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

Metode de detecţie a semnalelor audio specifice destinate implementărilor de tip *embedded*, cu aplicaţii în prevenirea exploatărilor forestiere ilegale

Lucrare de disertaţie

prezentată ca cerinţă parţială pentru obţinerea titlului de *Master* în domeniul *Telecomunicaţii*
programul de studii de masterat *Tehnologii multimedia în aplicaţii de biometrie şi securitatea informaţiei*

Conducător ştiinţific:
Prof. dr. Inginer Burileanu Corneliu

Absolvent:
Gaiţă Andrei

Anul 2018

Universitatea "Politehnica" din București
Facultatea de Electronică, Telecomunicații și Tehnologia Informației
Programul de masterat **Tehnologii multimedia în aplicații de biometrie și securitatea informației**

Anexa 2

TEMA LUCRĂRII DE DISERTAȚIE
a masterandului **GAIȚĂ D. Andrei , 421-BIOSINF**

1. Titlul temei: Metode de detecție a semnalelor audio specifice destinate implementărilor de tip embedded, cu aplicații în prevenirea exploatărilor forestiere ilegale

2. Descrierea contribuției originale a masterandului (în afara părții de documentare):
- Evidențierea comparativă a rezultatelor metodelor reprezentând starea artei în domeniul temei de cercetare.
- Proiectarea și construirea unei baze de semnale audio specifice.
- Testarea unor metode de detecție a sunetului produs de motofierăstrău atât cu baza de date proprie cât și cu o altă bază de date de referință disponibilă.
- Implementarea metodei care oferă cele mai bune rezultate pentru scopul propus, pe o platformă embedded.
- Se vor trage concluzii asupra rezultatelor practice obținute și a contribuției originale și se vor enunța perspective ulterioare de cercetare.

3. Resurse folosite la dezvoltarea proiectului:
Pentru testarea metodelor și analiza rezultatelor se va folosi mediul de simulare MATLAB, iar platforma embedded va fi reprezentată de un sistem de calcul Raspberry Pi.

4. Proiectul se bazează pe cunoștințe dobîndite în principal la următoarele 3-4 discipline:
Interfațare vizuală om-mașină, Tehnici de analiză și clasificare automată a informației

5. Proprietatea intelectuală asupra proiectului aparține: masterandului

6. Data înregistrării temei: 2017-12-03 23:18:13

Conducător(i) lucrare,
Prof. dr. ing. Cornelu BURILEANU
semnătura: .....

Student,
semnătura: .....

Responsabil program master,
Prof. dr. ing. Dragoș BURILEANU
semnătura: .....

Decan,
Prof. dr. ing. Cristian NEGRESCU
semnătura: .....

Cod Validare: **532ab4a645**

Anexa 5

Declaraţie de onestitate academică

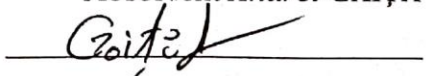
Prin prezenta declar că lucrarea cu titlul "*Metode de detectie a semnalelor audio specifice destinate implementărilor de tip embedded, cu aplicaţii în prevenirea exploatărilor forestiere ilegale*", prezentată în cadrul Facultăţii de Electronică, Telecomunicaţii şi Tehnologia Informaţiei a Universităţii "Politehnica" din Bucureşti ca cerinţă parţială pentru obţinerea titlului de Master în domeniul *Inginerie Electronică şi Telecomunicaţii*, programul de studii **BIOSINF**, 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 2018*

Absolvent *Andrei GAIŢĂ*



(semnătura în original)

CUPRINS

Lista figurilor

Lista tabelelor

Lista acronimelor

3G = Third Generation

ADT = Alternating Decision Tree

CASA = Analiza computaţională a scenei audio

DCT = Transformata Cosinus Discretă

FFT = Fast Fourier Transform
FIR = Filtru cu răspuns finit la impuls
FNR = False Negative Rate
FPR = False Positive Rate

GMM = Gaussian Mixture Model

HMM = Hidden Markov Model

LLD = Low Level Descriptor
LPC = Linear Prediction Coder

MFCC = Mel-frequency cepstrum
MLP = Multilayer Perceptron
MPEG-7 = Multimedia content description interface

PWP = Pachet integrat de analiză perceptuală wavelet

RF = Random Forests
ROC = Receiver operating characteristic

SNR = Signal-Noise Ratio
STI = Short Time Intensity
SVM = Support Vector Machines

TESPAR = Time Encoded Signal Processing and Recognition
TNR = True Negative Rate
TPR = True Positive Rate

WiMAX = Worldwide Interoperability for Microwave Access

Introducere

Despăduririle ilegale devin o ameninţare majoră pentru mediul înconjurător, având un impact devastator asupra pădurilor din întreaga lume. Printre efectele sale numărăm pierderi de biodiversitate şi schimbări climatice.

Există diverse soluţii pentru rezolvarea problemei despăduririlor. Una dintre acestea este monitorizarea pădurilor prin intermediul imaginilor satelitare de înaltă rezoluţie şi detecţia schimbărilor ce intervin, folosind metode de procesare a imaginilor. Marele dezavantaj al imaginilor satelitare este acela că procesarea imaginilor nu se face în timp real. Astfel, o detecție reuşită poate apărea prea târziu, atunci când o zonă întinsă de pădure este deja pierdută. Pentru a minimiza aceste pierderi, trebuie dezvoltat un sistem de alarmare sau notificare imediat după observarea unei mici schimbări în mediul respectiv. Un astfel de sistem s-ar dovedi mai eficient datorită faptului că detecţia este făcută în timp real, iar timpul de răspuns al autorităţilor responsabile este mai mic. De exemplu, un sistem de monitorizare video precum cel prezentat în [1] poate fi uşor modificat pentru a detecta şi tăieri ilegale pe lângă monitorizarea incendiilor. Autorii lucrării [1] propun o reţea video de supraveghere distribuită compusă din camere video digitale conectate la un server de *management* central via comunicaţiei *wireless* (pot fi folosite comunicaţii Wi-Fi, WiMAX, 3G etc.)

O altă abordare a unui sistem cu răspuns în timp real se bazează pe achiziţia şi procesarea de sunete. De obicei, în despăduririle ilegale vaste instrumentul cel mai folosit este motofierăstrăul. De aceea, o soluţie constă în detecţia şi urmărirea sunetului produs de un astfel de instrument.

Soluţia cunoscută sub numele de *Rainforest Connection* [2] foloseşte telefoane mobile pentru achiziţia şi procesarea de sunete, fiind alimentate cu energie solară. Fiecare telefon mobil este ataşat la un copac, fiind responsabil cu monitorizarea unei zone specifice. Dispozitivele sunt distribuite uniform în cadrul pădurii, formând o reţea celulară capabilă să detecteze tăieri ilegale de copaci într-o zonă mare a pădurii.

De-a lungul timpului, o problemă majoră în detecţia despăduririlor ilegale a fost determinarea exactă a coordonatelor sursei de sunet. În lucrarea [3] autorii prezintă o schemă de principiu pentru implementarea unei topologii de reţea celulară capabilă să determine locaţia exactă a sunetelor, prin minimizarea costurilor de implementare.

Paragrafele anterioare pun în evidenţă motivaţia alegererii acestei teme. Prin intermediul acestei lucrări, propun o nouă metodă de clasificare a semnalelor sonore ce pot fi întâlnite într-o scenă audio din pădure, sunete pe care le încadrez în două clase: cele produse de un motofierăstrău şi alte tipuri de sunete. Găsirea unor exemple negative de sunete precum vânt, ploaie, păsări ciripind, animale din pădure şi alte sunete nerelaţionate cu activităţi de despădurire, însă prezente în pădure, joacă un rol foarte important la performanţă metodei. Mai mult decât atât, discriminarea sunetelor între motoare de fierăstrău şi alte tipuri de motoare precum tractoare şi motociclete a fost luată în considerare pentru creşterea acurateţei detecţiei. În cazul exemplelor pozitive, distanţa dintre dispozitivul de achiziţie şi sursa de sunet trebuie să fie cât mai variată astfel încât să se obţină un sistem eficient, capabil să clasifice sunetele dintr-o anumită zonă. Pe parcursul dezvoltării sistemului, construirea bazei de date a fost un pas major şi deloc neglijat.

Lucrarea de faţă este structurată în 4 capitole, după cum urmează: primul capitol introduce noţiunile de bază ale determinării şi clasificării sunetelor, al doilea capitol pune în evidenţă cele mai frecvente metode de detecție a sunetelor din starea artei, următorul capitol descrie pe larg metoda propusă, iar în ultimul capitol se prezintă rezultatele metodei propuse, comparative cu alte metode convenţionale.

