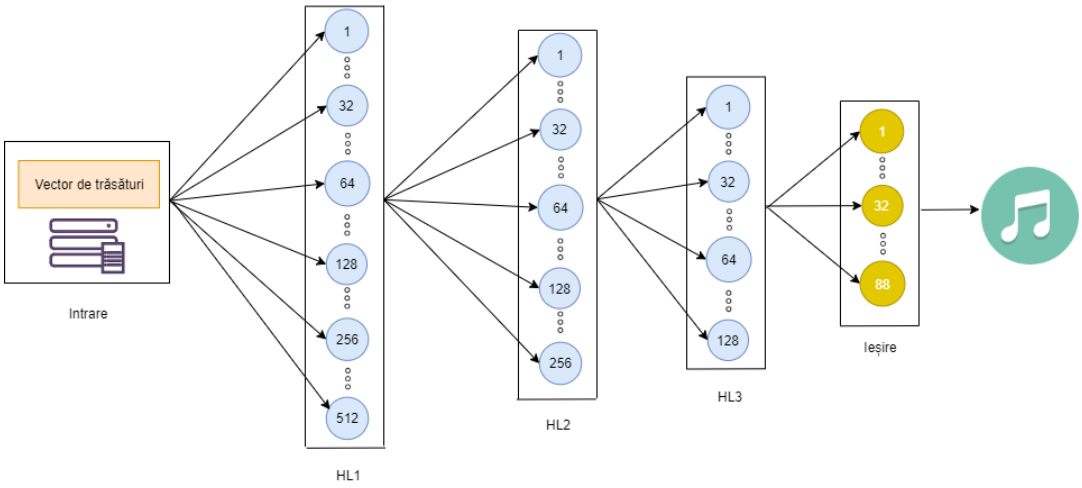


Figura 3.2: Arhitectura rețelei neurale pentru clasificarea notelor folosind prima bază de date

2. *Arhitectura rețelei neurale pentru clasificarea notelor folosind a doua bază de date*

Similar primei baze de date, cele 100 de înregistrări ale acestei baze au fost făcute la o frecvență de eșantionare de $f_e = 44.1kHz$. Deoarece frecvența maximă din această bază este de $f_{max2} = 2093Hz$, se poate face o reeșantionare, obținând o nouă frecvență de eșantionare de $f_e^{'} = 16kHz$. Acest lucru permite obținerea unui spectru de 8kHz și reducerea numărului de eșantioane N al lungimii ferestrelor. Conform formulei (3.1), pentru cele 3 lungimi de ferestre se obțin:

- $N1 = 512$, pentru $T_{w1} = 32ms$

- $N2 = 736$, pentru $T_{w2} = 46ms$
- $N3 = 1472$, pentru $T_{w3} = 92ms$

Din figura 3.3 se observă faptul că este nevoie de un număr mai mic de straturi. Astfel, primul strat ascuns va avea 256 neuroni, iar cel de-al doilea strat (stratul intermediar) va avea dimensiunea de 64 neuroni, fiind urmat de stratul de ieșire format din cele 61 de clase.

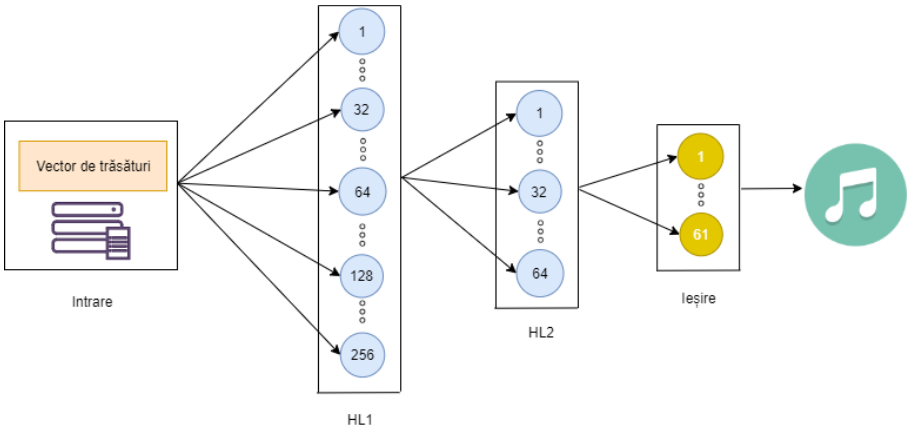


Figura 3.3: Arhitectura rețelei neurale pentru clasificarea notelor folosind a doua bază de date

Se remarcă astfel o reducere a dimensiunii rețelei neurale atât în numărul de straturi cât și în dimensiunea fiecăruia, deci pentru cea de-a doua bază se va folosi o rețea mai simplă și mai rapidă.

S-a încercat realizarea unei constrângeri pozitive asupra ponderilor rețelei neurale. Niște studii recente [45] au arătat că dacă ponderile unei rețele dense sunt strict pozitive, atunci constanta Lipschitz a rețelei poate fi aproximată eficient. Este bine-cunoscut faptul că această constantă este strâns legată de robustețea rețelelor împotriva atacurilor adversiale. Experimental s-a determinat că este posibilă păstrarea performanței rețelei, antrenată sub constrângeri de pozitivitate, prin adăugarea unui nou strat ascuns cu 512 neuroni. Acest rezultat preliminar constituie baza unei noi direcții de explorare, aceea a antrenării de rețele robuste la posibile atacuri. Arhitectura folosită este detaliată în figura 3.4:

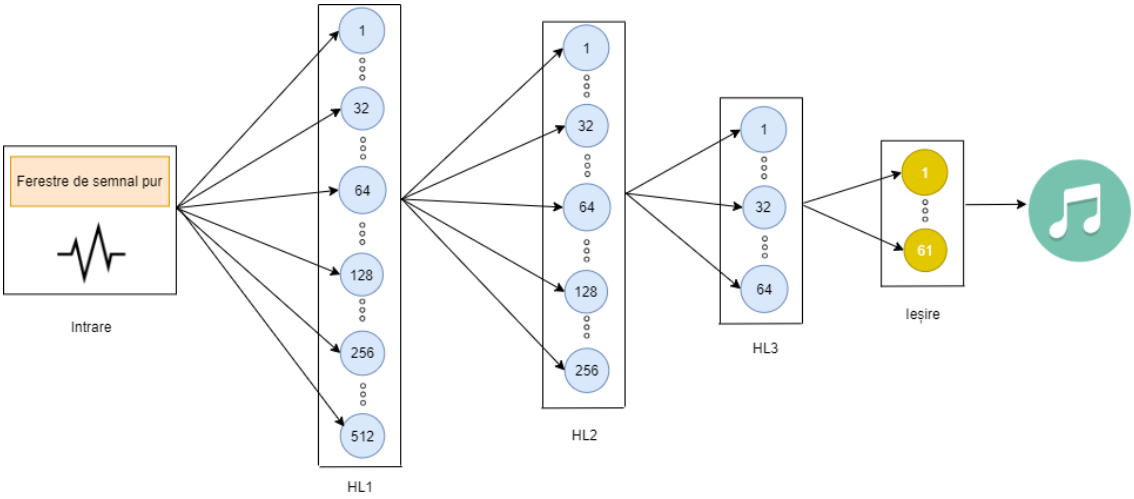


Figura 3.4: Arhitectura cu constrângerea pozitivă a ponderilor rețelei neurale

Deoarece bazele de date sunt suficient de mari, s-a folosit un *lot* (*eng. batch*) de 512. După fiecare strat ascuns se introduce o etapă de *normalizarea a lotului* (*eng. batch-normalization*), această tehnică îmbunătățind viteza, performanța și stabilitatea rețelei neurale. Ca funcție de activare s-a folosit *ReLU*, cu excepția ultimului strat (cel de ieșire), unde s-a folosit funcția *Softmax* pentru a calcula probabilitățile claselor (în intervalul [0 – 1]). Deoarece clasificarea notelor este o problemă de tip multi-clasă, pentru *funcția cost* s-a folosit *categorical cross-entropy*. Pentru ajustarea ponderilor s-a utilizat optimizatorul *Adam*, datorită vitezei rapide de învățare în comparație cu optimizatorul *SGD*.

