UNIVERSITATEA BABEŞ-BOLYAI CLUJ-NAPOCA

FACULTATEA DE MATEMATICĂ ŞI INFORMATICĂ

SPECIALIZAREA INFORMATICĂ ROMÂNĂ

LUCRARE DE LICENȚĂ

RECUNOAȘTEREA AUTOMATĂ A PARTITURILOR MUZICALE

Conducător științific
Grigoreta Sofia Cojocar

Absolvent
Răzvan Cosmin Linca

2020

Cuprins

Capitolul 1

Introducere

Subiectul propus urmărește prezentarea și implementarea unui proces complex obținut prin îmbinarea a două domenii aparent diferite: automatizarea unui proces ce ține de domeniul muzical și inteligența artificială, cu precădere metodele ce stau la baza învățării supervizate.

Prima parte (automatizarea recunoașterii unei partituri muzicale), definește procesul automat capabil să analize o mostră muzicală, să determine succesiunea continuă de acorduri diferite și să reprezinte automat această succesiune într-un format standard, numit tabulatură muzicală (format specific acordurilor acustice).

Analiza unei mostre muzicale și extragerea unor trăsături specifice se realizează aplicând algoritmi de procesare pentru semnalul audio, algoritmi care vor fi analizați și discutați separat și prin comparație în următoarele capitole.

Partea a doua (inteligența artificială), reprezintă unealta care stă la baza proceselor de învățare și deosebire a trăsăturilor unor acorduri din cadrul unei mostre muzicale. Se vor enunța și prezenta în detaliu câțiva algoritmi de învățare automată cu aplicabilitate pentru problema recunoașterii automate a partiturilor muzicale, cu scopul de a ajunge treptat la un algoritm complex și de actualitate pentru acest domeniu, capabil să analizeze automat semnalul audio, să clasifice cu o precizie cât mai mare fiecare secvență din cadrul unei piese acustice, fără a fi necesară intervenția umană în corectarea rezultatului.

1.1 Motivația

Muzica acustică contemporană se află într-o perioadă de creștere din punct de vedere tehnologic, odată cu evoluția paralelă a rețelelor de socializare(platforme video și de streaming, platforme de socializare ș.a.m.d), care au conectat artiști profesioniști și amatori ai domeniului.

Această evoloție a determinat o depărtare a interpretului de partitură, acesta fiind ori un artist profesionist care învață un cântec prin simpla ascultare sau vizualizare a unei înregistrări, sau un amator pasionat care învață prin urmărirea repetată a unor tutoriale aflate pe platformele online. De altfel, este cunoscut faptul că mulți compozitori și chitariști renumiți nu puteau să citească sau să scrie partituri muzicale, bazându-se pe talentul muzical în a memora fragmente sau a improviza pe loc diverse ritmuri. Printre ei se numără Jimi Hendrix, Eric Clapton, Elvis Presley sau interpreții trupei The Beatles [23].

Chitariștii susțin că notația clasică prin partitură muzicală este foarte bine orientată către instrumente cu clape, cum ar fi pianul, fiind în general neutilizată pentru chitară. Pe de altă parte, notația prin tabulatură este optimizată pentru instrumentele cu corzi și freturi, fiind de folos mai mare pentru aceștia. Cu toate acestea, o cantitate uriașă de piese au fost scrise doar în notația clasică prin partitură, iar transcrierea unei piese de la partitură la o tabulatură corectă este un proces complex care necesită cunoștințe de specialitate. Prin urmare, materialul devine inaccesibil pentru chitariștii aspiranți care nu sunt atât de specializați.

Principala motivație a realizării proiectului are astfel la bază dorința de a automatiza procesul de transcriere a conținutului audio direct într-o reprezentare simplă și sugestivă, folosind notația prin tabulatură. Acest proces ar trebui să fie unul simplu și rapid, sistemul fiind capabil să prezică în timp real ce acord este cântat exact în acel moment, determinând o fluiditate ridicată a aplicației.

O motivație secundară are la bază provocarea lansată de îmbinarea dintre cele două domenii aparent diferite: muzica, prin parcurgerea unor algoritmi de procesare a semnalului sonor, și inteligența artificială, cu focus pe înțelegerea rețelelor neuronale profunde. Este o oportunitate pentru oricine abordează aceste subiecte de a acumula infomații, abilități și deprinderi noi, prin analiza și studierea inițială a domeniilor, iar a apoi prin propunerea și descrierea unei soluții.

1.2 Analiza proiectului

Recunoașterea automată a partiturilor muzicale a fost un domeniu cercetat în mod activ în domeniul obținerii informațiilor muzicale, *Music Information Retrieval (MIR)*, în ultimii 20 de ani. Această categorie de algoritmi sunt o parte esențială a multor aplicații muzicale, cum ar fi sisteme de transcriere automată pentru diverse instrumente, aplicații de învățare în cadrul educației muzicale sau algoritmi de recomandare a muzicii sau a genurilor muzicale.

Acest domeniu urmează, în general, o paradigmă de calcul precisă. Acestea implică două faze distincte:

- Faza 1: Extragerea unor descriptori specifici din semnalul audio. Această etapă poartă numele de *extragerea trăsăturilor (feature extraction)*. Extragerea și analiza ulterioară a trasăturilor are la bază algoritmi de prelucrarea și procesare a semnalului audio. Domeniul de procesare a semnalului oferă diferite reprezentări posibile ale unui semnal audio care pot fi utilizate pentru probleme de clasificare, cum ar problema de recunoaștere a acordurilor acustice muzicale.

  Se consideră, spre exemplu, utilizarea Transformatei Fourier în prelucrarea semnalului sonor. Procesul de extracție a caracteristicilor se încheie odată cu crearea unei reprezentări sugestive a semnalului de tip *chromagrama*. Chromagrama oferă o descriere suficient de potrivită din punct de vedere muzical a semnalului audio, fiind utilizată, în acest caz, pentru a determina cel mai probabil acord care se cântă în acel interval de timp.

- Faza 2: Clasificarea acordurilor pe baza reprezentării determinate în prima fază. Această etapă explorează diferite metode de clasificare cu aplicabilitate pentru problema enunțată.

  Se vor descrie și analiza în detaliu mai multe metode specifice, pe baza unor cercetări conexe ale domeniului. Astfel, se vor descrie cronologic algoritmi potriviți, pornind de la modelele probabilistice, ca modelele Markov cu stări ascunse, fiind prima abordare a problemei enunțată în mai multe lucrări științifiice, până la construirea și optimizarea unor rețele neuronale convoluționale, utilizate recent în acest domeniu, acestă metodă fiind și cea aleasă pentru implementarea unei soluții, care va fi descrisă pas cu pas.

  Pentru a exemplifica cât mai clar și concis modul de funcționare al algoritmului, se va construi o aplicație mobile, care va fi capabilă să afișeze, în timp real, rezultatele recunoașterii automate de acorduri acustice, pentru orice partitură muzicală dorită.

Capitolul 2

Noțiuni introductive

În acest capitol sunt prezentate principalele concepte din teoria muzicală, concepte necesare în înțelegerea modalităților de prelucrare și reprezentare ale sunetelor.

2.1 Acorduri muzicale

2.2 Tabulatura

Capitolul 3

Metode de procesare ale semnalului sonor

În acest capitol sunt studiate diferite metode de procesare low-level a semnalului sonor, cu scopul obținerii unor caracteristici și a unei reprezentări care va fi utilizată mai departe în definirea și antrenarea unui model neuronal, prin învățare automată.

3.1 Semnalul sonor

Sunetul, prin definiție [19], este un semnal de tip mono-dimensional, dependent de timp, reprezentând presiunea aerului asupra canalului uman auditiv. Sistemul auditiv uman este capabil să perceapă orice semnal sonor care se află în intervalul de frecvență 20Hz - 20000Hz (=20kHz).

Frecvența se definește ca numărul de repetări ale unui fenomen sau eveniment periodic într-un interval de timp. Unitatea de măsură pentru frecvență este Hertz, simbolizat ca Hz, denumită astfel în cinstea fizicianului german Heinrich Hertz. Astfel, o frecvență f = 1 Hz este corespunzătoare unei perioade de timp de o secundă.

Putem astfel afirma că dacă un anumit eveniment se repetă la un interval de timp T, putem calcula frecvența f ca:

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

In ceea ce privește sunetul, frecvența este legată de noțiunea de înălțime muzicală. Astfel, pentru nota LA din gama centrală se definește frecvența de 440 Hz, ceea ce înseamnă că aerul pus în mișcare de unda sonoră corespunzătoare oscilează de 440 de ori pe secundă.

Sunetul capturat de către un microfon este o undă dependentă de timp, determinând variația presiunii aerului în câmpul sonor în care se află microfonul. Astfel, un semnal audio digital este obținut prin prelevarea și cuantificarea adecvată a ieșirii microfonului, reprezentat de unde electice. Deși orice frecvență peste 40kHz ar fi suficientă pentru a capta întreaga gamă de frecvențe perceptibile, rata de preluare utilizată pe o scară largă este de 44.100 Hz, stabilită în urma nevoii de a sincroniza sunetul cu datele de tip video.

Calitatea de tip *CD* se referă la o mostră audio digitală cu frecvența de 44.1 kHz și 16-bit (Adâncimea de biți sau *Bit depth*, reprezintă numărul de biți de informație aflați în fiecare mostră audio).

Pentru a înțelege mai bine calitatea sunetului în funcție de adâncimea de biți observăm Figura 3.1, unde se prezintă prin comparație forma undei în funcție de valoarea adâncimii de biți.