Capitolul 1

Metode de detecţie a modelelor audio

1.1 Scurtă introducere

Distrugerea constantă a mediului natural se dovedeşte tot mai mult a fi în detrimentul vieţii noastre, iar nevoia urgentă de a soluţiona această gravă problemă nu mai poate fi evitată. Există multe regiuni considerate a fi rezervaţii naturale, a căror protecţie survine din cauza pagubelor deja aduse zonei sau pentru prevenirea acestora în viitor. În ciuda anumitor măsuri luate de către autorităţi, implicarea oamenilor în activităţi ilegale precum tăierarea ilegală a copacilor, nu poate fi întotdeauna descoperită la timp. Această activitate este una dintre cele mai periculoase care duc la despăduriri rapide, lucru care s-a şi întâmplat în ultimele decenii pe întreaga planetă. De aici provin multe probleme sociale şi de mediu, precum eliminarea biodiversităţii, schimbări climatice, alunecări de teren, inundaţii frecvente şi chiar dispariţia anumitor specii de animale care trăiesc doar în pădure. Nu în ultimul rând, replantarea altor copaci şi ajungerea acestora la maturitate necesită o perioadă îndelungată de timp şi nu pot fi reparate multe dintre problemele enumerate înainte.

Detecţia imediată a activităţilor ce pot ameninţa mediul natural este vitală pentru a limita consecinţele negative cât mai mult. Monitorizarea constantă a anumitor regiuni este necesară pentru a prevedea conservarea acestora. Totuşi, supervizarea unei astfel de regiuni pentru o singură zi ajunge la costuri foarte ridicate, iar unele zone sunt greu de accesat în persoană. În multe cazuri, simpla prezenţă umană poate periclita ceea ce trebuie de fapt protejat. Din cauza acestor impedimente, se recurge la metode de supervizare autonome şi neasistate cunoscute sub numele de monitorizare automată. Sistemul propus este unul de supraveghere audio, pentru a diferenţia sunetele activităţilor umane faţă de sunetele mediului natural.

În momentul actual există o serie de astfel de sisteme destinate detecţiei sunetelor de drujbă sau alte altor instrumente de tăiere a copacilor. Un exemplu este proiectul *Rainforest Connection* [1], care foloseşte tehnologia telefoanelor mobile inteligente cu sistemul de operare Android şi capacitatea acestora de înregistrare audio pentru a determina dacă sunetele din pădure sunt produse de instrumente de tăiat copaci, sau sunt sunete naturale, specifice mediului respectiv (ploaie, animale, vânt etc.). Aceste telefoane mobile inteligente sunt montate în diverse puncte cheie din pădure, pentru a acoperi o suprafaţă cât mai mare, au datele mobile pornite constant, cu ajutorul cărora transmit datele audio înregistrate către un server central care realizează detecţia audio. Acestea sunt alimentate cu ajutorul energiei solare. Acest simplu sistem are multiple avantaje, precum: capacitatea de alertare a localnicilor cu ajutorul unei alarme într-un timp foarte scurt de la detecţia unui sunet maliţios, spre deosebire de sistemele ce folosesc imagini satelitare (o alarmă poate fi declanşată cu o întârziere de câteva zile de la preluarea acesteia), instrumentele şi tehnologiile folosite nu sunt create special pentru această întrebuinţare, iar costurile lor sunt destul de scăzute având în vedere nivelul de dezvoltare tehnologică actuală, detecţia sunetului de drujbă se face pe o rază de 1 km, ceea ce reprezintă o distanţă remarcabilă.

1.2 Concepte de bază şi terminologie

Recunoaşterea sunetului se bazează pe presupunerea că fiecare sursă de sunet pune în evidenţă un model acustic consistent, a cărui reprezentare este specifică modului de distribuţie a energiei. Acest model unic poate fi descoperit şi construit prin utilizarea algoritmilor de recunoaştere statistică a modelelor. În cadrul comunităţii specialiştilor în recunoaşterea de sunete, există două tipuri de clasificatori utilizaţi în acest scop: clasificatori discriminatori şi non-discriminatori. Primul tip de clasificatori este folosit pentru aproximarea unei limite între categoriile de date de antrenare. Câteva exemple de clasificatori discriminatori sunt: clasificatorul polinomial [4], perceptronul multi-strat [5] şi *Support Vector Machine (*SVM) [6]. Pe de altă parte, clasificatorii non-discriminatori sunt folosiţi pentru estimarea distribuţiei datelor de antrenare. Cele mai reprezentative exemple pentru acest tip de clasificatori sunt modelele mixte Gaussiene [4][5] şi modele Markov ascunse [6][7]. Principala caracteristică a acestui tip de clasificatori o reprezintă faptul că operează cu fiecare instanţă a unei clase, independent faţă de celelalte clase.

Într-un context general, clasificarea poate fi definită că acel proces de învăţare a asemănărilor şi deosebirilor dintre modele, care reprezintă instanţe ale unor obiecte dintr-o populaţie de obiecte neidentice. În cadrul unui proces de clasificare, obiectele sunt grupate în clase, conform asemănărilor şi deosebirilor sesizate. Astfel, pornind de la atribute specifice fiecărui obiect, se formează clase care descriu proprietăţile generale ale obiectelor respective. Obiectivul final al oricărui proces de clasificare este acela de a permite sistemului să identifice/recunoască obiectele din mediul său, pe baza proprietăţilor ce caracterizează clase de obiecte.

În consecinţă, în contextul problemei de clasificare, un sistem trebuie să parcurgă două etape:

- *clasificarea*, în cursul căreia sistemul învaţă caracteristicile generale ale claselor pe baza caracteristicilor specifice instanţelor,
- *recunoaşterea*, în cursul căreia sistemul suprapune caracteristicile unei instanţe peste caracteristicile claselor cunoscute şi determina clasa căreia aparţine instanţa respectivă.

Pentru majoritatea reţelelor neuronale artificiale se consideră că, pentru fiecare model din setul de antrenare, se cunosc atât vectorul de intrare, cât şi vectorul de ieşire. Algoritmii de antrenare încearcă să modifice structura reţelei (valorile ponderilor conexiunilor şi pragurilor neuronilor) astfel încât să asigure o corespondenţă cât mai bună între intrările şi ieşirile reţelei neuronale. Acest model de antrenare este cunoscut sub numele de *învăţare supravegheată*. În sinteză, se poate spune că orice model de antrenare pentru care se cunosc mărimile de ieşire intră sub incidenţa învăţării supravegheate.

Există însă numeroase situaţii practice - mai cu seamă în problemele de clasificare - în care nu se cunoaşte în prealabil răspunsul pe care ar trebui să-l furnizeze reţeaua pentru un anumit vector de intrare. În asemenea cazuri ar fi de dorit ca reţeaua să identifice de una singură caracteristicile datelor de intrare. Altfel spus, reţeaua ar trebui să "*înţeleagă*" singură modul în care trebuie să organizeze informaţia. Asemenea procese sunt cunoscute sub numele de *auto-organizare* sau *învăţare nesupravegheată* şi tratează cu precădere probleme de clasificare.

În cazul proceselor de auto-organizare, clasificarea mai este denumită şi *zonare* (în engleză, *clustering*). Această denumire provine de la dispunerea punctelor corespunzătoare vectorilor de intrare în anumite zone, sub forma unor nori sau ciorchini. În interiorul unei asemenea zone punctele sunt mai apropiate între ele sau în raport cu un centru comun, decât în raport cu centrele altor zone.

1.3 Clasificatori discriminatori

1.3.1 Clasificatorul polinomial

Reţelele polinomiale sunt cunoscute în literatură de mulţi ani, având proprietăţi excelente sub formă de clasificatori [8][1]. Datorită teoremei Weierstrass, clasificatorii polinomiali au devenit aproximatori universali ai clasificatorului optimal Bayes [9][10].

Metodele tipice de antrenare a clasificatorilor polinomiali sunt bazate în general pe metode statistice sau pe criteriul minimizării erorii pătratice medii. Accentul s-a pus cel mai mult pe clasificatorii polinomiali liniari sau de gradul doi. Ambele metode tradiţionale au avut însă limitări. Metodele clasice statistice estimează media şi covarianţa datelor audio pentru un model parametric. Dezavantajul acestei abordări este reprezentat de faptul că date din afara clasei nu sunt folosite pentru a maximixa performanţa. Metodele discriminatoare implică de cele mai multe ori matrici de mari dimensiuni, ceea ce conduce la probleme dificil de rezolvat.