3.2.2 În detecția onset / offset

În rezolvarea aceastei sarcini s-a folosit doar fereastra de 92ms pentru a observa o porțiune cât mai mare de semnal, dând astfel un context în momentul tranzițiilor, iar suprapunerea de 90% este realizată pentru a observa cât mai multe apariții ale momentele în care sunt produse schimbări în semnalul audio.

Pentru prima metodă (metoda tranzițiilor)

Pentru că aceasta este tot o sarcină de clasificare, s-au folosit aceiași parametrii prezentați în clasificarea notelor (*batch* de 512, *batch-normalization*, funcția de activare *ReLU*, respectiv *Softmax* pe ultimul strat, funcția cost *categorical cross-entropy*), dar a fost necesară și adăugarea unei regularizări de tip *dropout* de 0.2 după fiecare strat ascuns pentru a evita procesul de supraînvățare, deoarece clasificarea acestor tranziții este mai dificilă decât clasificare notelor.

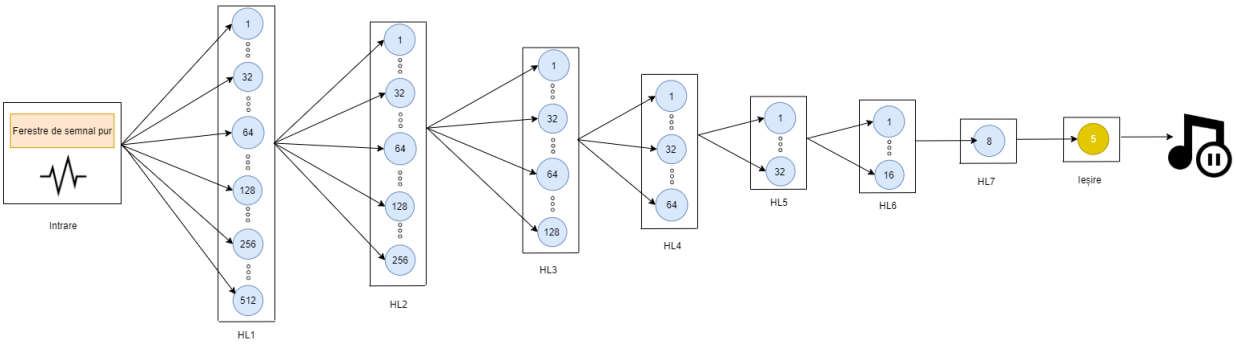


Figura 3.5: Arhitectura rețelei neurale pentru detecția celor 5 tranziții

Datorită faptului că aceste tranziții sunt mai greu de învățat pentru o rețea neurală, arhitectura prezentată în figura 3.5 va fi mai mare, în scopul învățării unor funcții mai complexe ce pot caracteriza aceste schimbări. Astfel, la intrare vom avea ferestre de dimensiunea $N = 1472$ și folosind 7 straturi ascunse ce descresc ca număr de neuroni cu un factor de 2 față de stratul anterior, rețeaua va fi capabilă să recunoască cele 5 clase de la ieșire.

După predicția rețelei este necesară o etapă de postprocesare, în care se aplică o filtrare mediană cu un factor de ordinul $k = 3$ cu scopul de a înlătura predicțiie izolate false. Urmărind tranzițiile după apariția unei note, offset-ul va fi determinat ca prima tranziție de pauză întâlnită pe durata unei note sau, în lipsa acestora, va fi ultima apariție a unei tranziții ce presupune existența unei note.

Pentru cea de-a doua metodă (clasificare binară onset)

În privința parametrilor rețelei neurale, singura modificare față de prima metodă este faptul că se folosește funcția cost *binary cross-entropy*, deoarece aceasta este o problemă de clasificare binară, fiind doar 2 clase introduse, clasa 0 ce reprezintă faptul că nu este produsă nicio schimbare în melodie, respectiv clasa 1 ce semnifică practic momentul apariției unei note. Arhitectura rețelei din figura 3.6 în schimb va fi mai mică, deoarece este mai ușor de estimat începutul unei note și discutăm doar de 2 clase în loc de 5.

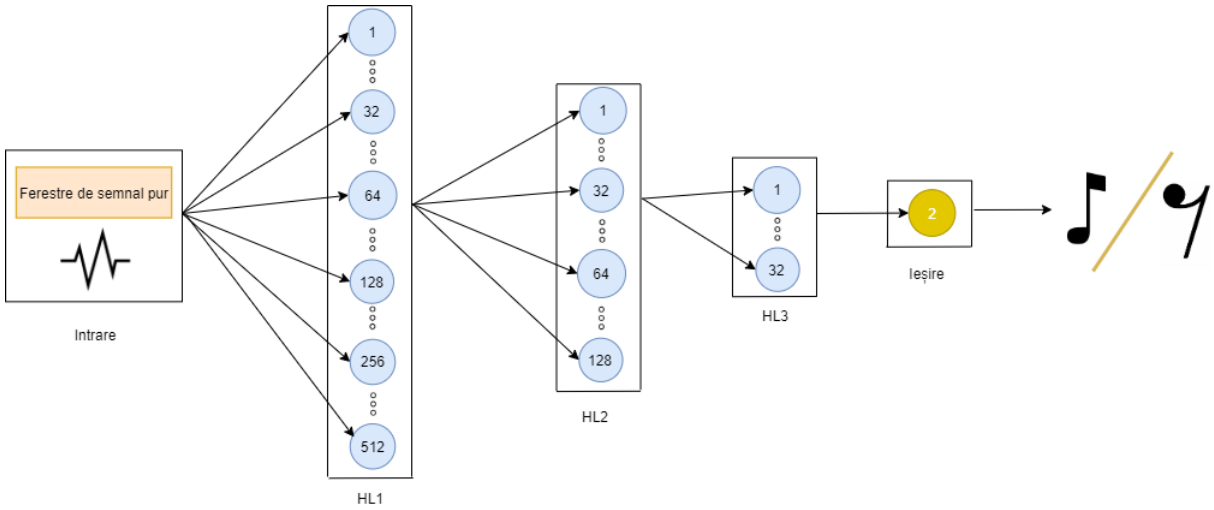


Figura 3.6: Arhitectura rețelei neurale pentru detecția onset-ului

Deoarece la intrare vom avea ferestre de dimensiunea N = 1472, iar la ieșire avem cele 2 clase, în urma experimentelor s-a constatat faptul că 3 straturi ascunse sunt suficiente pentru o învățare bună a rețelei, fiecare strat fiind practic de 4 ori mai mic decât cel precedent.

După predicția onset-urilor de către rețea, se vor folosi cadre de note succesive pentru a determina offset-ul acestora.

Pentru a înțelege mai bine cum se determină offset-ul, este necesar să ilustrăm cele două cazuri posibile, adică tranziția aproape instananee de la o notă la alta – figura 3.7, sau existența unei pauze între note – figura 3.8.