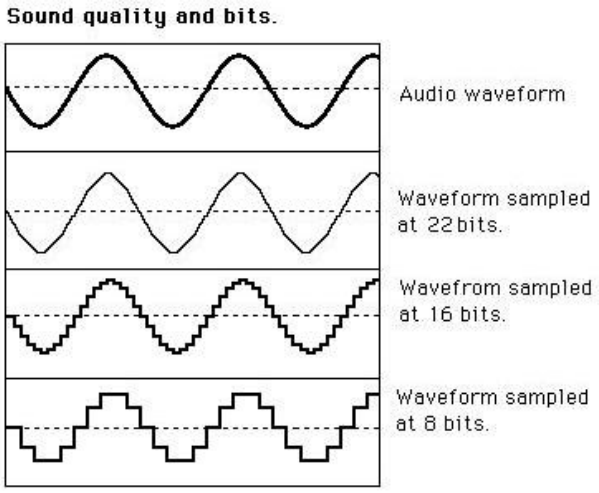

Figura 3.1: Calitatea undei sonore în funcție de adâncimea de biți [1].

Prin intensitate sonoră se înțelege senzația produsă de amplitudinea unei unde sonore, cunoscută și ca volumul vibrației, asupra organului uman auditiv. Cu cât amplitudinea este mai mare, cu atât crește și intensitatea sunetului care rezultă. Intensitatea sonoră se măsoară în unități denumite în fizică decibeli(dB) sau foni (un decibel este echivalent cu un fon).

Auzul uman este limitat de un interval în ceea ce privește intensitatea sunetului, și anume:

1. Marginea inferioară:
    - prag auditiv cu valoarea de 0 dB;
    - sunete foarte slabe ca intesitate, cum ar fi: sunete din natură, nivelul de zgomot dintr-o bibliotecă, cu valorea cuprinsă între 10-20 dB.
2. Marginea superioară:
    - sunete extrem de puternice ca intensitate, cum ar fi: decolarea unui avion cu reacție, cu valorea cuprinsă între 120-130 dB;
    - prag dureros cu valorea de 140 dB [11].

3.2 Transformata Fourier pe termen scurt

Pentru a putea înțelege metoda transformatei fourier pe termen scurt, este nevoie de definirea bazelor matematice ale Transformatei Fourier (FT) și procesele care compun metoda.

3.2.1 Interpretare matematică

Se consideră, drept exemplu, un semnal pur cu frecvența f de 3Hz, pe un interval de timp vizibil. (Figura 3.2)

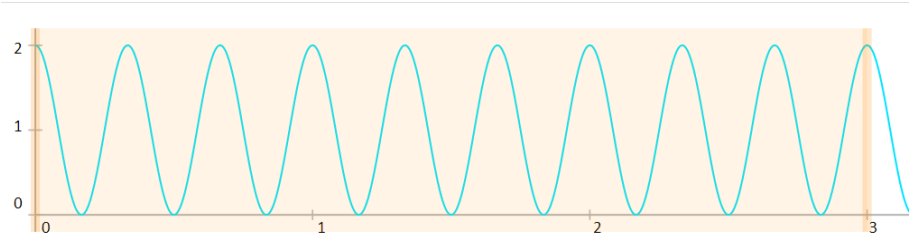

Figura 3.2: Undă sonoră cu frecvența de 3Hz [10].

Se transpune această undă dependentă de timp în jurul unui cerc prin intermediul unei corzi, în așa fel încât valorile extreme ale funcției din prima reprezentare să fie asociate cu distanțele maxime în raport cu originea din cea de-a doua reprezentare. De aici se observă că trebuie luată în considerare o nouă frecvență, și anume frecvența de realizarea a unui rotații complete de cerc parcurgând coarda, notată *fcerc*.

Se presupune că această coardă are o masă și se consideră punctul central din figură ca fiind *centrul de masă* al corzii. Se observă că odată cu modificare frecvenței cerc, se modifică și poziția centrului de masă, care se situează într-o vecinătate restrânsă a originii, cu o singură excepție: momentul în care frecvența de realizare a unei rotații în jurul cercului, *fcerc*, este egală cu frecvența undei sonore, notată *f*.

Punctul specific centrului de masă poate fi reprezentat sub forma unui număr complex de forma $xi + yj$. Se alege o reprezentare de tip complex în detrimentul unei reprezentări standard în coordonate (x,y) deoarece numerele complexe oferă o descriere potrivită pentru rotiri și ondulări raportate la un cerc de rază, notat r. De exemplu, formula lui Euler, una dintre cele mai faimoase formule trigonometrice($e^x = cos(x) + isin(x)$), enunță faptul că e^{ix} trasează cercul unitate din planul numerelor complexe, când x ia valori reale. Astfel, dacă *f=1/10*, atunci o rotație completă se va realiza odată la 10 secunde. De asemenea, convenția în cadrul transformatei Fourier este ca rotația să se realizeze în jurul acelor de ceasornic, expresia devenind $e^{-2\pi ift}$.

Fie o funcție $g(t)$ ce descrie un semnal dependent de intensitate și timp. Dacă se înmulțește $g(t)$ cu expresia construită se observă că punctul din planul complex se plimbă de sus în jos potrivit valorilor funcție $g(t)$. Astfel, expresia $g(t)e^{-2\pi ift}$ încapsulează întreaga idee de ondulare a funcției g în jurul unui cerc cu o variabilă pentru frecvența f.

Totuși, scopul este de a urmări mișcarea centrului de masă pe care îl are coarda. Pentru a aproxima centrul de masă pot fi considerate mai multe *mostre* din semnalul original reprezentat de funcția g, apoi urmărite pozițiile lor în reprezentarea din jurul cercului, și realizată o medie. Cu cât se iau în considerare mai multe puncte, cu atât rezultatul va fi mai precis. Generalizând, se consideră o integrală a acestei funcții. De la această integrală până la transformata Fourier mai este un singur pas, și anume trebuie considerată integrala de tip continuu, pe intervalul $(-\infty, \infty)$. Astfel, formula este următoarea:

$$F(\varepsilon) = \int_{-\infty}^{\infty} f(x) \cdot e^{-2\pi i \varepsilon x} dx$$

O proprietate importantă este forma continuă a tranformatei, întrucât integrala este definită pe intervalul de timp $(-\infty, \infty)$. În practică însă, colectarea de date audio se realizează într-un interval finit de timp (de la momentul de start t_0 la momentul $t_N - 1$), ceea ce implică calculului unei transformate de tip discret, *Discret Fourier Tranform*(DFT).

Transformata de tip discret are aceeași definiție ca transformata de tip continuu, diferența fiind doar în stabilirea unui interval finit cunoscut, calcul integralei fiind înlocuit de o sumă finită care are următoarea formă:

$$X(\varepsilon_k) = \sum_{n=0}^{N-1} f(t_n) \cdot e^{-2\pi \varepsilon_k t_n}, k = 0, 1, 2..., N-1$$

Semnificația fiecărei variabile din formulă este următoarea:

- f(t_n) - semnalul de intrare, la momentul n;
- $X(\varepsilon_k)$ - valoarea complexă a spectrului corespunzător lui x, la frecvența k;
- N - totalul semnalelor de intrarea (mostre);
- $t_n = nT$, unde T reprezintă intervalul de eșantionare a semnalului, fiind măsurat în secunde și n, o valoare de tip întreg, $n >= 0$;
- $f_s = \frac{1}{T}$ - reprezentând rata de eșantionare, măsurată în Hz; 1Hz = 10000 de eșantioane pe secundă, *samples per second*;
- $\varepsilon_k = k\Omega$, este al k-lea eșantion din totalul mostrelor de intrare;
- $\Omega = \frac{2\pi}{NT}$, interval de eșantionare cu frecvența radiană, rad/sec [4].

În literatură se obișnuiește ca T să fie egal 1 în ecuația precedentă si astfel se obține $t_n = n$. Atât semnalul $x(n)$ cât și transformarea $X(\varepsilon_k)$ sunt cantități discrete reprezentând N mostre. Semnalul audio este foarte non-staționar. Dezavantajul metodei DFT este că informația temporală, *temporal information*, se pierde. Din acest motiv, este necesar să se compună o analiză locală a frecvenței pe secțiuni și nu luând semnalul întreg [4]. Această metodă se numește *Transformata Fourier pe termen scurt (STFT)*. Procesul este ușor de realizat calculând DFT pe porțiuni ale semnalului, numite cadre sau frame-uri. Parametrii care trebuie specificați sunt:

- N_{FT}, dimensiunea totată a semnalului;
- $L, L <= N_{FT}$, dimensiunea unui cadru din semnal;
- saltul, *hop size*, notat Hop, reprezentând distanța în eșantioane între două cadre consecutive.

Astfel, formula de calcul pentru STFT este următoarea:

$$X_{stft}(\varepsilon_k, r) = \sum_{n=0}^{N_{FT}-1} (x(n - rHop)\varepsilon_n)e^{-2\pi\varepsilon_k n}, k = 0, 1, 2..., N_{FT}$$

3.2.2 Aplicabilitate

Recapitulând, transformata Fourier este o un tip de operație care se aplică unei funcții complexe și produce o altă funcție complexă care conține aceeași informație ca funcția originală, dar reorganizată dupa frecvențele componentelor.

Se dorește ca, utilizând principiile matematice ale transformatei Fourier enunțate mai sus, să se găsească o reprezentare a undei sonore, pe baza căreia se pot extrage caracteristici utile, aplicând transformata Fourier sau orice altă metodă derivată. Astfel, se condideră o funcție reprezentată de un semnal dependent de timp, care este chiar unda sonoră, dar limitată la un interval temporal. Transformata Fourier a funcției descompune semnalul după frecvență și produce un spectru al acestuia.

Proprietatea de bază a transformatei este dată de reorganizarea informației după frecvențe (temporale, spațiale sau de alt fel), fiind extrem de utilă în prelucrarea semnalelor de diverse tipuri, în înțelegerea proprietăților unui număr mare de sisteme fizice sau în rezolvarea unor ecuații și sisteme de ecuații [5]. De asemenea, transformata Fourier permite analiza semnalului și identificarea anumite frecvențe dorite, cu scopul de a le amplifica sau a le suprima, și astfel, determinându-se sintetizarea unui nou semnal.