1.3.2 Perceptronul multi-strat (MLP)

MLP reprezintă o reţea formată din neuroni simpli, cunoscuţi sub numele de perceptroni [11]. Conceptul de bază de simplu perceptron a fost introdus prima oară de către Rosenblatt în anul 1958. Perceptronul calculează o singură ieşire pe baza unor valori multiple şi reale de intrare, prin formarea unei combinaţii liniare în corelaţie cu tăria intrărilor, iar mai apoi ieşirile sunt trecute printr-o funcţie neliniară de activare. Din punct de vedere matematic, această relaţie poate fi scrisă sub următoarea formă:

$$y = \varphi(\sum_{i=1}^{n} w_i x_i + b) = \varphi(\mathbf{w}^T \mathbf{x} + b)$$

Figura 1.1 - Relaţia de definire a perceptronului [11]

unde w = vectorul tăriilor intrărilor,
x = vectorul intrărilor,
b = bias,
φ = funcţia de activare.

În figura 1.2 este reprezentat un grafic al fluxului de semnal al acestei operaţii.

Perceptronul iniţial considerat de Rosenblatt folosea pentru funcţia de activare o funcţie treaptă. În zilele actuale, mai ales în reţelele multistrat, funcţia de activare este deseori aleasă ca fiind tangentă hiperbolică sau funcţia sigmoid $1/1 + e^{-x}$. Aceste două funcţii sunt utilizate datorită faptului că sunt comode din punct de vedere matematic şi datorită faptului că sunt aproape liniare în apropiere de origine şi se saturează destul de rapid la îndepărtarea de origine. Acest lucru permite reţelelor cu perceptron multistrat să modeleze la fel de bine atât mapări neliniare puternice sau medii.

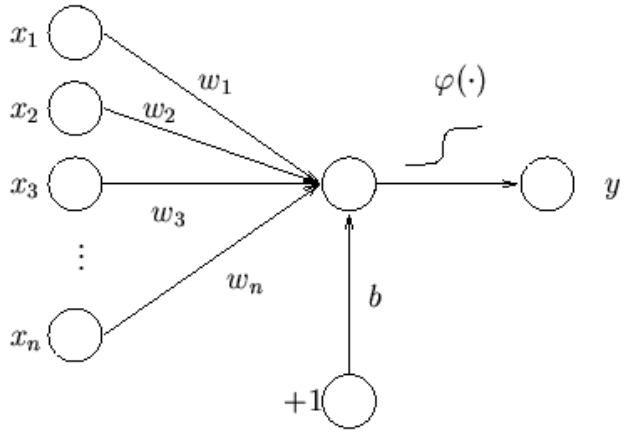


Figura 1.2 - Fluxul de semnal al operaţiei matematice MLP [11]

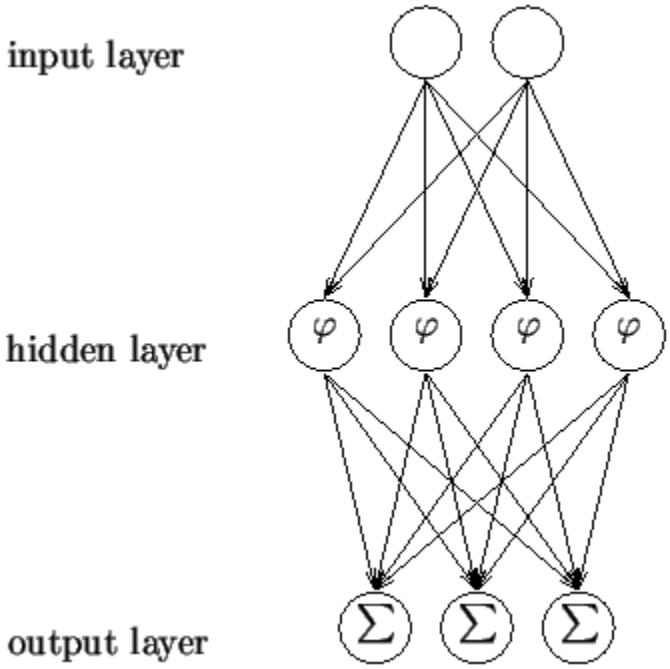


Figura 1.3 - Reţea neurală MLP [11]

Un singur perceptron nu este foarte folositor din cauza capacităţilor limitate de mapare. Totuşi, perceptronii pot fi folosiţi ca blocuri de construcţii pentru structuri mai mari şi mai practice. O reţea cu perceptron multistrat tipică este formată dintr-un set de noduri sursă ce formează un strat de intrare, unul sau mai multe straturi ascunse de noduri de calcul şi un strat al valorilor de ieşire. Semnalul de intrare se propagă prin reţea strat cu strat. O astfel de reţea este reprezentată în figura 1.3.

1.3.3 *Support Vector Machine (*SVM)

SVM reprezintă modele de învăţare supervizate, ce au asociate algoritmi de învăţare care analizează datele folosite pentru clasificare şi analiză [12]. Pentru un set de exemple de antrenare, fiecare marcat ca aparţinând unei categorii, un algoritm de antrenare SVM construieşte un model ce asignează noi exemple într-o categorie sau în alta. Astfel, acest clasificator este unul liniar, binar şi non-probabilistic. În cadrul modelului SVM, exemplele de antrenare sunt reprezentate ca puncte în spaţiu, mapate astfel încât exemplele din cadrul unei categorii sunt divizate clar de un gol cât mai lat posibil. Noi exemple sunt mapate în acelaşi spaţiu şi se prezice cărei categorii aparţin.

În mod formal, un SVM construieşte un hiperplan sau un set de hiperplanuri într-un spaţiu multi-dimensional sau chiar infinit-dimensional, care pot fi folosite pentru clasificare, regresie sau alte cerinţe. O separare bună a punctelor în spaţiu este realizată în cadrul unui hiperplan care are cea mai mare distanţă până la cel mai apropiat punct din cadrul setului de date de antrenare (cunoscută şi sub numele de margine).

Situaţia originală care implică un spaţiu finit-dimensional s-a dovedit a nu fi îndeajuns de puternică pentru a separa liniar punctele din spaţiu. De aceea, s-a recurs la ideea de spaţiu infinit-dimensional, cu presupunerea că separarea punctelor se va face mult mai uşor.

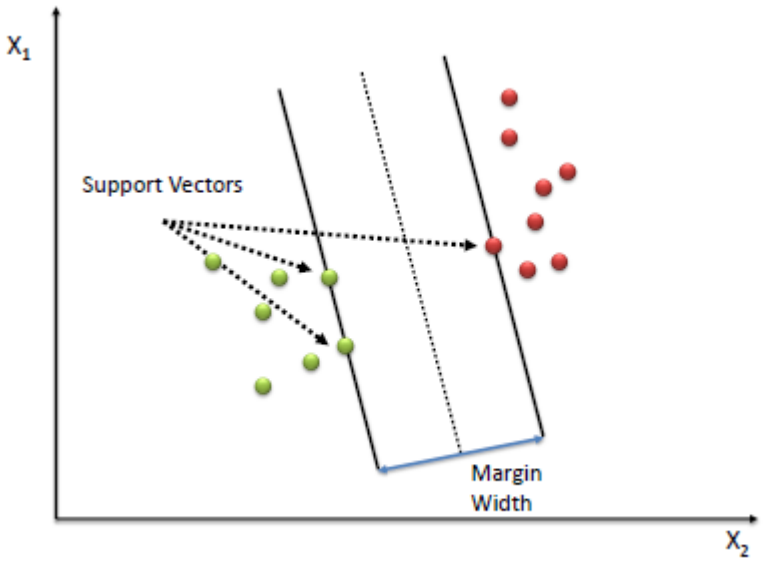


Figura 1.4 - Planul punctelor divizate în două categorii [12]

1.4 Clasificatori non-discriminatori. Modele Markov ascunse

Modelele Markov ascunse (HMM) reprezintă un utilitar impresionant pentru modelarea datelor [13]. Acestea sunt folosite în aproape toate sistemele curente de recunoaştere a vorbirii, în numeroase aplicaţii de biologie moleculară, în compresia de date şi în alte arii de inteligenţă artificială şi recunoaştere de modele. Recent, modelele Markov ascunse au început să fie folosite şi în aplicaţii de *computer vision*.

Modelul Markov ascuns reprezintă o modalitate de a reprezenta distribuţii de probabilităţi pe baza unor secvenţe de observare. Observaţia se face la un anumit moment de timp t, de către o variabilă Y_t. Aceasta poate fi un simbol al unui alfabet discret, o variabilă reală, un întreg sau orice alt obiect, atâta timp cât putem defini o distribuţie de probabilitate asupra ei. Putem asuma faptul că observaţiile sunt făcute la intervale de timp discrete, egal distanţate, astfel că *t* poate fi un index de timp de valoare întreagă.