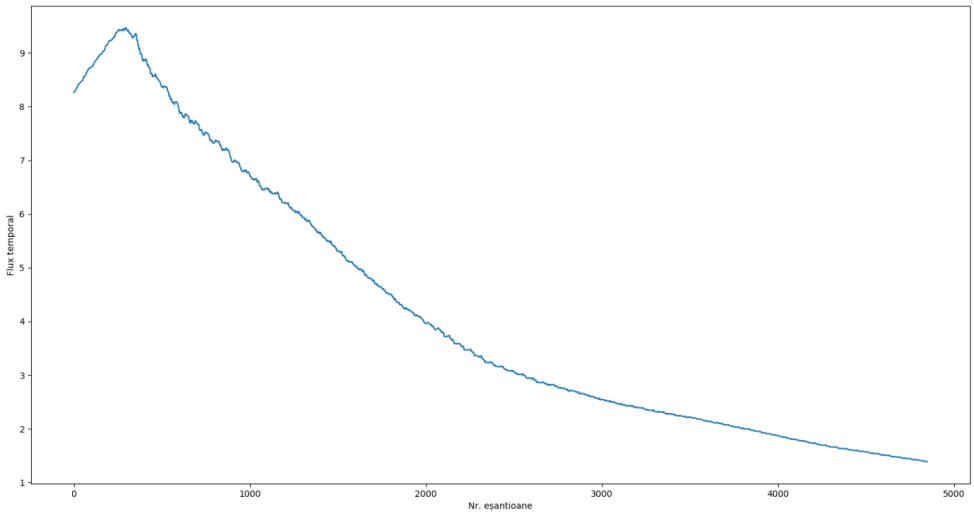


Figura 3.7: Fluxul temporal în cazul tranziției rapide de la o notă la alta

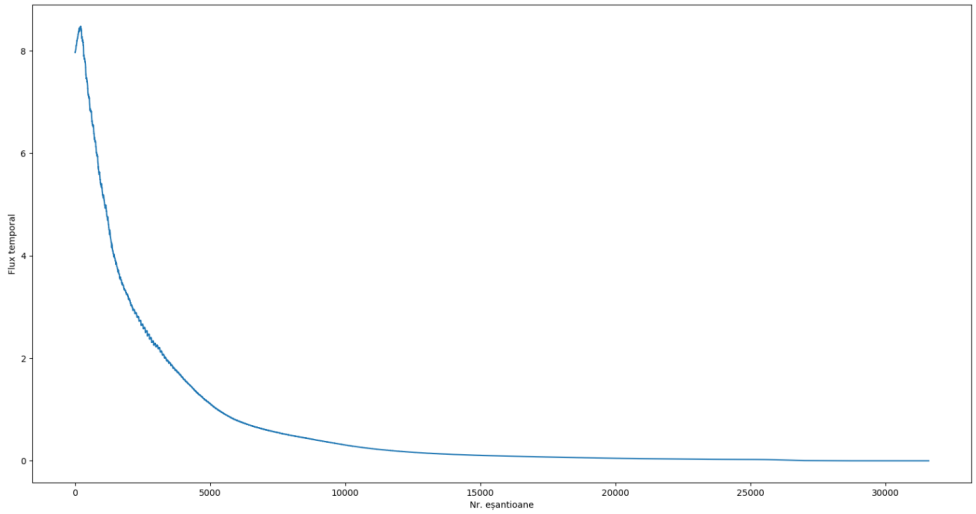


Figura 3.8: Fluxul temporal în cazul unei pauze între note

Se observă astfel cum în cazul existenței unei pauze între note există o plafonare după un timp și sunt atinse valori mult mai mici, condiție esențială pentru determinarea offset-ului. Offset-ul va fi selectat ca momentul în care fluxul temporal al cadrului de notă atinge valoarea de 0.2% din prima valoare a acestuia, adică fluxul ce surprinde în totalitate perioada de început și deci de atac a notei. În cazul în care această valoare de prag nu este atinsă, offset-ul va fi considerat ca momentul de început al notei următoare.

3.2.3 În estimarea intensității

Deoarece aceasta este o problemă de regresie, se va folosi ca funcție cost eroarea pătratică medie (*eng. Mean Squared Error*, pe scurt, *MSE*), iar funcția de activare pe ultimul strat va fi liniară, deoarece nu dorim calculul unor probabilități de clase, restul straturilor ascunse folosind în continuare funcția *ReLU*. Se va folosi în continuare *normalizarea pe batch* după fiecare strat ascuns, precum și un *dropout* de 0.1, iar dimensiunea lotului ales va fi de 32 în acest caz.

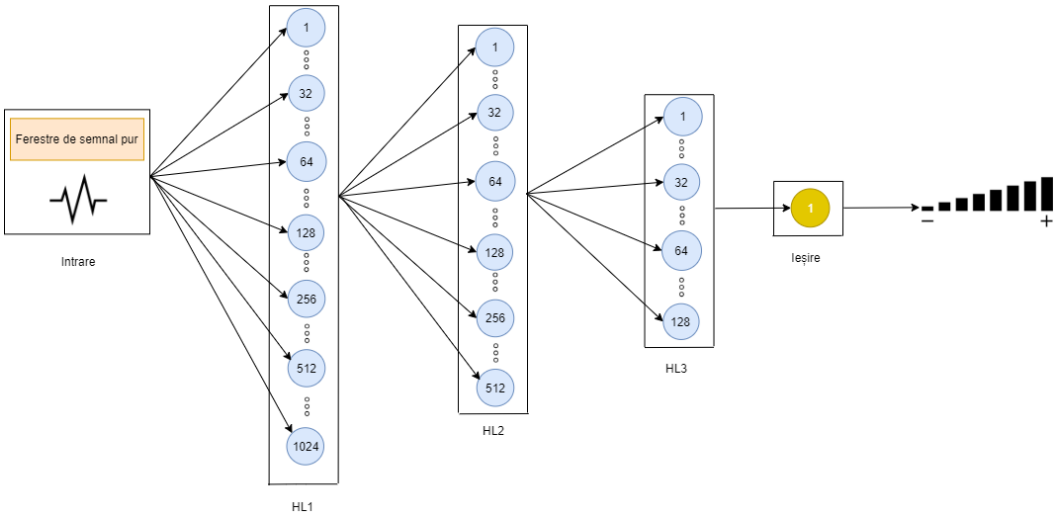


Figura 3.9: Arhitectura rețelei neurale pentru estimarea intensității

Rețeaua de regresie prezentă în figura 3.9 va fi compusă din 3 straturi ascunse, având dimensiunile 1024, 512, respectiv 128, necesare pentru a putea aproxima valoarea intensității de pe ultimul strat, având ca intrare ferestre de semnal pur de 92ms, deci 1472 de eșantioane.

Capitolul 4
Experimente și rezultate

4.1 Clasificarea notelor

4.1.1 Utilizând prima bază de date

	Acuratețe [%]					
	Fereastră de 32ms		Fereastră de 46ms		Fereastră de 92ms	
Trăsături	Antrenare	Validare	Antrenare	Validare	Antrenare	Validare
FFT(hamming)	89.53	84.22	88.57	85.10	88.35	86.66
DWT(db1)	85.33	82.41	85.83	83.31	89.47	85.84
Cepstru	85.71	81.81	84.47	82.81	88.64	85.80
Semnal pur	84.13	81.81	87.49	83.61	88.50	85.00
Autocorelația	80.97	78.00	82.64	79.41	87.54	83.79

Tabela 4.1: Analiza semnalului utilizând prima bază de date

Se observă din tabelul 4.1 faptul că indiferent de lungimea ferestrei de semnal aleasă sau tipul de trăsătură (acuratețea maximă de 86.66% folosind transformata Fourier și fereastra *Hamming*), rezultatele sunt inferioare față de starea artei, cu o acuratețe de peste 95% . Din acest motiv, această bază nu a mai fost folosită pentru restul sarcinilor, fiind păstrate rezultatele doar pentru a le compara cu cele obținute folosind cea de-a doua bază de date.