Înzestrată cu aceste propietăți, dar și cu altele, transformata este prezentă în foarte multe tehnologii și domenii moderne(ultimul domeniu în care și-a găsit aplicabilitea fiind mecanica cuantică), deoarece este mai fiabilă și mai robustă decât alte tehnologii de analiză a semnalului de orice fel și compunere a spectrului.

Dezavantajul principal constă în dificultatea și complexitatea calculelor care determină întregul proces. În cazul integrării transformatei într-un program soft, programatorul trebuie să transpună procesul într-un algoritm eficient și să fie conștient de faptul că unele calcule utilizează procesoarele într-un mod intens.

La nivelul limbajelor de nivel înalt, cum ar fi Python, există librării care vin în ajutorul programatorilor cu metode de procesare audio implementate într-un mod eficient(de exemplu, utilizarea librăriei Librosa).

3.3 Transformata Q constantă

Principala problemă în aplicarea Transformatei Fourier în aplicații muzicale este faptul că intervalele de eșantionare ale frecvenței pentru o mostră muzicală sunt liniar distribuite, acest aspect fiind diferit în cadrul transformatei Q constantă, unde intervalele sunt distanțate în mod geometric.

Transformarea Q constantă (CQT) a fost introdusă de Brown în [6] și este asemănătoare transformatei Fourier. Utilitatea acestei transformări se bazează pe faptul că, având un mod adecvat de alegere a parametrilor, frecvențele f_k ale transformării vor corespunde în mod direct cu cele ale notelor muzicale. Frecvențele corespunzătoare se determină astfel prin formula:

$$f_k = f_0 \cdot 2^{\frac{k}{\beta}} \ , k = 0, 1, ...$$

unde, k reprezintă a k-a frecvență, f_0 corespunde celei mai mici frecvențe iar β reprezintă numărul de filtre dintr-o octavă.

De asemenea, valoarea pentru rezoluția în domeniul timp a transformatei Q, *time resolution*, crește odată cu creșterea frecvenței. Acest aspect este similar cu comportamentul sistemului uman auditiv [4]. Concluzia acestei proprietății este că atât simțul uman auditiv, cât și computerul digital, au nevoie de mai mult timp pentru a percepe frecvența unui ton muzical scăzut.

Metoda nu este foarte eficientă în ceea ce privește timpul total de compunere a frecventelor și de determinare a reprezentări, dar reprezintă o optimizarea a tranformatei Fourier clasice, fiind utilizată pe o scară mai largă, prin aplicarea diverselor strategii de calcul paralel pentru a scădea timpul de execuție, în cazul în care tranformata este aplicată pe un set de date numeros.

Librăria Librosa oferă o soluție eficientă și ușor de utilizat pentru calculul transformatei Q constantă a unui semnal audio, fiind nevoie doar de încărcarea semnalului și de cunoașterea parametriilor ca *hop length*, reprezentând valoare pentru salt, și opțional, *n chroma*, pentru a specifica numărul de semitonuri distincte dorite pentru octava muzicală (implicit, valoarea pentru acest parametru este 12). Astfel, o metodă simplă de calcul arată astfel:

```
def compute_chroma_cqt(audio_file_path):
    #   Se incarca fisierul audio folosind sistemul de incarcare
    # oferit de librosa
    audio, sr = librosa.load(audio_file_path)
    #   Determina chromagrama folosind transformarea Q constanta
    chromagram = librosa.feature.chroma_cqt(audio, sr=sr,
       hop_length=512, n_chroma=24)
```

3.4 Chromagrama

Chromagrama este reprezentarea unui grup de caracteristici pe care un program ce are la bază unul din algoritmii de procesare audio, îl extrage dintr-un semnal audio. Reprezentarea este bazată pe profilul notelor rădăcină, cunoscut ca *pitch class profile (PCP)*, acesta fiind un descriptor în contextul unui sistem de recunoaștere a acordurilor.

Chromagrama este formată astfel dintr-o secvență de vectori caracteristici, fiecare având rolul de a măsura intensitatea relativă a fiecărei note, raportată la fiecare frame, într-un interval de timp. Se obține astfel o diagramă de tipul pitch class versus time, adică o imagine a semnalului audio, ce conține informații concludente, care pot ajuta în determinarea celui mai probabil acord.

În literatură există diferite metode de a construi o chromagramă. Principalii pași enunțati de către Bello si Pickens în [18] vor fi folosiți ca bază de lucru și în procesele acestui algoritm. Pașii pentru compunerea chromagramei sunt următorii:

- Convertirea semnalului audio într-o reprezentare intermediară, numită spectogramă, prin aplicarea unei transformări Fourier derivată sau optimizată;
- Se aplică un filtru asupra frecvențelor din cadrul spectogramei, pentru a se încadra în intervalul 100Hz - 5000Hz;
- Se construiește profilul notelor rădăcină, pe baza frecvențelor estimate. Este o procedură pentru a determina nota rădăcina *(pitch class)*, din valorile frecvențelor;
- Se realizează o normalizare a caracteristicilor, cadru cu cadru, prin divizarea cu cea mai mare valoare, pentru a elimina zgomotul, rezultând imaginea chromagramei.

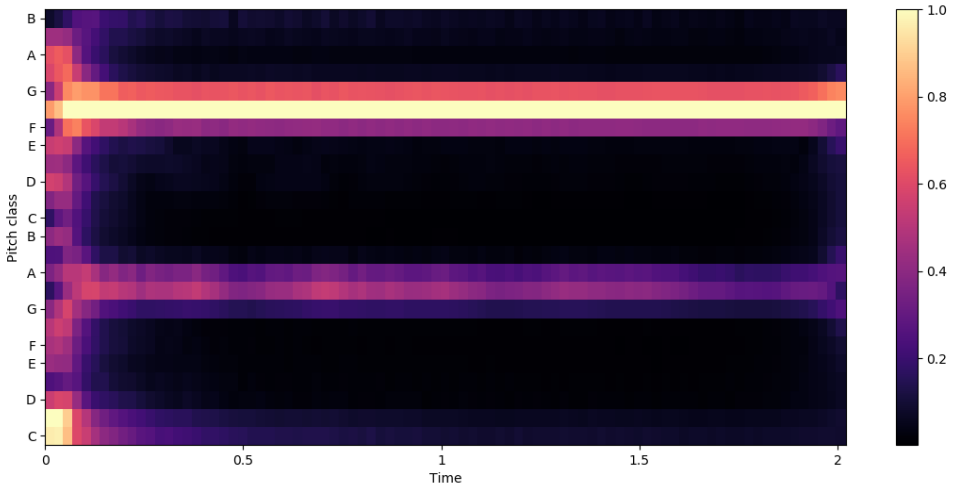

Figura 3.3: Imaginea obținută prin construirea chromagramei acordului A, reprezentare grafică utilizând Librosa și matplot.

Capitolul 4

Metode de clasificare

4.1 Sisteme de învățare automată

Învățarea automată, *machine learning*, este o ramură a Inteligenței Artificiale concentrată pe algoritmi specializați pe invățarea din date. Procesul de învățare constă în deducerea unor tipare, pe baza unor reguli bine stabilite. Mai precis, se urmărește crearea unui program complex capabil să generalizeze un comportament indiferent de datele de intrare.

Problemele clasice de învățare automată includ clasificare de imagini sau de înregistrări audio, recunoaștere vocală, evaluarea riscurilor financiare pentru diverse investiții, dezvoltarea unor strategii în jocuri sau simulări, predicția unor diagnostice medicale etc. În mod general, abordările algorimilor sunt structurate după obiectivul lor de învățare:

1. Învățare supervizată
   În învățarea supervizată, datele de antrenament care alimentează algoritmul includ predicția corectă, numite etichete, sau *labels*. Un exemplu de metodă clasică este *metoda clasificării*. O problemă clasică care are la bază clasificarea este problema filter-ului de email-uri spam. Aici antrenarea se realizează cu multe exemple de email-uri, fiecare din ele având clasa corespunzătoare atașată (spam sau not-spam). Astfel, modelul va învăța cum să clasifice orice mail nou întâlnit.
   O altă metodă specifică este *metoda regresiei*. Problemele de regresie au ca scop prezicerea unei valori numerice target, pe baza unor trăsături numite predictori(de exemplu, prezicerea prețului pentru o mașină pe baza unor trăsături ca brand, an de fabricație și distanța parcursă). Pentru a se realiza antrenarea, este necesară utilizarea unui set de predictori împreună cu etichetele corespunzătoare.
   Cei mai importanți algoritmi de învățare supervizată sunt următorii [12]:

   - Cei mai apropiați k vecini;
   - Modele Markov cu stări ascunse (HMMs) - *statistical learning*
   - Regresie liniară;

- Regresie logistică;
- Mașini cu suport vectorial (SVMs);
- Arbori de decizie;
- Rețele neuronale artificiale.
- Rețele neuronale profunde (CNNs, DBNs).

2. Învățare nesupervizată:
   În cadrul învățării nesupervizate datele utilizate pentru antrenament nu sunt etichetate. Astfel, în funcție de date și de domeniul problemei, algoritmii sunt împărțiți în subcategorii, după cum urmează:

   (a) Clustering: se definește ca un proces de organizare a obiectelor care sunt asemănătoare dintr-un anumit punct de vedere. Astfel, noțiunea de *cluster* definește o mulțime de obiecte similare între ele și diferite de obiectele care aparțin altui cluster. Pentru exemplificare, se consideră o problemă care dorește împărțirea pe clustere a vizitatorilor unui blog. În niciun moment dezvoltatorul sistemului nu va interveni pentru a ajuta algoritmul cu informații despre vizitatori. Conexiunile între utilizatori vor fi determinate în mod independent de către algoritm.
   Spre exemplu, se poate observa că 30% dintre utilizatori sunt bărbați care sunt pasionați de carți polițiste și în general citesc blog-ul în jurul amiezei, în timp ce 20% sunt tineri pasionați de sci-fi si citesc blog-ul în weekend etc. Astfel, dacă se utilizează un algoritm de clustering bazat pe ierarhii, *hierarchical clustering*, se pot realiza subdiviziuni pe baza acestor informații care pot ajuta la distribuirea ulterioară în clustere.