Modelul Markov ascuns are acest nume datorită a două proprietăţi definitorii. Prima se referă la faptul că observaţia de la timpul *t* a fost generată de către un proces a cărui stare S_t este ascunsă de observator. A doua proprietate stipulează faptul că starea procesului ascuns satisface următoarea proprietate Markov: cunoscând valoarea stării anterioare S_{t-1}, starea curentă S_t este independentă de toate stările anterioare lui *t-1*. În alte cuvinte, starea unui proces la un anumit moment de timp încapsulează toate informaţiile necesare pe care trebuie să le cunoaştem legate de istoria procesului, astfel că putem prezice viitorul procesului.

În urma celor prezentate, putem reprezenta distribuţia compusă a tuturor secvenţelor de stări şi de observaţii într-un produs de tipul:

$$P(S_{1:T}, Y_{1:T}) = P(S_1)P(Y_1|S_1)\prod_{t=2}^{T} P(S_t|S_{t-1})P(Y_t|S_t),$$

Figura 1.5 - Relaţia distribuţiei compuse [13]

Capitolul 2

Metode de detecţie audio a drujbelor din starea artei

2.1 Metoda TESPAR

Conceptele TESPAR [14] sunt procesele prin referinţa la studii de caz recente care pun în evidenţă clasificări ale semnalelor aflate într-un spectru de frecvente larg, incluzând lungimi cu una de ordinul nanosecundelor până la lungimi de undă în domeniul sub-Herzi. Caracteristicile de bază ale metodei TESPAR în zona de procesare a vorbirii sunt următoarele:

- capacitatea de a separa şi clasifica semnale de interes ce sunt insesizabile în domeniul frecvenţei,
- abilitatea de a coda forme de undă variabile în timp ale vorbirii în configuraţii optime pentru procesare de către reţele neurale artificiale,
- abilitatea de a desfăşura, din punct de vedere economic, arhitecturi paralele masive pentru producerea fuziunii de date.

În cadrul literaturii de specialitate a teoriei informaţiei, prima teoremă a lui Shannon legată de eşantionarea unui semnal analog stă la baza unei varietăţi de transformări liniare, precum Fourier sau predicţia liniară. Astfel, strategiile de codare actuale implică un grup de trei necesităţi:

a) utilizarea descriptorilor de amplitudine,
b) utilizarea eșantionarii uzuale,
c) un domeniu de aproximare, bazat fie pe magnitudine, fie pe amplitudine, dependent de numărul de biţi per eşantion, număr folosit pentru a procesa valorile eşantionate în ordine.

În aceeaşi perioadă cu faimoasa lucrare a matematicianului Shannon care a enunţat teorema eşantionării, alţi oameni de știință investigau efectele "limitării amplitudinii" pentru formele de undă ale semnalului vocal, un proces folosit pe larg în comunitatea amatorilor radio. În continuare, conceptul de limitare a amplitudinii a fost extins către formatul de limitare infinită, la care toată informaţia despre amplitudine este eliminată din cadrul semnalului, rezultând o transformare binară ce conservă doar punctele de trecere prin 0 ale semnalului original. Folosind metoda limitării amplitudinii, inteligibilitatea aleatoare a cuvintelor din cadrul unui semnal vocal atingea un scor de aproximativ 97.9%, lucru ce demonstrează clar faptul că o proporţie substanţială a informaţiei de interes din cadrul unu semnal de acest tip se află cu precădere în trecerile prin 0. Mai mult decât atât, aceste dovezi pun sub semnul întrebării cele 3 proprietăţi a), b) şi c) enunţate mai sus. Odată ce s-au eliminat toate informaţiile legate de valorile intermediare, eşantioanele obţinute reprezintă "durata" intervalelor dintre trecerile prin 0 ale formei de undă. Astfel, aceste durate bazate pe trecerile prin 0 sunt derivate din semnal şi nu generate la intervale de timp egale, iar aceste eşantioane vor exista la distanţe inegale. Toate aceste observaţii au furnizat catalistul intelectual care a rezultat în crearea şi dezvoltarea metodei TESPAR.

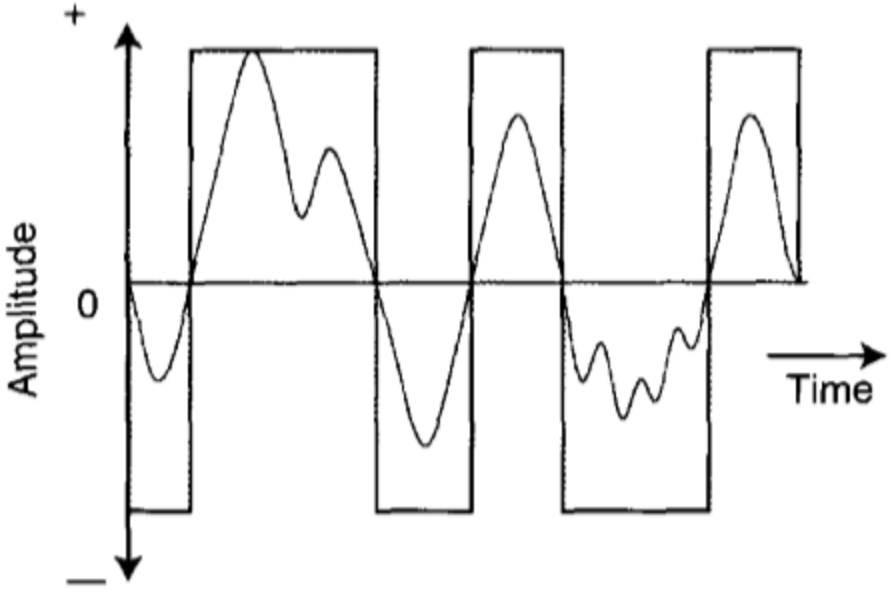


Figura 2.1 - Procesul de limitare a amplitudinii [14]

Desigur, orice metodă prezintat atât avantaje, cât şi dezavantaje. De aceea, câţiva oameni de ştiinţă au propus introducerea unui concept în cadrul modelului limitării infinite, fiind cunoscut sub numele de zerouri complexe. Problema practică a extragerii zerourilor, precum construirea unei secvenţe de zerouri reale şi complexe care să corespundă unei forme de undă date nu este trivială. Zerourile reale ale unei funcţii, care reprezintă de fapt convenţionalele treceri prin 0 ale semnalului sunt mult mai uşor de determinat decât zerourile complexe. Inspecţia vizuală a formei de undă ar putea furniza informaţii legate de locaţia a câteva zerouri complexe, însă singura metodă cantitativă cunoscută pentru a găsi locaţia tuturor zerourilor complexe implică factorizarea numerică a $2\mathrm{TW}^{th}$- ordinul unei polinom trigonometric. Calculul unei astfel de funcţii nu este fezabil, de aceea metoda TESPAR nu foloseşte de acest calcul. Procedura propusă de TESPAR presupune segmentarea semnalului între zerouri reale succesive şi combinarea acestei durate cu aproximări simple ale formei semnalului între aceste două locaţii. Mai simplu, determinarea şi clasificarea zerourilor complexe se poate face direct din forma de undă.

Într-o astfel de împărţire a semnalului în zerouri reale şi complexe, se observă faptul că zerourile reale din domeniul timp sunt identice cu locaţiile zerourilor reale din domeniul zero. Zerourile complexe apar în perechi conjugate, ce sunt asociate cu perturbări precum minime, maxime, puncte de inflexiune. În cadrul celei mai simple implementări TESPAR, doar doi descriptori asociaţi fiecărui segment pot fi folosiţi pentru a genera un alfabet de simboluri TESPAR de bază:

- durata dintre 2 zerouri reale succesive,
- forma semnalului între cele 2 zerouri succesive.

În modelul de bază TESPAR nu toate zerourile complexe pot fi identificare din forma semnalului, ceea ce înseamnă că aproximarea este limitată doar la acele zerouri ce pot fi determinate. Fiecare segment al semnalului eşantionat poate fi clasificat cu ajutorul a două elemente: durata (D) şi numărul de minime (S).