4.1.2 Utilizând a doua bază de date (baza proprie)

	Acuratețe [%]					
	Fereastră de 32ms		Fereastră de 46ms		Fereastră de 92ms	
Trăsături	Antrenare	Validare	Antrenare	Validare	Antrenare	Validare
Cepstru	99.49	98.18	99.66	98.24	99.84	98.47
FFT(hamming)	99.00	98.10	99.59	98.15	99.46	98.25
Semnal pur	98.54	97.59	98.06	97.55	99.16	98.00
DWT(db1)	98.38	97.43	98.48	97.32	99.02	97.87
Autocorelația	97.98	97.03	98.34	96.97	98.34	97.38

Tabela 4.2: Analiza semnalului utilizând a doua bază de date

Sesizăm din tabelul 4.2 faptul că rezultatele sunt mult mai bune, deci contează foarte mult baza de date folosită pentru antrenarea rețelelor neurale. Astfel, se remarcă faptul că diferența dintre rezultatele folosind semnalul pur și cea mai bună trăsătură (în acest caz, *cepstrul*, cu o acuratețe de 99.84% pe antrenare și 98.47% pe setul de validare) este destul de mică, în general tot ceea ce depășește pragul de acuratețe de 98–99% fiind considerat foarte bun și îmbunătățit prin modificarea arhitecturii și a parametrilor utilizați în rețea. Din acest motiv, putem considera că semnalul pur este suficient pentru clasificarea notelor folosind rețele neurale pe această bază de date, fără a fi nevoie de folosirea unor transformate complicate și astfel acesta va fi folosit pentru testare.

Folosind semnalul pur pentru intrarea în rețeaua neurală, numită în acest caz și rețea *End-to-End* deoarece semnalul nu este prelucrat în vreun mod, precum și constrângerea în privința ponderilor pozitive, pentru melodia de testare, ce conține 60 de note, s-a obținut o acuratețe de 99.39%. Această acuratețe a fost obținută evident pe ferestrele de 92ms suprapuse 90%. În realitate toate notele au fost prezise corect, după cum se observă în figura 4.1, deoarece în partea de postprocesare se alege cea mai frecventă notă apărută pe durata acesteia, obținută din detecția onset / offset.

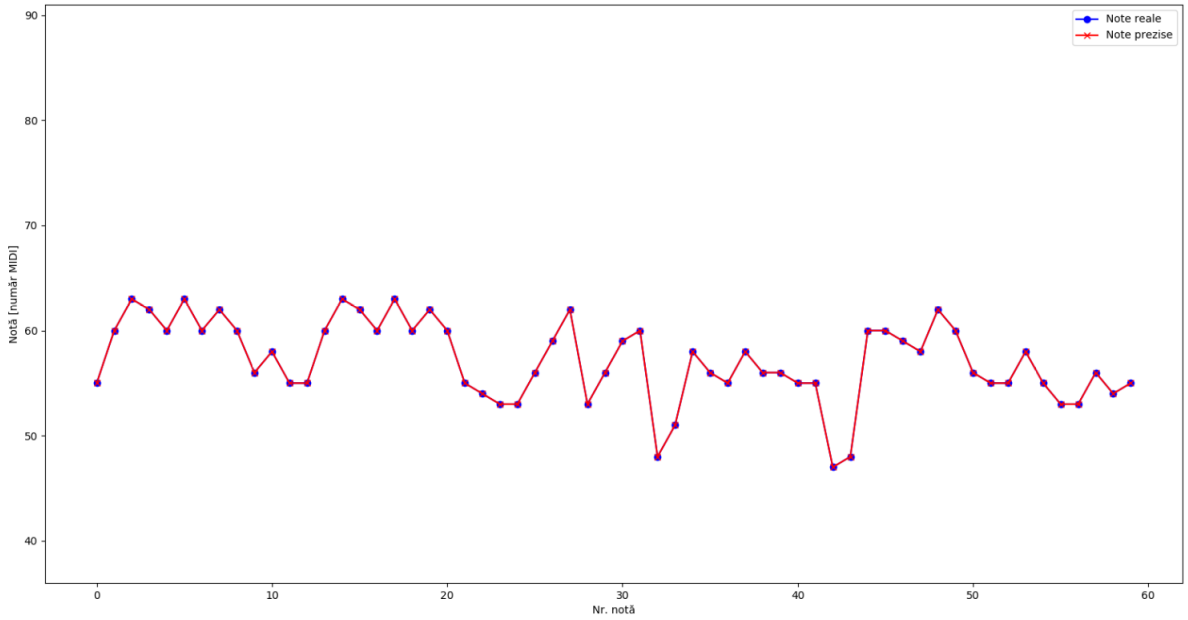


Figura 4.1: Comparație note

4.2 Detecția onset / offset

Pentru a compara cele două metode utilizate, vom defini un onset prezis corect dacă diferența absolută dintre cel real și cel prezis nu depășește 25ms, iar în cazul offset-ului, diferența maximă admisă va fi de 50ms, valori uzuale folosite în literatură. Această acuratețe este exprimată astfel:

$$acc_{onset}(i) = \begin{cases} 1, & \text{pentru } \mid onset_{real} - onset_{prezis} \mid \leq 25ms \\ 0, & \text{în rest} \end{cases} \tag{4.1}$$

$$acc_{offset}(i) = \begin{cases} 1, & \text{pentru } \mid offset_{real} - offset_{prezis} \mid \leq 50ms \\ 0, & \text{în rest} \end{cases} \tag{4.2}$$

$$acc_{onset/offset} = \frac{1}{N}\sum_{i=1}^{N} acc_{onset/offset}(i) \times 100 \tag{4.3}$$

De asemenea, pentru a obține și un număr pe lângă acuratețea stabilită mai sus, vom studia *eroarea medie pătratică* totală obținută între toate duratele notelor reale și prezise, conform formulei:

$$MSE = \frac{1}{N}\sum_{k=1}^{N}(durata_{\text{reală}} - durata_{\text{prezisă}})^2 \tag{4.4}$$

unde $durata = offset - onset$, iar N reprezină numărul de note din melodia de testare.

4.2.1 Folosind prima metodă

Rețeaua neurală folosită pentru metoda tranzițiilor oferă o acuratețe de 88.93% pe setul de antrenare, 86.49% pe setul de validare, iar pe melodia de testare se obține 92.39%, predicția după postprocesare fiind prezentată în figura 4.2.