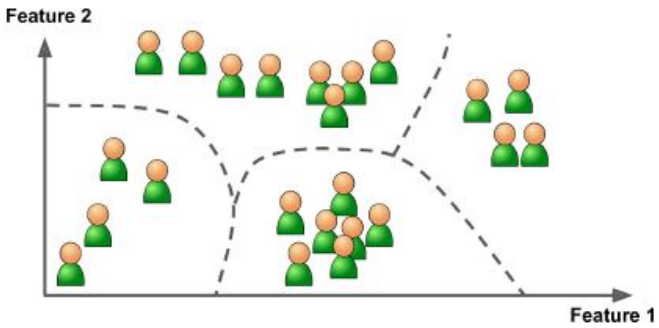

Figura 4.1: Abordarea generală a grupării în clustere [12].

   Cei mai cunoscuți algorimi de clustering sunt: *K-Means*, *Hierarchical clustering* și *Clustering bazat pe densitatea aplicațiilor cu zgomot (DBSCAN)*.

   (b) Vizualizare și reducere dimensionalității: algoritmii de vizualizare se definesc ca algoritmi care se încarcă cu numeroase date, iar pe baza lor se obține ca output un

grafic 2D sau 3D ca reprezentare a datelor, cu scopul de a înțelege cum sunt organizate acestea și posibilitatea descoperirii unor modele neașteptate. O sarcină conexă este reducerea dimensionalității, în care obiectivul este simplificarea datelor fără a pierde multe informații.

Spre exemplu, distanța parcursă de o mașină poate fi corelată cu vârsta sa, astfel algoritmul de reducere va îmbina cele două caracteristici într-o singură caracteristică ce reprezintă uzura mașinii.

Printre cele mai utilizate metode se numără: *Analiza componentelor principale(PCA)*, *Kernel PCA*, *Încorporarea linear locală(LLE)* [12].

(c) Reguli de asociere: învățarea prin intermediul regulilor de asociere are ca scop analiza cantităților mari de date și descoperirea relațiilor interesante între atribute.

De exemplu, să presupunem că deținem un supermarket. Efectuarea unei reguli de asociere pe jurnalele de vânzări dezvăluie faptul că persoanele care cumpără sos de friptură și cartofi, tind să cumpere și friptură. Astfel, este de dorit ca astfel de articole să fie plasate aproape unele de altele.

Printre algoritmii cei mai utilizați se numără: *Apriori* și *Eclat* [12].

3. Învățare semi-supervizată

   Învățarea semi-supervizată urmărește definirea unor algoritmi de învățare care au date de antrenament doar parțial etichetate. În majoritatea cazurilor, datele neetichetate predomină în setul de date. În mare parte algoritmii sunt construiți prin combinarea algorimiilor de învățare supervizată și nesupervizată.

   Spre exemplu, una din metode, rețelele *deep belief* au la bază componente nesupervizată numite mașini cu restricție Boltzmann, *restricted Boltzmann machines* (RBMs), componente așezate sub forma unei stive în definirea modelului [12]. RBM-urile sunt antrenate secvențial într-o manieră nesupravegheată, și apoi, întregul sistem este ajustat folosind tehnici de învățare supravegheată.

4. Învățare prin întărire

   Învățarea prin întărire este o metodă diferită de cele enunțate anterior. Sistemul de învățare, numit agent, este capabil să observe mediul, să selecteze și să efectueze acțiuni, în urma cărora primește *recompense*. (sau penalități, sub forma de recompense negative). Acest mecanism obligă sistemul să învețe de la sine cea mai bună strategie, numită *politică*, pentru a obține cu timpul cele mai bune recompense. O politică definește ce acțiuni să aleagă agentul atunci când se află într-o situație nouă, necunoscută.

4.2 Modele Markov cu stări ascunse

Modelele Markov cu stări ascunse, *Hidden Markov Models(HMM)*, se bazează pe determinarea unui lanț Markov. Un lanț Markov este un model care ne informează despre probabilitățile secvențelor unor variabile aleatorii, numite stări, fiecare putând prelua valori din cadrul unui set. Aceste seturi pot reprezenta orice, de la cuvinte până la diverse simboluri (de exemplu, simboluri care definesc vremea).

Un lanț Markov face o presupunere foarte puternică în ceea ce privește prezicerea viitorului într-o succesiune, starea actuală fiind singura care contează. Stările anterioare stării curente nu au niciun impact asupra previziunii din viitor. Spre exemplu, s-ar putea prezice vremea de mâine, examinând vremea de astăzi, fără a fi permisă analiza vremii de ieri.

Astfel, fie o secvență de stări $q_1, q_2, ...q_i$. Un model Markov încorporează presupunerea Markov în privința probabilităților acestei secvențe. Presupunerea Markov afirmă faptul că prezicerea viitorului depinde doar de prezent, fără a conta trecutul. Matematic, se definește astfel:

$$P(q_i|q_1...q_{i-1}) = P(q_i|q_{i-1})$$

Un lanț Markov este util atunci când trebuie calculată o probabilitate pentru o secvență de evenimente observabile. În multe cazuri însă, evenimentele care sunt de interes sunt ascunse/invizibile și nu pot fi observate în mod direct. Un model Markov cu stări ascunse permite abordarea ambelor tipuri de evenimente, observabile și ascunse, considerate drept factori cauzali în modelul probabilistic.

Modelele Markov cu stări ascunse sunt caracterizate de trei probleme fundamentale:

1. Probabilitatea (*Likelihood*): Fiind dat un HMM $\lambda = (A,B)$ și un set de observații O, să se determine probabilitatea $P(O|\lambda)$;

2. Decodarea (*Decoding*): Fiind dat un set de observații O și un HMM $\lambda = (A,B)$ să se determine cea mai bună secvență de stări ascunse Q;
   Cel mai comun algoritm de decodare este algoritmul lui Viterbi, *Viterbi algorithm*. Este un algoritm de programare dinamică pentru a găsi cea mai probabilă secvență de stări ascunse, numită calea Viterbi. Rezultatul este o succesiune de evenimente observate, mai ales în contextul modelelor Markov cu stări ascunse.

3. Învățarea (*Learning*): Fiind dat un set de observații O și un set de stări într-un HMM, să se determine prin învățare parametrii A și B.
   Algoritmul standard de antrenare a unui HMM este cunoscut ca *forward-backward* sau *Baum-Welch algorithm*, fiind un caz special a algoritmului *Expectation-Maximization(EM)*. Algoritmul permite antrenarea atât a paramentrilor de tranziție (A), cât și a celor de

emisie (B). EM este un algoritm iterativ care calculează estimări inițiale pentru probabilități, apoi folosește aceste estimări pentru a calcula o estimare mai bună, procesul continuând în acest fel, îmbunătățind iterativ probabilitățile pe care le învață [16].

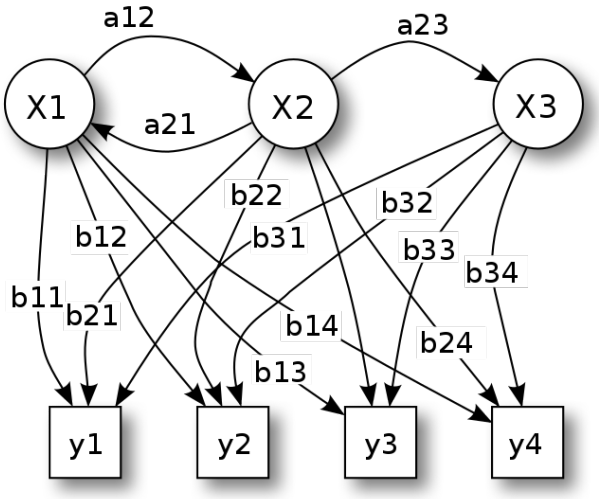

Figura 4.2: Model Markov cu stări ascunse, fiecare simbol reprezentând următoarele: X - stările, Y - posibilele observații, a - probabilitățile de tranziție între stări, b - probabilitățile de emisie [20].

Pentru recunoaștere automată a acordurilor muzicale, se aplică problema de învățare a unui model Markov cu stări ascunse, utilizând pentru antrenare vectorii care determină reprezentarea prin chromagrama. Modelul conține o singură stare pentru fiecare acord distins de către sistem.

Astfel, cunoscându-se vectorii caracteristici observați(X), etichetele pentru acorduri(Q) și parametrii modelului curent(θ), se poate calcula utilizând algoritmul EM valoarea probabilității $P(X,Q|\theta)$ pe baza unei expresii definite ulterior. Algoritmul garantează că estimările se vor îmbunătății de la o etapă la alta, ajungând într-un optim local, în ceea ce privește setul de parametrii. Astfel, soluția estimează în mod rezonabil un set de parametrii care maximizează probabilitea definită.

O astfel de abordare este descrisă în una din primele lucrări științifice ale domeniului, despre care se va discuta în Capitolul 6, secțiunea Cercetări conexe.

4.3 Rețele neuronale artificiale

"Birds inspired us to fly, burdock plants inspired velcro, and countless more inventions were inspired by nature. It seems only logical, then, to look at the brain's architecture for inspiration on how to build an intelligent machine [12]."

Creierul uman este cea mai complexă structură a corpului uman, fiind capabil de procesare paralelă a informației, având o capacitate de stocare a ei de 85-100 de miliarde de neuroni, fiecare având aproximativ 10000 de conexiuni.