2.2 Metode bazate pe funcţia de autocorelaţie

2.2.1 Detecţie simplă bazată pe metoda autocorelaţiei

2.2.1.1 Descriere generală

Această primă metodă de detecţie audio a drujbelor este menţionată în lucrarea [15], ţinându-se cont de limitările unui dispozitiv *embedded* care are capacitatea de a capta sunetul extern şi de a realiza o procesare uşoară asupra acestuia, pentru ca apoi să transmită datele prin GPRS către o staţie de bază aflată în apropiere. Aceste dispozitive *embedded* au capacitatea de a înregistra sunete cu frecvenţă de eşantionare mare. Totuşi, deoarece sunetele produse de un motofierăstrău au frecvenţă joasă, iar memoria dispozitivului *embedded* utilizat este limitată, se foloseşte o frecvenţă de eşantionare de 8333 Hz, cu 10 biţi per eşantion. Pentru a uşura calculele şi pentru minimizarea volumului de date, se realizează o decimare cu un raport 1:4 şi o filtrare trece-jos, cu ajutorul unui filtru FIR. Pentru a putea face o detecţie corectă într-un interval de distanţe 0-50 metri faţă de dispozitiv, s-a implementat un mecanism de auto-reglaj al amplitudinii. În urma acestor optimizări, dispozitivul *embedded* are capacitatea de a stoca şi de a procesa un număr de 4 eşantioane audio succesive, fiecare dintre acestea cu o durată de 1.8 secunde.

2.2.1.2 Extragerea parametrilor

Conform lucrării [16], se cunoaşte faptul că sunetul produs de motoarele drujbelor se încadrează între 100 şi 300 Hz, caracteristică ce depinde de producător şi de versiunea motorului. Pot exista şi alte surse acustice care să producă sunete în aceeaşi gamă de frecvenţe, însă sunetul motorului de drujbă are o amprentă spectrală diferită. S-ar putea folosi spectrul de frecvenţe ca parametru de detecţie, însă calculul transformatei Fourier rapide depăşeşte cu mult capacitatea de procesare a dispozitivului *embedded* utilizat de autorii lucrării precizate anterior, astfel că este nevoie de o altă soluţie în acest sens. Aceştia au decis în final să folosească funcţia de autocorelatie, deşi pare să aibă o complexitate de calcul mai mare (n x n), spre deosebire de transformată Fourier (n x log n), însă ambii termeni ai sumei sunt valori ale eşantioanelor semnalului şi se exclude din start calculul unei funcţii exponenţiale.

Formula generală a funcţiei de autocorelaţie este următoarea:

$$R(m) = \sum_{n=-\infty}^{\infty} s_n \cdot s_{n+m}$$

în timp ce în cazul de faţă aceasta este:

$$R(m) = \sum_{n=0}^{|s|} (s_n - \bar{s}) \cdot (s_{((n+m)\ mod|s|)} - \bar{s})$$

unde s_n este eşantionul de intrare nr. n , $\bar{s}$ este media acestora şi $|s| = 256$ este nr. de eşantioane de intrare. Rezultatul acestei funcţii este ilustrat în figura 2.2 unde se poate observa diferenţa dintre un sunet produs de un motofierăstrău şi un sunet produs de vânt.

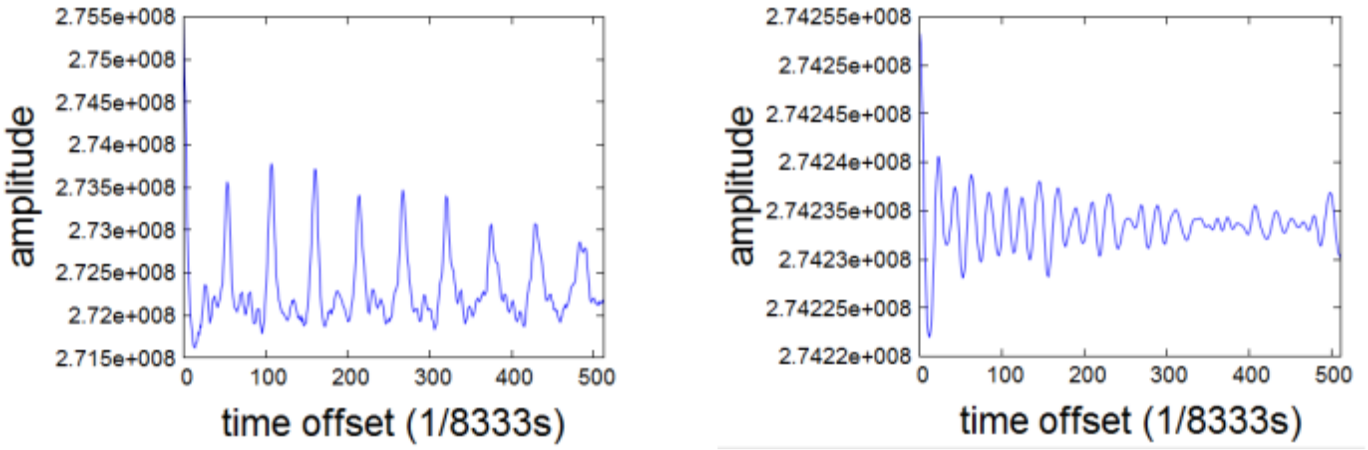


Figura 2.2 - Rezultatul autocorelaţiei în cazul sunetului produs de drujbă (în stânga) şi în cazul unui sunet produs de vânt (în dreapta) [15]

Se poate observa din figura 2.2 că funcţia de autocorelație prezintă maxime periodice în cazul sunetului produs de drujbă, în timp ce în cazul sunetului produs de vânt nu există caracteristici vizibile definitorii. Pentru început se determină maximul local, după care se efectuează extragerea unor parametri din această funcţie. În continuare, sunt calculate următoarele caracteristici:

- distanţa medie între maximele locale max_i:

  $$avgDist = \frac{1}{|max|}\sum_{i=1}^{|max|} dist_i$$

  unde max este numărul de maxime locale şi

  $$dist_i = poziţie(max_i) - poziţie(max_{i-1})$$

- varianţa distanţei între minimele locale:

  $$varianţa = \frac{1}{|max|}\sum_{i=1}^{|max|}(dist_i - avgDist)^3$$

- numărul de maxime locale:

  $$|max|$$

- energia pe termen scurt a semnalului de intrare:

  $$STE = \sum_{n=0}^{|S|} s_n^2 \ , n \in N$$

2.2.1.3 Clasificarea la nivelul fiecărui dispozitiv

Conform precizărilor anterioare, memoria dispozitivului este limitată ceea ce impune stocarea a maxim 4 eşantioane, fiecare cu o durată maximă de 1/8 secunde. Durata de extragere şi de clasificare a parametrilor este de 3 secunde. Clasificarea s-a făcut folosind diferiţi arbori de decizie şi anume : *Alternating Decision Tree* (ADT) [17], *Best-First decision tree* [18], *Decision Stump* [19], *J48* (C4.5) [20], *J48Graft* (grafted C4.5) [21], *LogitBoost Alternating Decision Tree* (LADTree) [17], *Random Forest* [22], *Random Tree* [19], *Reduced-Error Pruning Tree* [19], *Simple Classification and Regression Tree* (CART) [23]. Pentru recunoaşterea tiparelor, arborii de decizie sunt buni clasificatori, iar principalele avantaje în această situaţie sunt: implementare facilă, care necesită un nivel scăzut de memorie şi sunt destul de rapizi.

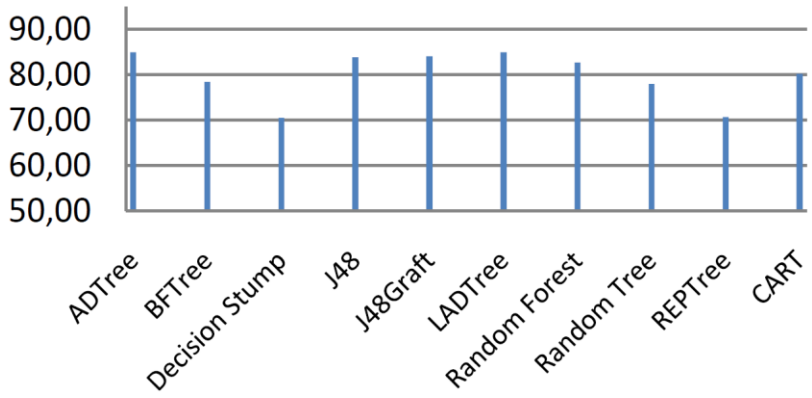


Figura 2.3 - Rata de identificări cu succes folosind diferiţi arbori de decizie [15]

Figura 2.3 ilustrează rata identificării cu succes, în care se folosesc mai mulţi clasificatori bazaţi pe arbori de decizie, evaluaţi pe prima de bază. Pentru o imagine de ansamblu a acestor date, trebuie luate în considerare rata de identificare negativă şi alarmă falsă, deoarece într-un sistem real de supraveghere o intervenţie a echipajului uman ar produce costuri suplimentare. Astfel, în figura 2.4 se observă rata de identificare cu succes raportată la numărul de alarme false produse de fiecare algoritm de clasificare în parte:

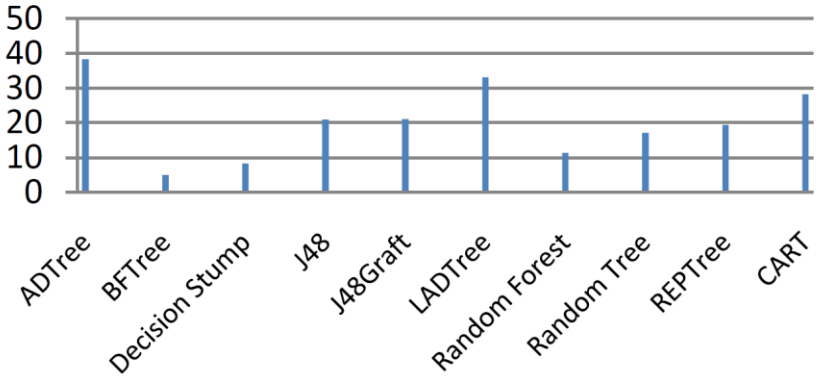


Figura 2.4 - Rata de identificări pozitive normată la rata de identificări negative [15]

Analizând figura 2.4, observăm că algoritmul de clasificare ADT oferă cel mai bun rezultat, atât din punctul de vedere al identificărilor pozitive, cât şi din punctul de vedere al celor negative. ADT (arbore de decizie alternativă) este construit din clasificatoare slabe. Pentru a obţine rezultate cu o performanţă mai ridicată se poate folosi o tehnică de îmbunătăţire (*boosting).* La ieşirea acestui clasificator binar avem un număr real între 1 şi 100. Clasificarea binară se face setând un prag pentru valoarea de la ieşire, de regulă 50. În Tabelul 2.1 este prezentată matricea de confuzie pentru arborele ADT.

	Decizie	
Sunet	Motofierăstrău	Altceva
Motofierăstrău	76.19	23.81
Altceva	2.67	97.33

Tabelul 2.1- Matricea de confuzie (în procente %) a arborelui de decizie ADT [15]

2.2.2 Detecţia acustică bazată pe analiza parametrilor extraşi din autocorelaţia semnalului

Metoda propusă are la bază informaţii esenţiale, precum banda de frecvenţă în care se găsesc cele mai importante esenţiale spectrale din semnalul sonor produs de un motofierăstrău. Implementarea unor algoritmi complecşi cu ajutorul unui sistem *embedded* presupune o serie de probleme referitoare la timpul de calcul până la obţinerea rezultatelor (riscându-se astfel pierderea unor informaţii) şi la aspectul energetic: reducerea timpului de calcul implică un sistem mai performant, dar în majoritatea cazurilor acesta va impune o putere electrică ridicată. Ne propunem astfel implementarea unui algoritm care să fie capabil de realizarea unei decizii binare pe baza unor criterii cât mai simple.

Calculul funcţiei de autocorelatie poate avea o complexitatee ridicată de calcul, mai ales din punct de vedere temporal. Funcţia de autocorelaţie a unei secvenţe de lungime finită (s_n), cu $|s|$ elemente, este următoarea:

$$R_{ss}(m) = \sum_{n=0}^{|s|} (s_n - \overline{s})(s_{((n+m) mod |s|)} - \overline{s})$$

Figura 2.5 - Relaţia funţiei de autocorelaţie

unde $\bar{s}$ reprezintă media secvenţei. Datoria simetriei ecuaţiei, aceasta trebuie calculate doar pentru m = 0, 1, 2, …, $\frac{|s|}{2}$.

În cadrul metodei autocorelației, se calculează valoarea a patru descriptori ai secvenţei de autocorelație:

- STI (Short Time Intensity) - primul element al secvenţei autocorelației:

$$STI = R_{ss}(0) = \sum_{n=0}^{|s|} s_n^2, n \in$$

Figura 2.6 - Primul element al secvenţei autocorelației

- numărul maximelor locale (N_p) în funcţia $R_{ss}(m)$. Se consideră că fiecare vârf al secvenţei de autocorelație este un maxim local dacă valoarea sa este mai mare decât un anumit prag (α) ce depinde de STI şi dacă nu mai are niciun alt maxim într-o fereastră de căutare dată (lăţimea ferestrei este β).
- distanţa medie dintre două maxime locale:

$$avgDist = \frac{1}{N_p - 1} \sum_{i=2}^{N_p} D_i$$

Figura 2.7 - Distanţa medie dintre două maxime locale

unde D_i reprezintă distanţa dintre *ith* şi vârful *i - ith.*

- variaţia distanţelor dintre maximele locale (uniformitatea frecvenţei fundamentale):

$$variance = \frac{1}{N_p - 1} \sum_{i=2}^{N_p} (D_i - avgDist)^2$$

Figura 2.8 - Variaţia distanţelor dintre maximele locale

În cadrul celor 4 descriptori, STI poate fi considerat un descriptor slab, deoarece depinde în mare parte de setările de câştig şi de distanţa dintre microfon şi sursa sunetului. Totuşi, valoarea acestui descriptor poate fi normalizată înainte de clasificare, ştiind valorile de câştig ale amplificatorului.

În cadrul creării unui sistem de clasificare eficient, nu doar descriptorii trebuie să fie creați corespunzător, ci şi mecanismele de decizie, pentru a genera rezultate precise şi corecte. Există multe instrumente bine cunoscute de clasificare ce pot fi considerate, precum reţelele neurale, SVM, arbori de decizie. Limitările platformei *hardware* legate de puterea de calcul şi utilizarea memoriei trebuie avute în vedere. Arborii de decizie pot fi uşor aplicaţi, există mecanisme de comparare uşor de aplicat cu ajutorul operaţiilor cu valori întregi (spre deosebire de SVM, care evaluează cereri prin adunări şi înmulțiri de numere reale).

2.3 Analiza computaţională a scenei audio (CASA)

2.3.1 Descriere

Lucrarea [24] propune o abordare folosind tehnologia CASA [25], o tehnologie în continuă dezvoltare care îşi propune descrierea completă a unei scene acustice cu ajutorul unei serii de metode şi algoritmi. Pentru a realiza o descriere completă a unei scene acustice este nevoie de separarea surselor de sunet diferite, localizarea acestora în contextul scenei şi recunoaşterea individuală a fiecăreia în parte. Lucrarea [25], considerată una de referinţă în acest domeniu, pune în evidenţă principii de bază, algoritmi, arhitecturi de sisteme şi potenţiale aplicaţii pentru această tehnologie. Lucrarea abordează diverse metode absolut necesare într-un sistem CASA:

- estimarea multiplelor frecvenţe fundamentale,
- abordări CASA bazate pe vectori de caracteristici şi modele,
- separarea sunetelor bazată pe localizarea în spaţiu,
- tehnici de procesare audio în medii reverberant,
- separearea semnalului de vorbire faţă de semnalul de muzică,
- recunoaşterea automată a vorbirii în medii cu zgomot,
- modelarea neurală a scenei audio.

Lucrarea menţionată anterior este structurată în două mari părţi: prima abordează extragerea şi analiza seturilor de caracteristici pe baza cărora se va face clasificarea, iar cea de-a doua parte pune în evidenţă modul de aplicare a algoritmilor modelelor Markov ascunse (HMM) pentru rezolvarea problemelor de detecţie şi clasificare a sunetelor.

2.3.2 Parametri acustici

În cadrul lucrării de faţă, se folosesc 3 tipuri de parametri acustici pentru clasificare:

- *coeficienţi Mel-Cepstrali (**MFCC**)* : sunt folosiţi 23 astfel de coeficienţi. După filtrarea transformatei Fourier rapide cu un banc de filtre triunghiulare în scară Mel a ferestrelor de semnal, se aplică logaritmarea şi transformata Cosinus Discretă (**DCT**) pentru a obţine aceşti parametri. Aceştia sunt importanţi din puncul de vedere al percepţiei umane asupra sunetelor. Ultima etapă din acest algoritm de extragere este foarte importantă, având rolul de a decorela datele, folosind puţine resurse computaţionale.
- *descriptori audio MPEG-7 de nivel redus (**LLD**)* : protocolul audio MPEG-7 oferă o serie de unelte cu scopul de a genera automat un descriptor capabil să ilustreze informaţii esenţiale despre semnalul audio. Aceşti parametri sunt prezentaţi în Tabelul 3.2. iar o prezentare mai detaliată se poate găsi în lucrarea [15].