Utilizând mărimile prezentate mai sus pentru comparație, obținem:

$$acc_{onset} = 98.33\%, \text{fiind prezise corect 59 din 60 de onset-uri} \tag{4.5}$$

$$acc_{offset} = 86.67\%, \text{fiind prezise corect 52 din 60 de offset-uri} \tag{4.6}$$

$$MSE = 1.30 \times 10^{-3} \tag{4.7}$$

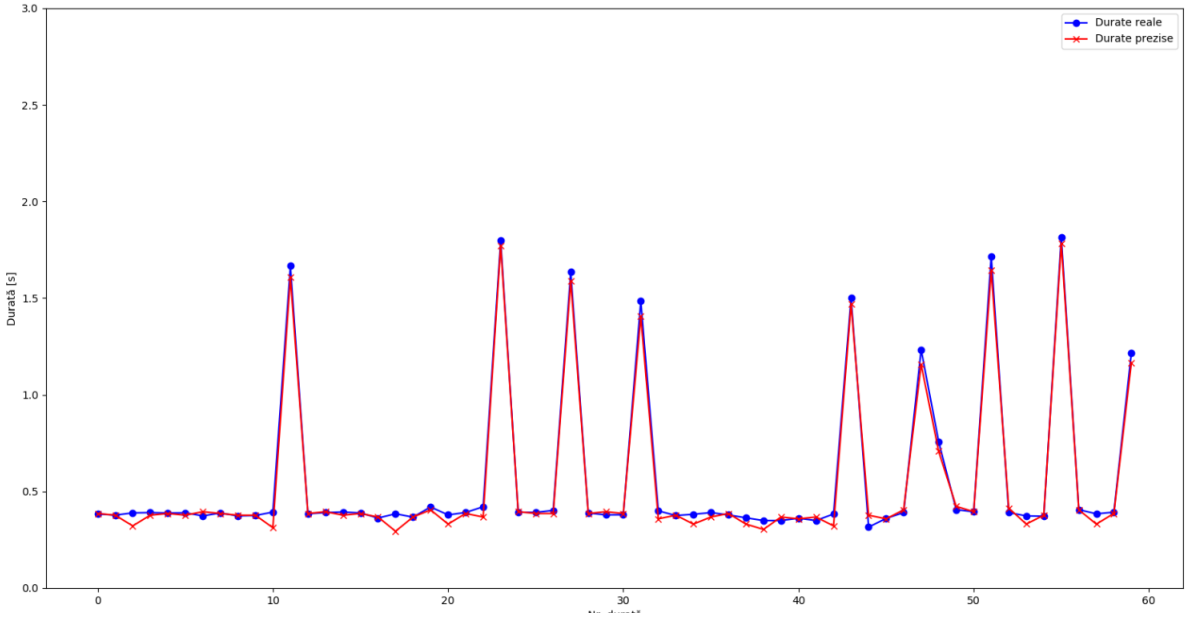


Figura 4.2: Comparația duratelor prin metoda tranzițiilor

4.2.2 Folosind a doua metodă

Acuratețea obținută pentru clasificare binară a onset-ului este de 98.01% pentru setul de antrenare, 98% pentru validare și 98.29% pentru melodia test.

Conform mărimilor pentru comparație, rezultatele obținute sunt:

$$acc_{onset} = 100\%, \text{prezise corect 60 din 60 de onset-uri} \tag{4.8}$$

$$acc_{offset} = 95\%, \text{prezise corect 57 din 60 de offset-uri} \tag{4.9}$$

$$MSE = 0.57 \times 10^{-3} \tag{4.10}$$

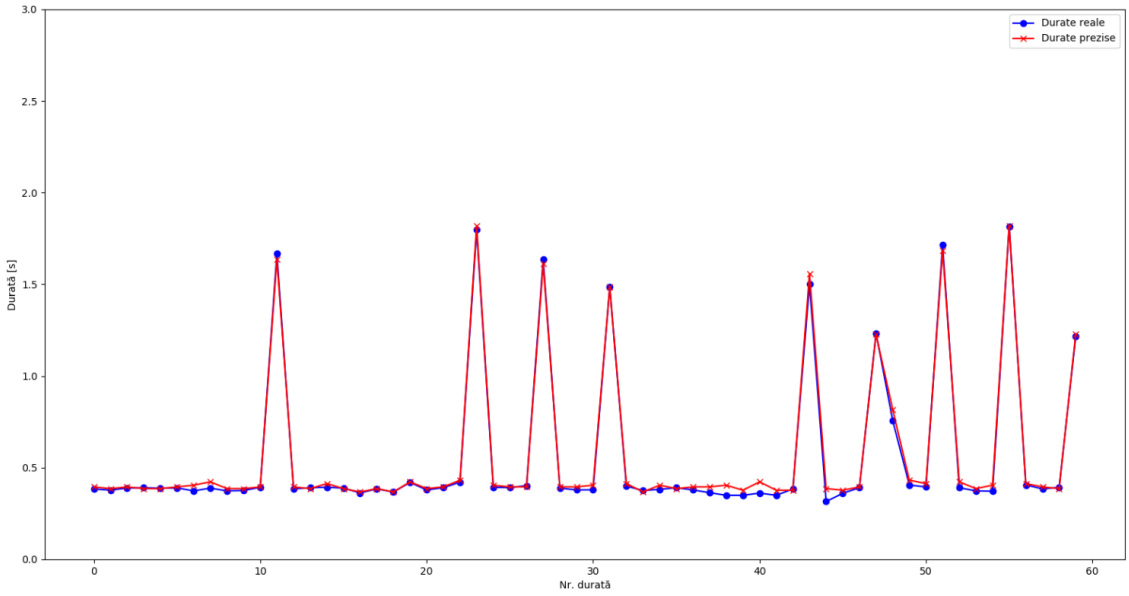


Figura 4.3: Comparația duratelor prin metoda clasificării binare a onset-ului

Se remarcă, atât vizual din figura 4.3, cât și prin analiza acurateții și a erorii pătratice, faptul că cea de-a doua metodă oferă rezultate mai bune în comparație cu prima metodă, în special în prezicerea unui offset cât mai apropiat de cel adevărat.

4.3 Estimarea intensității

Pentru comparația rezultatelor și la această sarcină, vom defini aceleași mărimi ca la *detecția onset / offset*, cu precizarea că o intensitate va fi considerată corectă dacă diferența dintre cea reală și prezisă este maxim 10:

$$acc(i) = \begin{cases} 1, & \text{pentru } \mid intensitate_{\text{reală}} - intensitate_{\text{prezisă}} \mid \leq 10 \\ 0, & \text{în rest} \end{cases} \tag{4.11}$$

$$acc_{intensitate} = \frac{1}{N} \sum_{i=1}^{N} acc(i) \times 100 \tag{4.12}$$

$$MSE = \frac{1}{N} \sum_{k=1}^{N} (intensitate_{\text{reală}} - intensitate_{\text{prezisă}})^2 \tag{4.13}$$

unde N este numărul de note din melodia test.

Pentru obținerea intervalelor pentru fiecare notă se va folosi metoda clasificării binare a onset-ului, datorită performanțelor mai bune.