Primul element care este astfel preluat de rețelele neuronale este neuronul. Neuronul este cea mai mică unitate fundamentală a sistemului nervos central, având rolul de a primi, conduce, procesa și transmite mai departe diferite semnale electrice primite de la organele de simț sau de la alți neuroni. Neuronul biologic este compus din:

- Corp celular (Soma);
- Axon - având rolul de a transporta semnale către alți neuroni, sau celule țintă. Acesta are la capăt terminații sinaptice, prin care se poate lega cu dendritele altor neuroni sau direct cu corpul lor;
- Dendrite - prin intermediul cărora se primesc semnale de la axonii altor neuroni;
- Sinapse - reprezintă conexiuni care se realizează între axonul unui neuron și dendritele altui neuron.

Preluând aceste informații biologice, se poate realiza o mapare a fiecărui element, în vederea obținerii unei reprezentări pentru o rețea neuronală artificială.

RN Biologică	RN Artificială
Soma	Nod
Dendrite	Intrare
Axon	Ieșire
Activare	Procesare
Sinapsă	Conexiune ponderată

Tabelul 4.1: Componente: rețea neuronală biologică vs. rețea neuronală artificială

Astfel, rețelele neuronale artificiale(ANN) se definesc ca structuri artificiale care încearcă să reproducă modul de funcţionare a creierului uman. Sunt construite din mai multe unități de procesare sau *neuroni artificiali* grupaţi în straturi, fiecare strat având un număr variabil de elemente. Fiecare neuron poate primi informații de la alți neuroni, fiind acceptată chiar primirea de informații de la el însuși.

Un neuron artificial modelează comportamentul unui neuron real. Astfel conexiunile dintre neuroni, numite ponderi sinaptice, sunt folosite în stocarea informației. După o procesare locală a semnalului de intrare pe baza informaţiei stocată în ponderile sinaptice (multiplicarea acesteia cu valorile informaţionale stocate) se produce o integrare (sumare) globală a rezultatelor obţinute (proces similar cu cel ce are loc în corpul celular al unui neuron biologic real). Dacă răspunsul obţinut depăşeşte un anumit prag, informaţia este transmisă mai departe (utilizarea unei funcții de activare) [8].

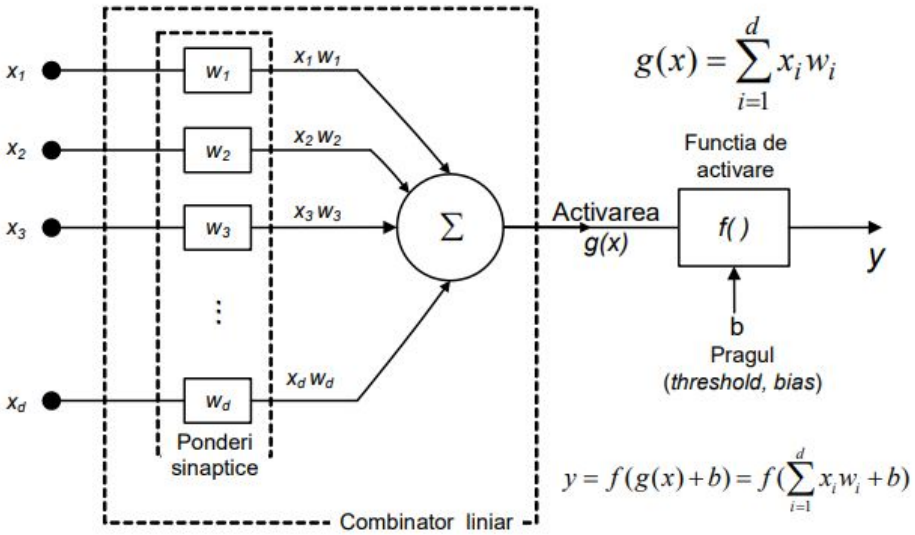

Figura 4.3: Structura generală a unui neuron artificial [8].

Valorile stocate prin intermediul ponderilor își schimbă valorile folosind algoritmilor de învățare. Astfel, în cadrul unui ANN, în general, cel care construiește rețeaua nu trebuie să specifice valori pentru parametrii sistemului (ponderile fiecăruil neuron în parte). Valorile pentru acești parametrii vor fi extrase, în mod automat, prin intermediul algoritmilor de antrenare sau adaptare. Fiecare algoritm se clasifică în unul din obiectivele de învățare enunțare mai sus: supervizată, nesupervizată, semisupervizată sau prin întărire.

Pașii care trebuie urmați pentru construirea unei rețele neuronale artificale sunt:

- Construirea stratului de intrare cu m noduri;
- Constuirea stratului de ieșire cu n noduri;
- Determinarea numărului de staturi ascunse și construirea lor, cu unul sau mai mulți neuroni pe fiecare strat;
- Inițializarea ponderilor între nodurile aflate pe straturi diferite;
- Stabilirea funcției de activare corepunzătoare fiecărui neuron (de pe straturile ascunse).

Prin specificarea tutoror elementelor enunțate mai sus, se generează *topologia* rețelei. Din punct de vedere al topologiei, ANN-urile se clasifică în două tipuri:

- Feed-Forward: Informația se propagă de la un strat la altul, doar de la intrare spre ieșire, iar ieșirea fiecărui nod depinde intrările sale care sunt conectate cu ieșirile neuronilor de pe stratul precendent. Sunt folosite în special pentru învățarea supervizată.
- Recurente: Prin intermediul legăturilor existente, se creează bucle ce determină ca ieșirea diferiților neuroni să fie dependentă și de valorile anterior calculate (spre exemplu, ieșirile altor neuroni de pe același strat). Au aplicabilitate atât pentru învățarea supervizată, cât și pentru cea nesupervizată.

Odată ce rețeaua a fost proiectată, procesul de antrenare (învățare) poate începe. Acest proces urmărește obținea unor valori optime pentru ponderile aflate între oricare două noduri ale rețelei, prin *minimizarea erorii*. Eroarea se calculează determinând diferența dintre rezultatul real și rezultatul calculat de rețea.

În general, rețelele neuronale artificiale sunt formate dintr-un număr mic de straturi (3-5), cele mai frecvente fiind rețelele cu 3 straturi, adică cu un singur strat ascuns. Pentru construirea unor rețele mai complexe, nu se utiliează acest tip de rețele, fiind preferate rețelele neuronale profunde (*deep neural networks*). Această categorie de rețele utilizează un număr superior de straturi, cu scopul de a dobândi capacitatea de a învăța reprezentări ale datelor de intrare cu nivele multiple de abstractizare (de exemplu, pentru clasificarea a milioane de imagini, recunoașterea audio, construirea unor sisteme de recomandare audio/video sau înțelegerea limbajului natural).

4.4 Rețele neuronale profunde

Nimic din natură sau din evoluția tehnologiei, până în ziua de azi, nu se compară cu abilitățile complexe de procesare a informației și de recunoaștere a diferitelor modele complexe pe care le are creierul uman. Încercarea tehnologiei este a avansa în această direcție, dezvoltând algoritmi care imită rețeaua creierului uman, acestea fiind numite rețele neuronale profunde (*deep neural networks*).

Rețelele neuronale profunde au o structură unică, deoarece au o componentă ascunsă (formată din straturi ascunse) relativ mare și complexă între straturile de intrare și ieșire. Pentru a fi considerată o rețea neuronală profundă, această componentă ascunsă trebuie să conțină cel puțin două straturi. Datorită structurii lor, rețelele neuronale profunde au o capacitate mai mare de a recunoaște tipare decât rețelele superficiale.

Există câteva arhitecturi neuronale profunde consacrate, cum ar fi:

- Rețele neuronale convoluționale (CNN)
- Rețele neuronale recurente (RNN)
- Rețele de convingeri profunde *(Deep belief networks - DBN)*

4.4.1 Rețele neuronale convoluționale

Rețelele neuronale convoluționale(CNN) sunt foarte similare cu rețelele neuronale obișnuite. Diferența constă în faptul că rețeaua face o presupunerea explicită că valorile de input sunt imagini, ceea ce îi permite să codifice anumite propietăți în cadrul arhitecturii. De asemenea, spre deosebire de o rețea obișnuită, straturile unui CNN au neuronii aranjați în 3 dimensiuni: lățime, înălțime și adâncime(de precizat, adâncimea se referă la a treia dimensiune de activare, nu la adâncimea rețelei neuronale complete, valoare care este egală cu numărul total de straturi dintr-o rețea) [7].

În general, rețelele neuronale convoluționale, cunoscute și ca ConvNets, utilizează 3 tipuri de straturi pentru a construi arhitectura, și anume: convolutional layer, pooling layer și fully-connected layer.

Un strat de convoluție *(Convolutional Layer)*, este responsabil de scanarea unei reprezentări sursă de tip imagine, aplicând un filtru de o anumită dimensiune, cu scopul de a extrage caracteristici care pot fi importante pentru clasificare. Acest filtru mai este numit nucleu de convoluție *(convolution kernel)*. Nucleul conține parametrii care pot fi ajustați pentru a atinge cele mai precise predicții.

Spre exemplu, în cadrul unui nucleu de dimensiunea 5 x 5, pentru fiecare regiune de 5 x 5 pixeli, modelul calculează o serie de produse între valorea pixelului din imagine, la o anumită poziție și parametrul corespunzător definit în filtru.

După încheierea convoluției, caracteristicile sunt comprimate *(downsampled)*, urmând ca aceeași structură convoluțională să se repete. La început, convoluția identifică trăsături din imaginea originală, apoi identifică sub-caracteristici în părți mai mici ale imaginii. În cele din urmă, acest proces este menit să identifice caracteristicile esențiale care pot ajuta la clasificarea imaginii. Straturile de convoluție produc astfel unul sau mai multe imagini numite hărți de trăsături *(feature maps)*, imagini care conțini caracteristici ce apațin imaginii originale, înainte de punerea în aplicare a nucleului de convoluție.