Descriptor	Dimensiune	Abreviere
Audio Waveform	2	AW
Audio Power	1	AP
Audio Specrum Centroid	1	ASC
Audio Spectrum Spread	1	ASS
Audio Spectrum Flatness	19	ASF
Harmonic ratio	1	HR
Upper Limit of Harmonicity	1	ULH
Audio Fundamental Frequency	1	AFF

Tabelul 2.2 - Descriptorii audio MPEG-7 de nivel redus [24]

- *pachet integrat de analiză perceptuală wavelet (PWP):* parametrii acustici prezentaţi în tabelul 2.2 oferă o definire a semnalului atât în domeniul timp, cât şi în domeniul frecvenţă. Lucrarea [15] pune în evidenţă cum parametri din domenii diferite pot îmbunătăţi performanţa de clasificare a întregului sistem. Spre deosebire de sinusoidele predictibile, funcţiile *wavelet* tind să fie neregulate şi asimetrice.

Cei trei parametri au fost folosiţi separat la început pentru a stabili nivelul de eficienţă al fiecăruia. În acest scop, au fost dezvoltate o serie de experimente pentru a stabili eficienţa.

2.3.3 Metodologia de estimare a modelelor audio

Recunoaşterea sunetelor se bazează pe faptul că orice sursă de semnal audio este caracterizată printr-un model de distribuţie al energiei în funcţie de frecvenţă. Acest model poate fi determinat folosind algoritmi statistici de recunoaştere a formelor. În cadrul comunicaţiei, cei mai folosiţi clasificatori ce se ocupă de recunoaşterea sunetelor se împart în două categorii, aşa cum am precizat în capitolul precedent: clasificatori discriminatori şi clasificatori non-discriminatori. Clasificatorii discriminativi încearcă să găsească o suprafaţă de separaţie între clasele date spre antrenare.

Această metodă folosește teoria Modelelor Markov Ascunse, unde fiecare stare este modelată prin Mixuri Gaussiene. Algoritmul Baum-Welch a fost folosit pentru a antrena un HMM pentru fiecare clasă de sunet în parte. Acesta împarte secvenţa de succesiune a caracteristicilor într-un număr predefinit de stări şi învaţă modul de asociere al acestora. Rezultatul este o matrice de tranziţie compusă din elemente ce indică probabilitatea succesiunii între stări. Un HMM este definit de următoarele elemente:

- numărul de stări, S,
- numărul de observaţii posibile, O,
- tipul de distribuţie pentru fiecare stare în parte, în acest caz fiind vorba despre GMM,
- matricea de tranziţie, ce conţine probabilităţile de succesiune de la o stare la cealaltă,
- distribuţia stărilor iniţiale, ce indică probabilitatea ca HMM-ul să înceapă dintr-o anumită stare.

2.3.4 Dezvoltarea sistemului

În scopul determinării topologiei care asigură cea mai mare precizie din punctul de vedere al clasificării, s-a conceput un experiment folosind ceoficienţii MFCC. Acest experiment constă din două etape: prima etapă foloseşte o singură stare, iar cea de a doua etapă utilizează o schemă ierarhică cu scopul principal de a separa activităţile umane de restul. Setul de date folosit pentru antrenare şi testare are un rol esenţial în privinţa rezultatelor sistemului construit, de aceea s-au înglobat diverse sunete dintr-o gamă largă, precum sunete folosite în industria cinematografică. Baza de date trebuie de asemenea să ofere o calitate ridicată a sunetelor şi un număr mare de înregistrări. Topologia în cauză este reprezentată în figura 2.9, iar caracteristicile bazei de date audio sunt descrise în tabelul 2.3:

Categorie audio	Nr. de eşantioane audio	Durata totală [s]
Păsări	187	6.159,78
Motocicletă	79	1.169,2
Maşină	81	2.195,2
Ploaie	53	2.209,8
Vorbire	1680	5.073,6
Sunet de armă	131	1.803,87
Vânt	66	3.234
Total	2277	21.845,35

Tabelul 2.3 - Structura bazei de date audio [24]

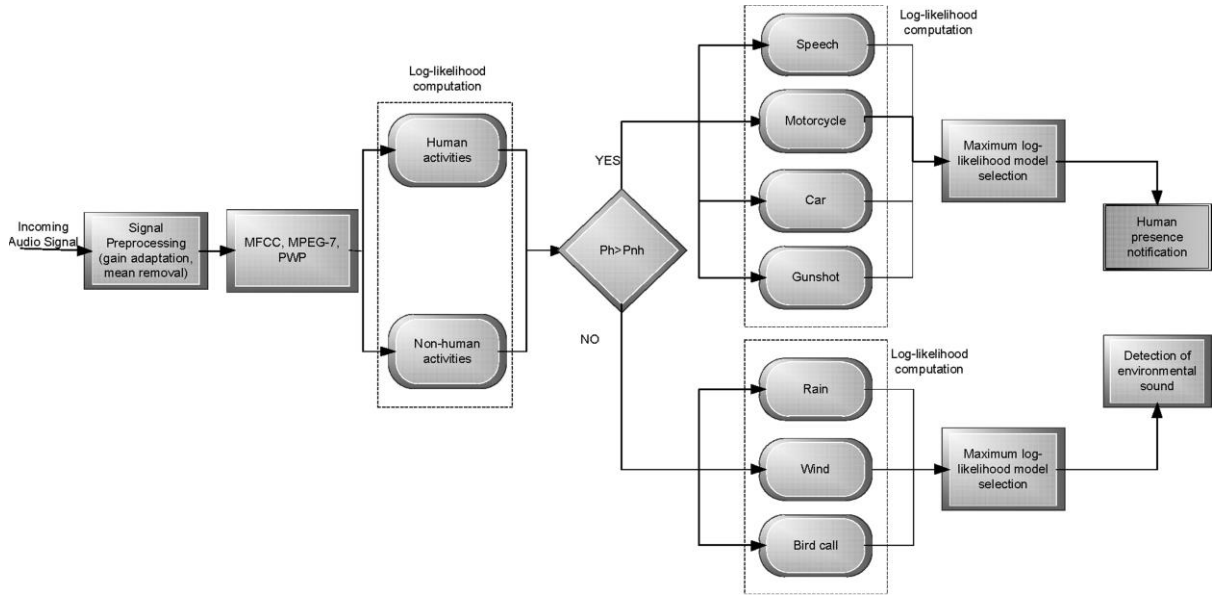


Figura 2.9 - Schemă bloc ce ilustrează cei doi paşi din topologia pentru detecţia prezenţei umane în mediile naturale [24]

Din cadrul acestor date, un procent de 70% a fost folosit pentru antrenare, iar restul de 30% a fost utilizat pentru procesul de evaluare. Datele au fost alese în mod aleator pentru a asigura relevanţa rezultatelor.

Din diagrama bloc a sistemului de detecţie ataşată anterior observăm că în prima fază, sistemul realizează o clasificare generalizată, în care se încadrează eşantioanele de sunet în două clase pe baza tipului de sursă audio: artificială (activitate umană) sau naturală (fenomen al naturii, sunet generat de animale/păsări). Scopul lucrării este determinarea sursei sunetului din punctul de vedere al activităţilor umane, deci zona superioară a graficului prezintă o importanţă deosebită pentru această lucrare.

Standardul MPEG-7 recomandă extracţia parametrilor de nivel redus la nivelul unei ferestre de 30 ms cu o suprapunere de 10 ms. Fereastra folosită este cea Hamming, cu scopul minimizării discontinuităţilor, iar lungimea FFT este 512.

2.3.5 Rezultate

Precizia totală obţinută este 81.5%, în timp ce precizia pentru primul pas al topologiei este 87.7%. Cele două matrici de confuzie obţinute la pasul 2 sunt prezentate în tabelele 2.4 şi 2.5.

	Răspuns			
Tip	Motocicletă	Maşină	Vorbire	Sunet de armă
Motocicletă	70.1	24.3	0	5.6
Maşină	20.4	75.8	0	3.8
Vorbire	0	0	100	0
Sunet de armă	15.7	3.1	0	80.2

Tabelul 2.4 - Matricea de confuzie pentru sunete specifice activităţii umane [24]

	Răspuns		
Tip	Motocicletă	Maşină	Vorbire
Motocicletă	70.1	24.3	0
Maşină	20.4	75.8	0
Vorbire	0	0	100
Sunet de armă	15.7	3.1	0

Tabelul 2.5 - Matricea de confuzie pentru sunete specifice naturale [24]

2.4 Metoda LPC

Metoda codării prin predicţie liniară [26] modelează fiecare eşantion al semnalului audio pe baza sumei mediate a eșantioanelor precendente. Coeficienţii sunt folosiţi pentru a realiza filtrul liniar de predicţie. Predictorul liniar calculează o anvelopă spectrală adaptată special pentru elementul audio, alegând convenabil ordinul acestuia. Datorită erorii minimizate de către predictorul liniar, vârfurile spectrale sunt puse în evidenţă în cadrul anvelopei, la fel ca în sistemul auditiv.