4.3.1 Metode deterministe

Metoda puterii

Utilizând mărimile de comparație, se obțin rezultatele:

$$acc_{intensitate} = 86.67\%, \text{fiind prezise corect 52 din 60 de intensități} \tag{4.14}$$

$$MSE = 63.017 \tag{4.15}$$

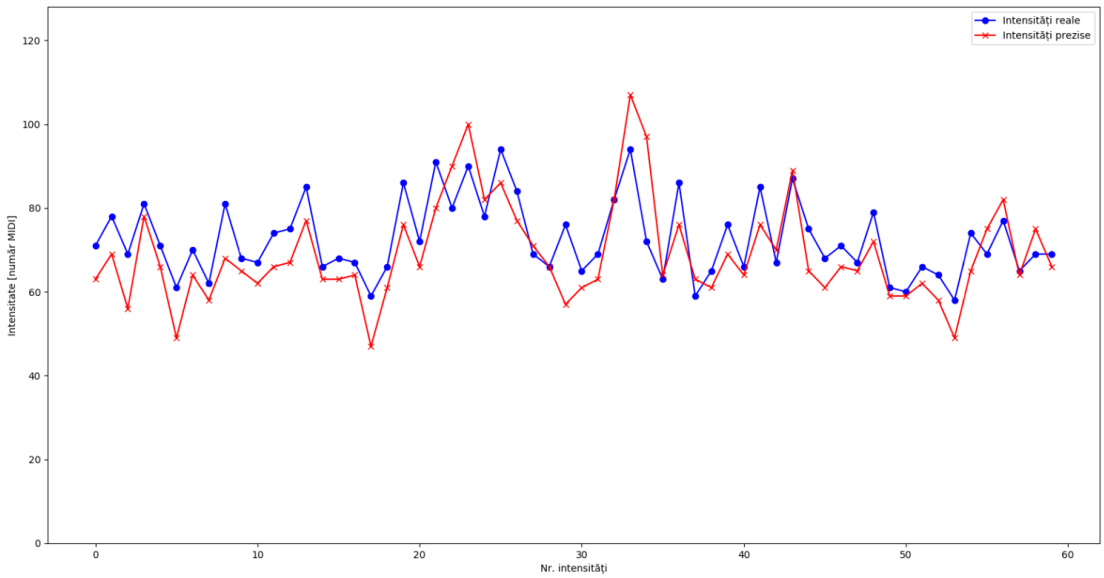


Figura 4.4: Comparația intensităților prin metoda puterii

Din figura 4.4 se observă faptul că această metodă urmărește relativ apropiat curba intensităților notelor, dar există vârfuri fie prea înalte fie prea mici pe anumite note ce determină o eroare pătratică medie mare.

Metoda amplitudinii maxime

Pentru această metodă, conform mărimilor pentru comparație, obținem:

$$acc_{intensitate} = 78.33\%, \text{prezise corect 47 din 60 de intensități} \tag{4.16}$$

$$MSE = 76.73 \tag{4.17}$$

În urma acestor rezultate se observă faptul că mai multe intensități sunt în afara diferenței limite de 10, precum și o eroare pătratică medie puțin mai mare.

După vizualizarea intensităților din figura 4.5 obținute prin metoda amplitudinii, putem spune că față de metoda puterii vârfurile de intensități false sunt mai reduse, dar per total metoda puterii urmărește mai bine variațiile intensității. Performanțele celor două metode sunt destul de apropiate, metoda puterii obținând rezultate puțin mai bune dar în detrimentul folosirii unui calcul mai complex și pe ferestre de semnal suprapuse, pe când cea de-a doua metodă se bazează strict pe maximul de amplitudine obținut în perioada de atac a notei.

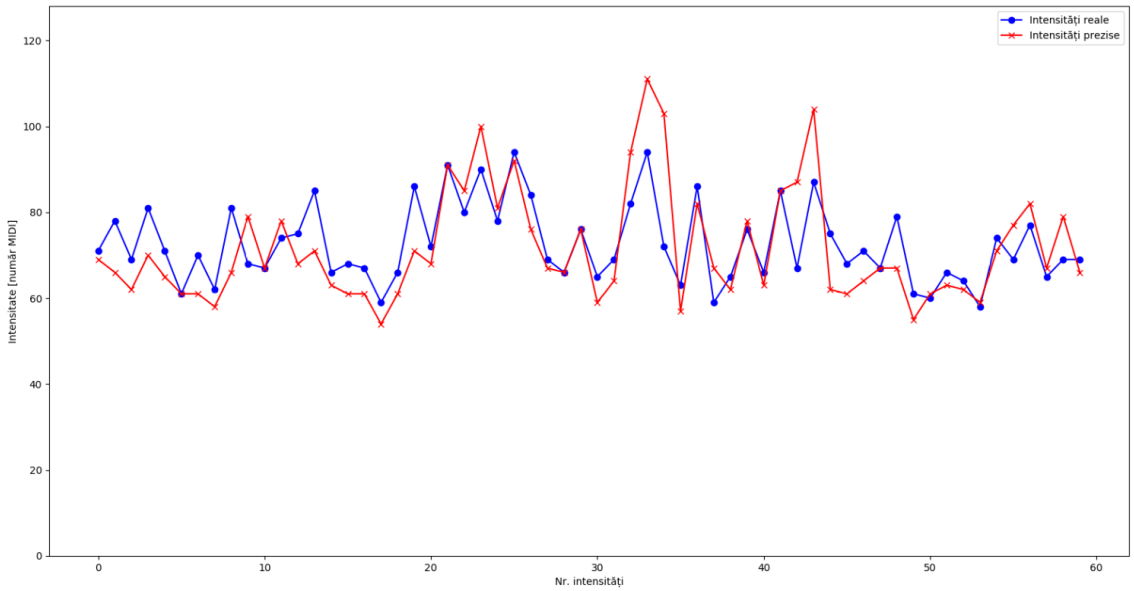


Figura 4.5: Comparația intensităților prin metoda amplitudinii

4.3.2 Metoda regresivă

Eroarea pătratică medie obținută de rețeaua de regresie pentru setul de antrenare este de 0.11×10^{-3} , pentru setul de validare de 2×10^{-3}, respectiv 0.25×10^{-3} pentru melodia de test. Folosind mărimile pentru comparație, obținem:

$$acc_{intensitate} = 100\%, \text{prezise corect 60 din 60 de intensități} \tag{4.18}$$

$$MSE = 5.85 \tag{4.19}$$

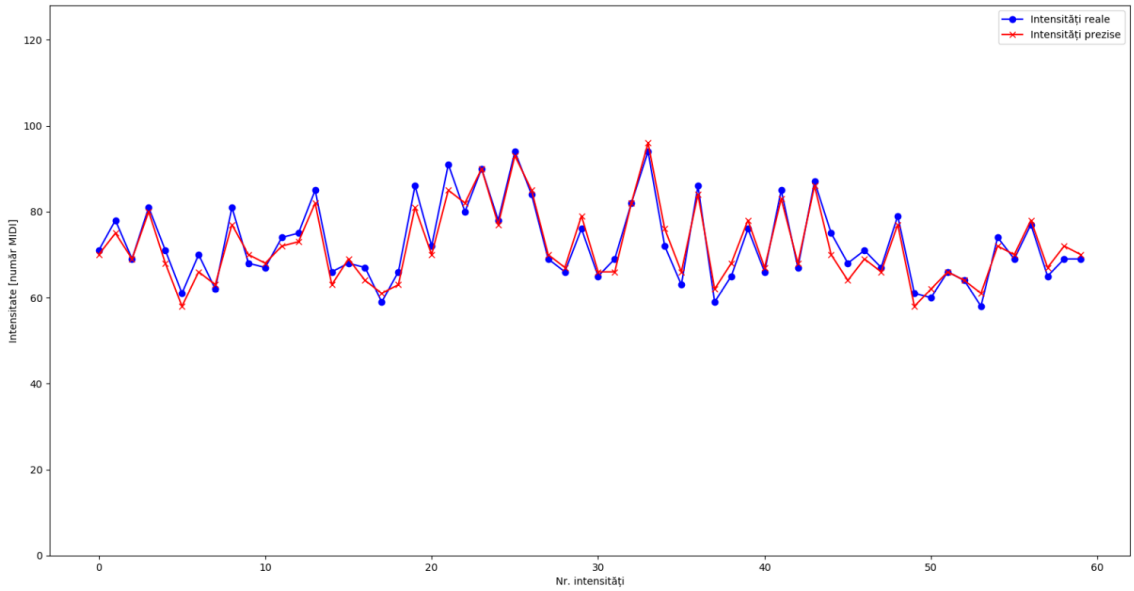


Figura 4.6: Comparația intensităților prin metoda regresivă

Se remarcă din figura 4.6, dar și prin mărimile de comparație, faptul că această metodă, bazată pe rețele neurale, este net superioară metodelor deterministe prezentate anterior.