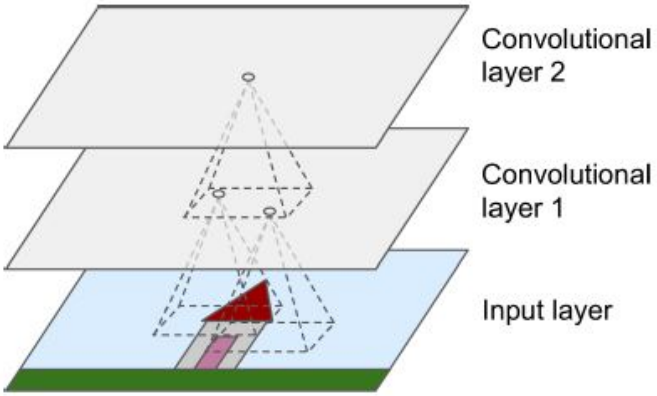

Figura 4.4: Două straturi de convoluție în cadrul unui CNN, aplicând câte un filtru pe o regiune din imaginea de input [12].

Stratul de agregare, *Pooling layer*, are rolul de a reduce/micșora dimensiunea imaginii de intrare, comprimând ieșirile unui strat convoluțional. Se consideră, ca exemplu, un filtru de dimensiunea 2 x 2 care urmează a fi aplicat folosind agregarea asupra unei imagini de dimensiunea 4 x 4. Folosind acest filtru, există două strategii care pot fi aplicate.

1. Calculul mediei valorilor din regiunea acoperită de filtru *(mean pooling)*.

2. Determinarea maximului din regiunea acoperită de filtru *(max pooling)*.

Straturile de convoluție și agregare sunt supuse unei funcții de activare cu rolul de a asigura comportamentul neliniar al rețelei. Ca funcție de activare, de cele mai multe ori se folosește ReLU *(Rectified Linear Unit)*. Mai este cunoscută ca operația de rectificare. ReLu este funcția de activare cel mai frecvent folosită în construirea modelelor de învățare profundă. Funcția returnează 0 dacă primește o intrare negativă, iar pentru orice valoare pozitivă x, se returnează aceea valoare. Pe scurt, se definește ca:

$$f_{ReLu}(x) = max(0,x)$$

O componentă complet conectată , *Fully connected*, are rolul de a realiza clasificarea propriu zisă. Această componentă este o rețea neuronală clasică, la intrarea căreia se furnizează feature map-urile, și la ieșirea căreia se aplică funcția de activare *softmax*.

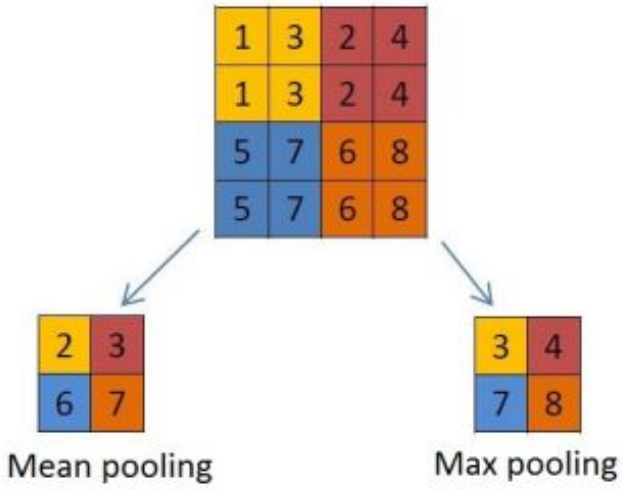

Figura 4.5: Exemplificarea operațiilor de mean pooling (stânga) și max pooling (dreapta). Fiecare element din straturile de tip pool este obținut din matricea de intrare, din zona cu aceeași culoare. În acest exemplu, din matricea de intrare de dimensiuni 4×4 se obține rezultatul cu dimensiunea 2×2 [21].

În matematică, funcția softmax, cunoscută și ca softargmax sau funcție exponențială normalizată, este o funcție care ia ca input un vector K de numere reale și îl normalizează într-o distribuție de probabilități, constând în probabilități K proporționale la valorile exponențiale ale numerelor din vectorul de intrare. Spre exemplu, înainte de aplicarea funcției softmax, unele componente vectoriale pot avea valori negative, sau mai mari decât 1, fiind posibil ca suma tuturor componentelor să nu fie egală cu 1. Dar, aplicând softmax, fiecare componentă se va afla în intervalul $(0,1)$, ceea ce înseamnă că vor putea fi interpretate ca probabilități. Funcția softmax se definește matematic astfel:

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}, i = 1, ..., K; z = (z_1, ..., z_k)$$

Astfel, ieșirea acestei componente este un vector de probabilități cu un număr de componente egal cu numărul de clase. Fiecare componentă a vectorului reprezintă probabilitatea ca imaginea dată ca input să se încadreze în clasa corespunzătoare.

Recapitulând întreg procesul, imaginea de la intrare conține o entitate care trebuie încadrată într-una din clasele preexistente. Input-ul este supus mai multor operații de convoluție, agregare și rectificare, de fiecare dată generându-se hărți de trăsături. Acestea conțin trăsături semnificative ale imaginii inițiale. Trăsăturile sunt apoi supuse unui proces de clasificare prin intermediul â unei rețele neuronale complet conectate, rezultând o serie de probabilități ca imaginea să aparțină fiecărei categorii.

Arhitecturile tipice de CNN conțin câteva straturi convoluționale (fiecare având în continuarea lui câte un strat ReLu), apoi un strat de agregare, apoi alte câteva straturi convoluționale,

apoi un alt strat de agregare, și așa mai departe. Imaginea devine din ce în ce mai mică pe măsură ce progresează prin rețea, devenind de asemenea din ce în ce mai profundă(adică, cu mai multe hărți de trăsături), datorită straturilor convoluționale.

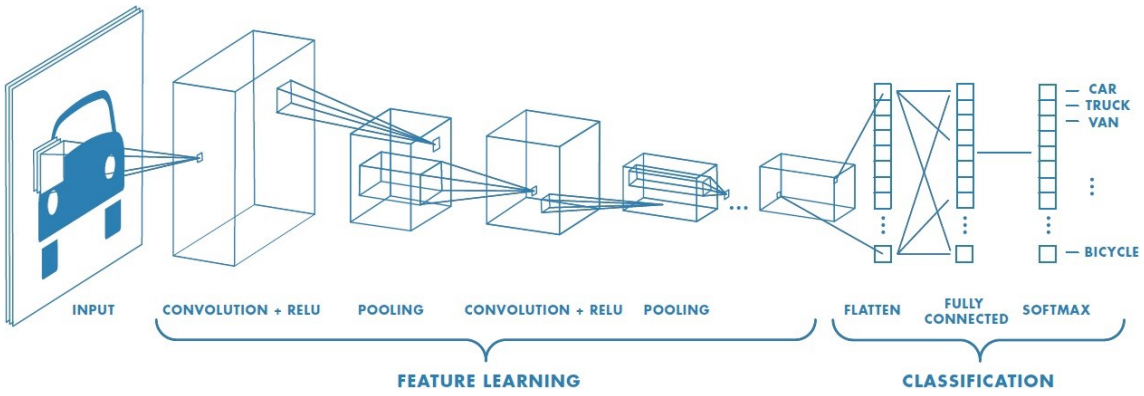

Figura 4.6: Arhitectură CNN. Se consideră o imagine ca input, care este supusă operațiilor corespunzătoare straturilor de convoluție și agregare. Rezultatul se propagă prin straturi complet conectate. Ieșirea rețelei constă într-un vector de probabilități ce încadrează imaginea într-o mulțime de clase preexistente [17].

Există o multitudine de arhitecturi actuale folosite în mod uzual, unele fiind specializate pe rezolvarea anumitor probleme. Cu toate acestea, unele sunt considerate de mare interes, fiind situate la baza acestui domeniu, printre care:

- LetNet-5 - Este cea mai veche arhitectură convoluțională, fiind descrisă pentru prima oară în 1998 cu scopul de a recunoaște cifre scrise de mână în cadrul documentelor, fiind concepută pentru imagini cu rezoluție mică;
- AlexNet - Este o arhitectură apărută după mai bine de un deceniu, în anul 2012, având 5 straturi convoluționale, acceptând imagini de intrare considerabil mai mari în rezoluție, având nuclee de convoluție mai mari în straturile inițiale (11 x 11). A fost antrenată cu ajutorul a două GPU-uri performante, fiind mult mai puternică decât arhitecturile precendete;
- VGG - Este arhitectura care a demonstrat creșterea performanței odată cu adâncimea mai mare a rețelei, având un număr semnificativ de straturi de orice tip(16, 19), fiind printre cele mai mari rețele utilizate în ceea ce privește numărul de parametri care trebuie învățați.

4.4.2 Rețele de convingeri profunde

Rețelele de convingeri profunde se prezintă ca o categorie de algoritmi care folosesc probabilitățile și învățarea nesupervizată pentru a produce rezultate. Sunt compuse din variabile latente(variabile care nu sunt observate direct, ci sunt mai degrabă deduse din alte variabile observate) binare, și conțin atât straturi nedirecționate, cât și straturi direcționate.

Spre deosebire de alte modele, fiecare strat din cadrul unei rețete de convingeri profunde învață caracteristici din întreaga entitate de input. Prin comparație, CNN-urile utiliează primele straturi pentru a filtra entitatea de intrare, obținând doar caracteristicile de bază, straturile ulterioare recombinând informația din straturile inferioare. În cadrul rețelei, nodurile din fiecare strat sunt conectate la toate nodurile din anterior si ulterior. Nodurile nu comunică lateral în cadrul aceluiași strat. De asemenea, nodurile din stratul ascuns îndeplinesc două roluri: acționează ca un strat ascuns față de nodurile care îl preced și ca straturi vizibile pentru nodurile care îl succed. Aceste noduri identifică corelații între date [24].