După realizarea tuturor acestor etape, informațiile obținute vor fi înregistrate într-un fișier `MIDI`, ce se poate converti într-un portativ, ilustrat în figura 4.7:

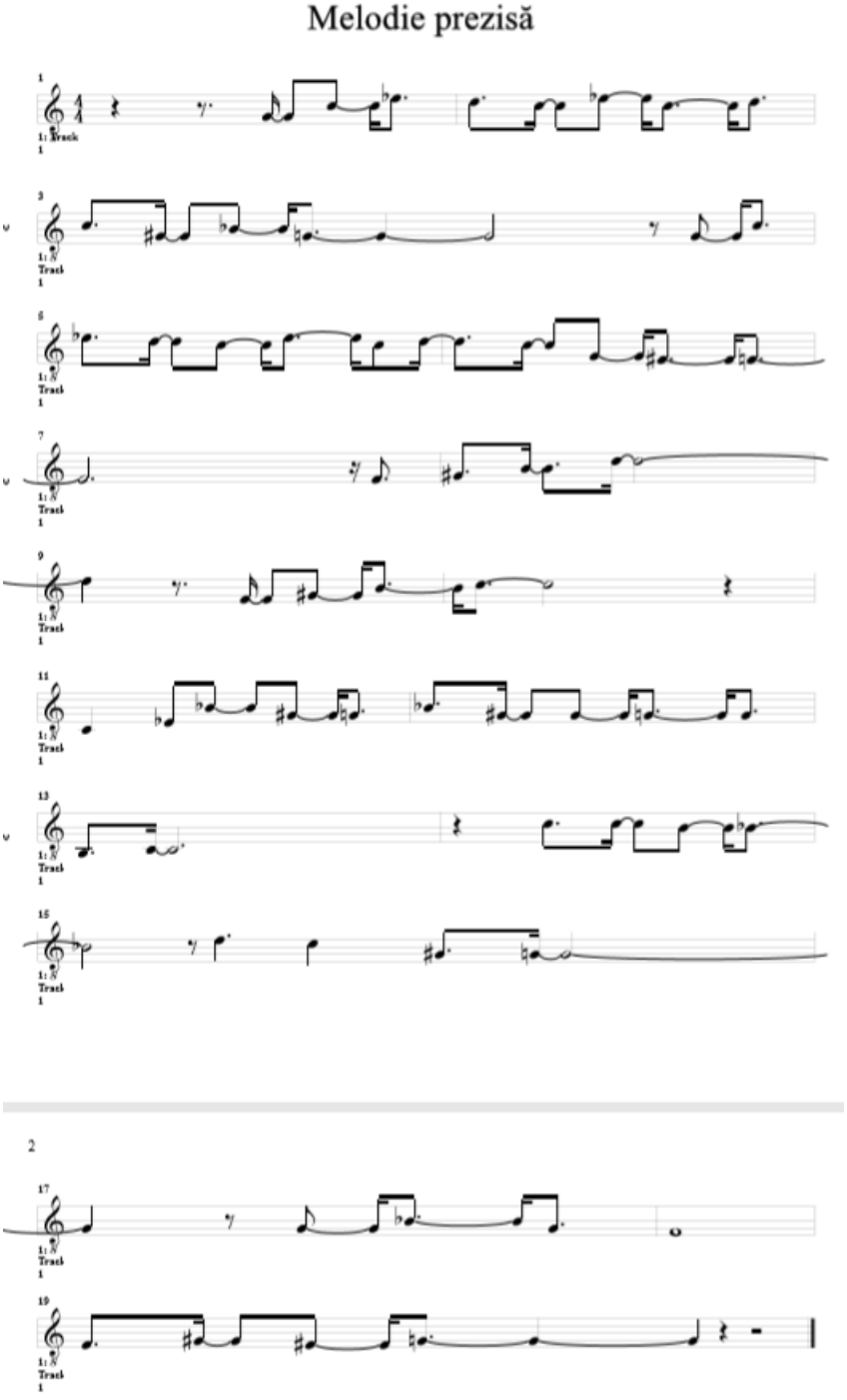


Figura 4.7: Portativul obținut pentru melodia de testare

Capitolul 5
Concluzii

5.1 Concluzii generale

Acest proiect de diplomă permite transcrierea automată a unei melodii monofonice utilizând rețele neurale pentru a extrage informațiile esențiale din melodia respectivă:

- *Înălțimile notelor* – identificate de rețea prin folosirea ferestrelor de semnal pur suprapuse, în scopul determinării frecvențelor fundamentale ale acestora;
- *Duratele notelor* – obținute prin detectarea momentelor de apariție a notelor și calculul fluxului temporal pe cadre de note succesive pentru determinarea sfârșitului fiecărei note;
- *Intensitățile notelor* – determinate prin introducerea unor ferestre de semnal, ce conțin perioadele de atac ale notelor, într-o rețea de regresie.

În concluzie, sistemul realizat poate reconstitui o melodie monofonică folosind doar semnalul audio și un număr suficient de straturi și neuroni pentru rețeaua neurală. Acest lucru reprezintă un avantaj enorm, deoarece nu sunt necesare alte transformări pentru a evidenția anumite proprietăți specifice semnalului, fiind suficientă doar o postprocesare după prezicerea rețelelor. Melodiile prezise sunt stocate în fișiere `MIDI` și pot fi transcrise într-un portativ prin intermediul anumitor programe.

5.2 Contribuții personale

- Realizarea unei baze de date muzicale proprii;
- Citirea și scrierea fișierelor `MIDI`, convertirea în fișiere `.wav` și realizarea fișierelor `.txt` specifice, folosind multiple script-uri în `Python`;
- Dezvoltarea și antrenarea rețelelor neurale variind diferiți parametrii pentru îmbunătățirea performanțelor, precum și testarea acestora;
- Preprocesarea semnalelor și postprocesarea predicțiilor date de rețele.

5.3 Dezvoltări ulterioare

Pe viitor, se dorește utilizarea rețelelor convoluționale (*eng. Convolutional Neural Networks*, pe scurt, CNN) și a rețelelor recurente (*eng. Recurrent Neural Networks*, prescurtat RNN) pentru a determina apariția unei note și sfârșitul acesteia, precum și pentru estimarea intensității.

De asemenea, se va studia transcrierea automată a melodiilor polifonice, domeniu mult mai complex și încă nerezolvat, fiind încă departe de performanțele umane.

Bibliografie

[1] D. Burileanu. Prelucrarea digitală a semnalelor. *Note de curs, UPB-ETTI*, 2019.

[2] `http://rf-opto.etc.tuiasi.ro/docs/files/RRCS_cap%202.pdf`.

[3] `http://www.afahc.ro/ro/facultate/cursuri/SCE_curs_vol_1.pdf`.

[4] D. Burileanu. Tehnici avansate de prelucrare digitală a semnalelor. *Note de curs, UPB-ETTI*, 2019.

[5] Keyth Wyatt and Carl Schroeder. Harmony and theory. *Musicians Institute Press*, 1998.

[6] `http://www.physics.pub.ro/Cursuri/Nicoleta_Eseanu_-_Fizica_(Fac.Transp.,IA)_(2010)/Cap.V.Acustica.pdf`.

[7] `http://www.fiziologie.ro/didactic/2019-2020/cursuri/Analizatorul%20auditiv2019.pdf`.

[8] `https://www.cochlear.com/ro/home/understand/hearing-and-hl/how-hearing-works`.

[9] Jong Wook Kim. *Automatic Music Transcription in the Deep Learning Era: Perspectives on Generative Neural Networks*. PhD thesis, New York University, 2020.

[10] Michael A Casey, Remco Veltkamp, Masataka Goto, Marc Leman, Christophe Rhodes, and Malcolm Slaney. Content-based music information retrieval: Current directions and future challenges. *Proceedings of the IEEE*, 96(4):668–696, 2008.

[11] Beth Logan et al. Mel frequency cepstral coefficients for music modeling. In *Ismir*, volume 270, pages 1–11, 2000.

[12] Christian Schörkhuber and Anssi Klapuri. Constant-q transform toolbox for music processing. In *7th Sound and Music Computing Conference, Barcelona, Spain*, pages 3–64, 2010.

[13] Christopher Harte and Mark Sandler. Automatic chord identifcation using a quantised chromagram. In *Audio Engineering Society Convention 118*. Audio Engineering Society, 2005.

[14] Juan Pablo Bello, Laurent Daudet, Samer Abdallah, Chris Duxbury, Mike Davies, and Mark B Sandler. A tutorial on onset detection in music signals. *IEEE Transactions on speech and audio processing*, 13(5):1035–1047, 2005.

[15] Ali Taylan Cemgil, Bert Kappen, Peter Desain, and Henkjan Honing. On tempo tracking: Tempogram representation and kalman filtering. *Journal of New Music Research*, 29(4):259–273, 2000.

[16] Peter Grosche, Meinard Müller, and Frank Kurth. Cyclic tempogram—a mid-level tempo representation for musicsignals. In *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5522–5525. IEEE, 2010.

[17] Rachel M Bittner, Brian McFee, Justin Salamon, Peter Li, and Juan Pablo Bello. Deep salience representations for f0 estimation in polyphonic music. In *ISMIR*, pages 63–70, 2017.

[18] Richard Vogl, Matthias Dorfer, Gerhard Widmer, and Peter Knees. Drum transcription via joint beat and drum modeling using convolutional recurrent neural networks. In *ISMIR*, pages 150–157, 2017.

[19] Sergio Oramas, Oriol Nieto, Francesco Barbieri, and Xavier Serra. Multi-label music genre classification from audio, text, and images using deep features. *arXiv preprint arXiv:1707.04916*, 2017.

[20] Jordi Pons Puig, Oriol Nieto, Matthew Prockup, Erik M Schmidt, Andreas F Ehmann, and Xavier Serra. End-to-end learning for music audio tagging at scale. In *Proceedings of the 19th International Society for Music Information Retrieval Conference, ISMIR 2018; 2018 Sep 23-27; Paris, France. p. 637-44.* International Society for Music Information Retrieval (ISMIR), 2018.

[21] Laurent Oudre, Yves Grenier, and Cédric Févotte. Template-based chord recognition: Influence of the chord types. In *ISMIR*, pages 153–158, 2009.

[22] Taemin Cho, Ron J Weiss, and Juan Pablo Bello. Exploring common variations in state of the art chord recognition systems. In *Proceedings of the Sound and Music Computing Conference (SMC)*, pages 1–8. Citeseer, 2010.

[23] Maksim Khadkevich and Maurizio Omologo. Use of hidden markov models and factored language models for automatic chord recognition. In *ISMIR*, pages 561–566, 2009.

[24] Emmanouil Benetos and Simon Dixon. Polyphonic music transcription using note onset and offset detection. In *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 37–40. IEEE, 2011.

[25] Tian Cheng, Matthias Mauch, Emmanouil Benetos, Simon Dixon, et al. An attack/decay model for piano transcription. ISMIR, 2016.

[26] Andrea Cogliati and Zhiyao Duan. Piano music transcription modeling note temporal evolution. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 429–433. IEEE, 2015.

[27] Anssi P Klapuri. Multiple fundamental frequency estimation based on harmonicity and spectral smoothness. *IEEE Transactions on Speech and Audio Processing*, 11(6):804–816, 2003.

[28] Chunghsin Yeh, Axel Roebel, and Xavier Rodet. Multiple fundamental frequency estimation and polyphony inference of polyphonic music signals. *IEEE transactions on audio, speech, and language processing*, 18(6):1116–1126, 2009.

[29] Valentin Emiya, Roland Badeau, and Bertrand David. Multipitch estimation of piano sounds using a new probabilistic spectral smoothness principle. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(6):1643–1654, 2009.

[30] A Michael Noll. Cepstrum pitch determination. *The journal of the acoustical society of America*, 41(2):293–309, 1967.

[31] John Dubnowski, Ronald Schafer, and Lawrence Rabiner. Real-time digital hardware pitch detector. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 24(1):2–8, 1976.

[32] Myron Ross, Harry Shaffer, Andrew Cohen, Richard Freudberg, and Harold Manley. Average magnitude difference function pitch extractor. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 22(5):353–362, 1974.

[33] David Talkin and W Bastiaan Kleijn. A robust algorithm for pitch tracking (rapt). *Speech coding and synthesis*, 495:518, 1995.

[34] Paul Boersma. Accurate short-term analysis of the fundamental frequency and the harmonics-to-noise ratio of a sampled sound. In *Proceedings of the institute of phonetic sciences*, volume 17, pages 97–110. Amsterdam, 1993.

[35] Alain De Cheveigné and Hideki Kawahara. Yin, a fundamental frequency estimator for speech and music. *The Journal of the Acoustical Society of America*, 111(4):1917–1930, 2002.

[36] Arturo Camacho and John G Harris. A sawtooth waveform inspired pitch estimator for speech and music. *The Journal of the Acoustical Society of America*, 124(3):1638–1652, 2008.

[37] Matthias Mauch and Simon Dixon. pyin: A fundamental frequency estimator using probabilistic threshold distributions. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 659–663. IEEE, 2014.

[38] Adrian von dem Knesebeck and Udo Zölzer. Comparison of pitch trackers for real-time guitar effects. In *Proc. 13th Int. Conf. Digital Audio Effects*, 2010.

[39] Onur Babacan, Thomas Drugman, Nicolas d'Alessandro, Nathalie Henrich, and Thierry Dutoit. A comparative study of pitch extraction algorithms on a large variety of singing sounds. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 7815–7819. IEEE, 2013.

[40] Valentin Emiya, Nancy Bertin, Bertrand David, and Roland Badeau. Maps-a piano database for multipitch estimation and automatic transcription of music. 2010.

[41] `http://web.cecs.pdx.edu/~ssp/Slides/Windowingx4.pdf`.

[42] `http://scs.etc.tuiasi.ro/iciocoiu/courses/CIPS/course7/Capitolul2.pdf`.

[43] `https://www.slideshare.net/zoevictoriaharris/handout-of-sound`.

[44] Alexander Lerch. *Software-based extraction of objective parameters from music performances*. PhD thesis, 2008.

[45] Ana Neacsu, Jean-Christophe Pesquet, and Corneliu Burileanu. Accuracy-robustness trade-off for positively weighted neural networks. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8389–8393. IEEE, 2020.