Pentru antrenarea rețelelor de convingeri profunde, se folosesc algoritmi de tip greedy. Această abordare este specifică algoritmilor care implică luare unei decizii la fiecare strat din secvență, găsind în cele din urmă un optim global. Învățarea se desfășoară de la un nivel la altul, ceea ce înseamnă că straturile rețelelor de convingeri profunde sunt instruite pe rând. Prin urmare, fiecare strat primește o versiune diferită a datelor și fiecare folosește ieșirea din stratul anterior ca intrare a acestora. Algoritmii de tip greedy sunt folosiți pentru a antrena rețele de convingeri profunde deoarece sunt rapizi și eficienți. Mai mult, ajută la optimizarea parametriilor la fiecare strat.

În ceea ce privește ariile de aplicabilitate, acest tip de rețele pot fi utilizate cu succes în recunoașterea audio sau de imagini, dar și implementarea unor algoritmi de colectare a datelor în timp real (de exemplu, urmărirea mișcării de obiecte sau de persoane). Recunoașterea de note sau acorduri dintr-o reprezentare audio are la bază clasificarea de imagini, în urma procesării sunetului și obținerea unei reprezentări pe baza trăsăturilor extrase.

Rezultatele unei astfel de abordări vor fi prezentate în Capitolul 6, secțiunea Cercetări conexe, pe baza unei lucrări științifice care abordează diferite strategii, printre care și cea a rețelelor de convingeri profunde.

Capitolul 5

Rezultate experimentale

5.1 Setul de date

Setul de date utilizat pentru antrenarea și evaluarea modelului convoluțional creat este format din combinarea a 3 seturi de date pentru chitară, disponibile online, asupra cărora s-au aplicat o serie de algoritmi de augmentare. Setul de date este astfel compus din:

- 7398 de acorduri individuale, grupate în 16 fișiere audio de tip wav, aparținând Institutului de semantică muzicală, Fraunhofer, din cadrul Universității tehnice Ilmenau, Germania. Setul de date se numește IDMT-SMT-CHORDS;
- 6580 de înregistrări audio, fiecare având în compoziție un singur acord, aparținând laboratorului de cercetări muzicale al Universității din New York, SUA, numit GuitarSet[3];
- 200 de fișiere audio pentru 10 tipuri de acorduri, fiind colectat de grupul de cercetare Motefiore al Universității din Liège, Belgia.

După aplicarea algoritmilor de augmentare asupra setului de date complet, s-a obținut un total de 58.577 de înregistrări audio de tip wav, grupate în cele 24+1 clase(24 de acorduri cu etichete corespunzătoare cunoscute, și o etichetă pentru orice alt acord care nu se încadrează).

Cele 24 de acorduri recunoscute de către sistem, împreună cu valorile numerice asociate sunt următoarele(perechi de forma acord muzical-valoare asociată): A-0, A#-1, A#m-2, Am-3, B-4, Bm-5, C-6, C#-7, C#m-8, Cm-9, D-10, D#-11, D#m-12, Dm-13, E-14, Em-15, F-16, F#-17, F#m-18, Fm-19, G-20, G#-21, G#m-22, Gm-23. Pentru orice alt acord care nu se încadrează în această înșiruire s-a creat perechea N-24 (acord necunoscut N, cu valoarea asociată de 24).

5.2 Augmentarea datelor

Metodele de augmentare a datelor au rolul de a crește în mod artificial dimensiunea setului de date, generând multe variante realiste ale fiecărei instanțe set. Această procedură reduce posibilitatea de overfitting.

Instanțele generate trebuie să fie cât mai realiste, astfel încât nimeni să nu fie capabil să diferențieze între varianta originală și cea augmentată(caz ideal, în realitate vor exista aspecte care le vor diferenția) [12]. Asupra acestui set de date s-au aplicat două metode de augmentare diferite, anume:

1. Modicarea vitezei de redare, pentru fiecare instanță în parte, prin două metode:
    - Incrementarea vitezei cu 1.25% față de original;
    - Decrementarea vitezei cu 0.75% față de original.
2. Injectarea unui zgomot de fundal, pentru fiecare fișier audio, prin adăugarea unei valori generate random în vectorul de date specific.

5.3 Parametrii experimentali

5.4 Rezultate

Urmeaza a fi procesate si descrise mai in detaliu

La antrenare
Train loss: 0.07867297860684848
Train accuracy: 0.982008695602417
Train precision: 0.990569531917572
Train recall: 0.970093846321106
Train f1-score: 0.9799584746360779

La testare
Test loss: 0.5396939868179629
Test accuracy: 0.8533378839492798
Test precision: 0.9028927087783813
Test recall: 0.8160762786865234
Test f1-score: 0.8566158413887024

Capitolul 6

Studiu de caz

6.1 Cercetări conexe

Ultimii 20 de ani au adus numeroase abordări în ceea ce privește recunoașterea automată a partiturilor muzicale. De la recunoașterea individuală a instrumentelor muzicale, până la recunoașterea completă a notelor și diferențierea instrumentelor dintr-o piesă completă, s-au încercat diferite metode, în funcție de cantitatea de date disponibilă până în acel moment, dar și de evoluția tehnologiei.

Una din primele lucrări care a stat la baza abordărilor ulterioare pentru o perioadă de timp este lucrarea propusă de către Sheh și Ellis [9]. Metoda are la bază învățare statistică utilizând modele Markov cu stări ascunse. Antrenarea modelului s-a realizat folosind algoritmul EM, tratând etichetele pentru acorduri ca valori ascunde pentru construcția EM. Tot pentru antrenarea modelului au folosit doar secvența de acorduri (*without chord boudaries* - neclasificate, în raport cu intervalele de timp în care se găsește un anumit acord) ca input pentru modelul Markov, aplicând astfel algoritmul Baum-Welch pentru a estima parametrii modelului. Algoritmul garanta că estimările se vor îmbunătății de la o etapă la alta, ajungând într-un optim local, în ceea ce privește setul de parametrii. După determinarea parametrilor, asupra modelului se aplica algoritmul lui Viterbi, pentru a afla cea mai optimă cale(cea mai probabilă secvență de acorduri pentru un semnal audio de intrare).

Rezultatele obținute au fost: 76% pentru precizia segmentării acordurilor din piese și 22% pentru precizia în recunoașterea acordurilor. Performanța scăzută pentru recunoaștere se datorează faptului că datele de antrenare au fost neclasificate, dar și faptului că acestea au fost insuficiente (20 de melodii pentru 147 de acorduri).

Plecând de la această abordare, aproximativ 3 ani mai târziu, Lee și Slaney au enunțat o metodă îmbunătățită [22], având un set de date adnotat și mai numeros, compus din 140 de melodii, obținând o precizie de 93.35% în recunoașterea acordurilor, din cadrul melodiilor testate.

Începând cu anul 2009, odată cu apariția unui set de date compus din piese de pe diverse

albumuri apațînând trupelor Beatles, Zweieck sau Queen, complet adnotate cu acorduri, strategia în rezolvarea acestei probleme s-a schimbat, și foarte multe lucrări au utilizat acest set de date, aplicând diverse metode și comparând rezultate.

O primă lucrare care a abordat mai multe strategii de clasificare, dar și de procesare sonoră, a fost lucrarea de masterat a lui Alessandro Bonvini, din anul universitar 2012-2013, [4]. În ceea ce privește algoritmii de învățare automată, a abordat problema utilizând 3 metode, și anume: regresie logistică (LgR), construirea unui ANN, având un singur strat ascuns *(Single Hidden Multilayer Perceptron (MPL))* și construirea unei rețele de convingeri profunde (DBN).

Bonvini a prezentat rezultatele comparativ, aplicând mai multe strategii de normalizare a datelor, testând la fiecare schimbare toți algoritmii. Se vor prezenta rezultatele obținute prin aplicarea strategiei de normalizare $L-\infty$. În ceea ce privește metricile de evaluare a performanței, a introdus două valori specifice numite Weighted Average Overlap Ratio(WAOR), indicând durata de timp în care segmentele clasificate în mod corect rămân suprapuse celor din realitate, și Segmentation Quality(SQ), sugerând calitatea segmentării acordurilor.

Rezultatele obținute au fost:

- LgR - WAOR: 66.9 %, SQ: 74.1 %;
- MPL - WAOR: 69.6 %, SQ: 74.9 %;
- DBN - WAOR: 71.9 %, SQ: 76.0 %.

Tot în preajma anului 2012, în cadrul conferinței internaționale de machine learning din Florida, SUA, s-a prezentat o lucrare care avea să schimbe încă odată direcția de a aborda problema recunoașterii automate a acordurilor, prezentând rețelele neuronale convoluționale ca o soluție cu eficiență ridicată, prin comparație cu abordările anterioare, în special prin referire la modelele Markov.

Lucrarea propusă de către Humphrey și Bello, [2], descrie două arhitecturi convoluționale, prezentând construcția strat cu strat și strategia antrenării. O precizare importantă este aceea că, în privința procesării audio și a extragerii de trăsături nu a existat o schimbare, metoda aleasă fiind transformarea Q constantă. Setul de date utilizat conține 475 de înregistrări audio, însumând aproximativ 50000 de acorduri diferite.

Antrenând și testând cele două rețele convoluționale, s-a obținut 77.4 %, respectiv 76.8 % acuratețe la testare, demonstrând că această abordare performează într-un mod competitiv cu alte abordări considerate consacrate până în acel moment.

Începând din acel moment, majoritatea lucrărilor au propus metode pe baza rețelelor neuronale convoluționale. Cea mai concludentă lucrare analizată, prin raportare la soluția propusă, este cea prezentată de Zhou și Lerch, [14], de la centrul pentru tehnologia muzicii, din cadrul institutului de tehnologie din Georgia, SUA. Lucrarea prezintă construcția a două rețele profunde cu 6 straturi, în două arhitecturii diferite: o arhitectură clasică, cu același număr de neuroni (1024) în fiecare strat, și o arhitectură de tip *bottleneck (cu număr scăzut de neuroni în mijlocul*

rețelei, și în vecinătatea mijlocului). Setul de date constă în 317 piese adunate din discografiile unor trupe celebre, fiecare piesă fiind divizată în peste 1000 de cadre, pentru recunoașterea individuală a acodurilor. Algoritmul propus este capabil să recunoască acordurile majore și minore pentru fiecare notă de tip rădăcină, rezultând un dicționar de etichete pentru acorduri de dimensiunea 24+1, având 24 de acorduri cu etichete corespunzătoare cunoscute, și o etichetă pentru orice alt acord care nu este recunoscut.

Rezultatele obținute sunt prezentate pe ambele arhitecturi, aplicând diferite metode de preprocesare a datelor. Cele mai bune rezultate au fost opținute folosind strategia *spliced filters*. Metrica de evaluare a perfomanței este durata totală a segmentelor cu predicție corectă. Astfel, rezultatele sunt:

- Arhitectura comună - 98.5% la antrenare și 87.6% la testare;
- Arhitectura bottleneck - 93.6% la antrenare și 91.9% la testare.

Se observă o îmbunătățire semnificativă a rezultatelor, construindu-se rețele convoluționale tot mai eficiente, creșterea fiind direct proporțională cu creșterea datelor etichetate corect și în detaliu.

Evoluția studiului în domeniu continuă până în preajma perioadei actuale, fiind prezentată o lucrare foarte importantă acum aproximativ un an de către Nadar, Abeßer și Grollmisch, [13], în cadrul unei conferințe de procesare sonoră, de la Malaga. Lucrarea urmărește extinderea dicționarului de etichete de la 24 la 84 de acorduri, extinzând aria claselor de acorduri care pot fi recunoscute. Dacă până acum se recunoșteau doar acordurile majore și minore ale notelor rădăcină, în această lucrare se dorește recunoașterea și a altor clase, printre care septacorduri sau acorduri diminuate. Setul de date este complex, fiind compus atât din clasicele piese din cadrul discografiilor unor trupe celebre, cât și dintr-un set de date special pentru chitară, creat de Institutul de semantică muzicală, Fraunhofer, din cadrul universității tehnice Ilmenau, Germania.

Modelul convoluțional prezentat este alcătuit din mai multe straturi convoluționale, intercalate de straturi de agregare de tip max pooling, rețeaua având un total de 12 straturi. Rețeaua a fost construită pentru a aborda două stragetii, una care propune recunoașterea pe baza dicționarului extins, dar care nu folosește întreg setul de date (S-84), și una care folosește dicționarul clasic, cu 24 de acorduri (S-24).

Metrica folosită este *f-score*. Astfel, cele mai bune rezultate opținute sunt:

- S-84 - 76%, folosind doar setul de date al institutului Fraunhofer;
- S-24 - 91%, folosind întreg setul de date.

6.2 Arhitectura sistemului

Diagrama cazurilor de utilizare, Modularizarea sistemului

6.3 Dezvoltarea sistemului

6.3.1 Descrierea soluție propuse

Pentru construcția modelului de recunoaștere a acordurilor acustice, se va utiliza o rețea neuronală convoluțională. CNN-urile au crescut în popularitate în ultimii ani, mai ales în domeniul viziunii computerizate (*computer vision*).

În cadrul rețelelor neuronale convoluționale, imaginea construită sau aleasă este folosită ca parametru de intrare pentru CNN, iar imaginea este trecută prin prin mai multe straturi, cum ar fi straturile convoluționale, straturile de agregare și starturile de activare. Straturile convoluționale sunt construite din mai multe filtre, care pot fi interpretate individual, fiecare învățând unele caracteristici superioare ale imaginii.

În cadrul problemei de recunoaștere automată a acordurilor muzicale acustice, detectarea acordurilor poate fi tratată în mod similar cu recunoașterea imaginilor, deoarece se pot crea imagini cu reprezentarea de tip chromagramă. Identificarea notelor sau a acordurile este mai simplă decât clasificarea imaginilor, întrucât nu implică învățarea anumitor texturi, rotiri sau scalări. Cu toate acestea, această problemă vine cu alte provocări. Notele muzicale din cadrul chromagramei nu sunt localizate într-o singură regiune în același mod în care sunt cele mai multe obiecte din imagini - o notă la o anumită frecvență fundamentală va fi compusă din armonice la multiplii acelei frecvențe.

În ciuda acestei provocări, CNN-urile au proprietăți avantajoase care pot fi aplicate cu succes în acestă problemă. Experimentele anterioare au sugerat că agregarea informațiilor considerând mai multe cadre muzicale din același sample, determină obținerea unei predicții cu o performanță mai mare. Astfel, convoluțiile determinate asupra datelor de intrare permit modelului creat să învețe caracteristici muzicale polifonice valoroase.

6.3.2 Mediu de lucru

Limbaj: Python; Env: Google Colab, Visual Studio Code, PyCharm

6.3.3 Extragerea trăsăturilor muzicale

6.3.4 Construirea și antrenarea modelului neuronal

6.3.5 Detectarea tranziției muzicale

6.3.6 Pachete și librării externe

LibROSA - librărie Python pentru analiză audio. Furnizează pachetele necesare construirii unui sistem de obținere a informațiilor muzicale *(music information retrieval systems)* [15].

Capitolul 7

Aplicație mobile

7.1 Mediu de lucru

Limbaj: Kotlin; Env: Android Studio

7.2 Arhitectura aplicației

7.3 Design

7.4 Funcționalități

Capitolul 8

Concluzii

8.1 Rezumat

8.2 Îmbunătățiri viitoare

Bibliografie

[1] *Analog to digital converion process*. https://victorimpmooc.wordpress.com. Accesat în 2020-03-05. July 2014.

[2] Eric J. Humphrey; Juan P. Bello. "Rethinking Automatic Chord Recognition with Convolutional Neural Networks". In: *Music and Audio Research Lab (MARL) New York University* (2012 11th International Conference on Machine Learning and Applications).

[3] Q.Xi; R.Bittner; J.Pauwels; X.Ye; J. P. Bello. "Guitarset, A Dataset for Guitar Transcription". In: *19th International Society for Music Information Retrieval Conference, Paris, France* (2018).

[4] Alessandro Bonvini. "Automatic chord recognition using Deep Learning techniques". In: *Journal: POLITECNICO DI MILANO Facoltà di Ingegneria dell'Informazione Corso di Laurea Magistrale in Ingegneria e Design del suono Dipartimento di Elettronica e Informazione* (2012-2013).

[5] R. N. Bracewell. *The Fourier Transform and Its Applications (ed. 3rd)*. McGraw-Hill, ISBN-13: 9781124130941, 2000.

[6] Judith C Brown. *Calculation of a constant q spectral transform*. The Journal of the Acoustical Society of America, 1991.

[7] *CS231n: Convolutional Neural Networks for Visual Recognition, by Stanford University*. https://cs231n.github.io/convolutional-networks/overview. Accesat în 2020-04-06.

[8] Dan-Marius Dobrea. *Tehnici de inteligență computațională. Aplicații în electronică și biomedicină*. Editura Performantica (editură acreditată CNCSIS), Iaşi, România, ISBN 978-606-685-546-4, 2017.

[9] Alexander Sheh; Daniel P.W. Ellis. "Chord Segmentation and Recognition using EM-Trained Hidden Markov Models". In: *International Symposium on Music Information Retrieval* (2003).

[10] *Fourier Transform Visualization, based on video presentation. Experiment using project*. https://prajwalsouza.github.io/Experiments/Fourier-Transform-Visualization.html.

[11] Munteanu Gabriela. *Educație muzicală, Sinteze și corelări interdisciplinare*. Editura Artrom, 2001.

[12] Aurélien Géron. *Hands-on Machine Learning with Scikit-Learn, Keras and TensorFlow*. O'Reilly Media, ISBN-13: 9781492032649, 2019.

[13] Christon-Ragavan Nadar; Jakob Abeßer; Sascha Grollmisch. "Towards CNN-based Acoustic Modeling of Seventh Chords for Automatic Chord Recognition". In: *Sound and Music Computing Conference (SMC), Malaga* (2019).

[14] Xinquan Zhou; Alexander Lerch. "Chord detection using Deep Learning". In: *ISMIR 2015, International Conference on Music Information Retrieval* (2015).

[15] *LibROSA*. https://librosa.github.io/librosa/index.html. Accesat în 2020-04-06.

[16] Daniel Jurafsky; James H. Martin. *Speech and Language Processing, An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Draft of October 16, ISBN-13: 9789332518414, 2019.

[17] *MathWorks-Convolutional Neural Network*. http://www.mathworks.com/solutions/deep-learning/convolutional-neural-network.html. Accesat în 2020-04-08.

[18] Juan P Bello; Jeremy Pickens. *A robust mid-level representation for harmonic content in music signals.* In prococeedings of ISMIR, pag. 304-311, 2005.

[19] Preeti Rao. "Audio Signal Processing, Chapter in Speech, Audio, Image and Biomedical Signal Processing using Neural Networks". In: *Bhanu Prasad and S. R. Mahadeva Prasanna, Springer-Verlag* (2007).

[20] *ResearchGate - Example of the probabilistic parameter of a hidden Markov model*. https://www.researchgate.net/. Accesat în 2020-04-05.

[21] *Robotics, Vision and Control (ROVIS) Laboratory*. http://www.rovislab.com/courses/ml/. Accesat în 2020-04-08.

[22] Kyogu Lee; Malcolm Slaney. "Automatic Chord Recognition from Audio Using an HMM with Supervised Learning". In: *ISMIR 2006, 7th International Conference on Music Information Retrieval* (2006).

[23] *The Music Studio*. https://www.themusicstudio.ca/blog/2017/11/909/. Accesat în 2020-03-29.

[24] *Working with CNN 2D Convolutions in Keras*. https://missinglink.ai/guides/keras/keras-conv2d-working-cnn-2d-convolutions-keras/. Accesat în 2020-04-06.