Pentru analiza LPC, intrarea predictorului liniar este reprezentată de eşantioanele semnalului audio digital *x(n)* (un proces staţionar şi aleator în sens larg), iar ieşirea este obţinută sub forma unui set de *M+1* coeficienţi, $b_{M,i}$, $i = \overline{0,M}$, unde M este ordinul filtrului liniar de predicţie. Coeficienţii $b_{M,i}$ sunt aleşi pentru a prezice cu acurateţe cât mai mare valoarea curentă a datelor eşantionate, *x(n)* dintr-un set de *M* cele mai recente eşantioane. Funcţia de estimare, sau de predicţie, are următoarea formă:

$$\hat{x}_M(n) = -\sum_{i=1}^{M} b_{M,i} x(n-i)$$

Figura 2.10 - Funcţia de estimare (predicţie) [26]

Unde $b_{M,i} = [1\ b_1\ ...\ b_M]$. Eroare de predicţie $e_M(n)$ poate fi calculate ca diferenţa dintre semnalul audio actual şi cel estimat:

$$e_M(n) = x(n) - \hat{x}_M(n) \overset{(1)}{=} x(n) + \sum_{i=1}^{M} b_{M,i} x(n-i)$$

Figura 2.11 - Eroarea de predicție [26]

Eroarea de predicţie reprezintă toate noile informaţii care intră în semnalul $\widehat{x_M}(n)$, la momentul n. Deoarece informaţia este nouă, eroarea de predicţie este "imprevizibilă". Componenta ce poate fi prezisă nu conţine informaţii noi. Eroarea de predicţie poate fi considerată a fi ieşirea unui filtru FIR cu următoarea funcţie de transfer:

$$H_M(z) = 1 + \sum_{i=1}^{M} b_{M,i} z^{-i}$$

Figura 2.12 - Funcţia de transfer a filtrului FIR [26]

ca răspuns la semnalul staţionar în sens larg de la intrare. Varianța erorii de predicţie este dată de ecuaţia:

$$\sigma_e^2 = E\left\{\left[e_M(n)\right]^2\right\} = \sum_{n=1}^{M}\left[e_M(n) - \bar{e}_M\right]^2$$

Figura 2.13 - Varianța erorii de predicţie [26]

unde $\bar{e}_M$ reprezintă media aritmetică a erorii de predicţie $e_M(n)$.

Cele două metode clasice implicate în predicţia liniară sunt metoda autocorelatiei şi metoda covarianţei. În cazul metodei autocorelației, recursia Levinson-Durbin este utilizată pentru a rezolva ecuaţiile normale care apar din formula celor mai mici pătrate. Metoda covarianţei este bazată pe o estimare imparţială a autocorelației.

Random Forests (RF) este unul dintre cei mai cunoscuţi algoritmi ce foloseşte arbori de decizie în scopul unui clasificator de bază. Arborii de decizie sunt folosiţi din ce în ce mai mult în momentul actual, reprezentând modele predictive de învăţare automată (ca ramură a inteligenţei artificiale) ce decid valoarea ţintă a unui nou eşantion, bazându-se pe diverse atribute ale datelor disponibile.

Construirea unui arbore de decizie este conformă cu procesul general de creare a unui ansamblu; construirea acestuia constă în 3 faze de bază:

1. obţinerea diversităţii ansamblului - RF manipulează seturi de date de antrenare; este creată o listă de seturi de învăţare, folosind metoda de eşantionare treptată;
2. construirea clasificatorilor de bază - RF face disponibil arborele aleator pentru diferite seturi de antrenare. În fiecare nod, un mic grup de atribute de intrare sunt selectate în mod aleator. Dimensiunea grupului este aleasă de obicei ca fiind cea mai mare valoarea întreagă mai mică decât $\log_2 M + 1$, unde M reprezintă numărul atributelor de intrare; în continuare sunt alese cele mai bune atribute sau cel mai bun punct de ramificare; toţi aceşti arbori nu sunt retezaţi;
3. combinarea clasificatorilor de bază - RF utilizează metoda votului majoritar.

Construirea unui RF constă în generarea unor vectori aleatori pentru a creşte un ansamblu de arbori şi permiterea acestora să voteze pentru cea mai populară clasă. Pentru a evita corelarea în cadrul arborilor, algoritmul de învăţare selectează aleator un subset de caracteristici pentru a găsi cea mai bună ramificaţie la fiecare nod. În RF, arbori diferiţi au acces la diferite subcolecţii aleatoare ale setului de caracteristici sau la diverse subcolecții aleatoare de date. Dacă un vector original de caracteristici are *D* caracteristici A_i, $i = \overline{1:D}$, atunci:

- fiecare arbore foloseşte o selecţie aleatoare de $m \sim \sqrt{D}$ caracteristici de tipul $\left\{A_{i_j}\right\}_{j=1}^{m}$; spaţial caracteristicilor este diferit (dar fix) pentru fiecare arbore şi este reprezentat de către F_k, $1 \leq k \leq K = \#$ arbori (K este de ordinul sutelor);
- pentru fiecare ramificaţie la un nod al arborelui pentru o variabială dată, se alege o variabilă A_i care va fi folosită pe baza informaţiei sale de conţinut;
- valoarea lui *m* este păstrată constantă în timpul creşterii ansamblului de arbori.

Pentru a se calcula informaţia de conţinut a unui nod:

- se presupune că setul de intrare la nod este *S;* în continuare, informaţia de conţinut a nodului *N* este:

$$I(N) = |S|H(S) - |S_L|H(S_L) - |S_R|H(S_R)$$

Figura 2.14 - Informaţia de conţinut a nodului N

unde:

(1) $|S|$ este dimensiunea eșantioanelor de intrare,
(2) $|S_{L,R}|$ este dimensiunea subclaselor de la stânga (NU) şi respectiv subclaselor de la dreapta (DA),

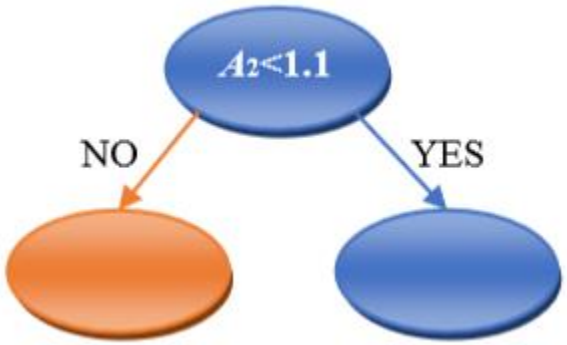


Figura 2.15 - Subclasele stânga-dreapta (NO-YES) [26]

(3) *H(S)* reprezintă entropia Shannon a $S = -\sum_{i=\pm 1} p_i \log_2 p_i$, cu $p_i = \hat{P}(\ C_i \mid S)$ fiind proporţia clasei C_i în eşantionul *S; H(S)* este numită variabilitatea în probabilităţi p_i ce formează eşantioane de intrare *S* pentru nodul current *N*,
(4) *I(N)* denotă informaţii de la nodul *N*.

- Pentru a găsi informaţia de conţinut medie $H_{av}\ (A_i\ )$ pentru A_i, pentru fiecare variabilă A_i, media peste toate nodurile N din toţi arborii care implică această variabilă este utilizată astfel:
  - în primul rând, toate variabilele A_i sunt clasificate în funcţie de informaţia de conţinut;
  - în al doilea rând, pentru fiecare valoarea fixă $n_1 < n$, RF este reconstruit folosind doar primele n_1 variabile;
- n_1 care maximizează eroarea de predicţie este selectată.

Fiecare arbore este construit cât mai mult posibil. Nu există retezări ale arborilor. Cele mai importante caracteristici ale RF sunt următoarele:

- este de necomparat în ceea ce priveşte acurateţea printre alţi algoritmi de minare a datelor;
- metoda de clasificare funcţionează eficient pe seturi de date extinse cu un număr ridicat de caracteristici;
- aceasta poate oferi o estimare privind variabialele care sunt importante în clasificare;
- generează o estimare internă fără precedent a erorii de generalizare a progresului de construire a arborilor;
- are metode de echilibrare a erorii în populaţii de clase cu seturi de date instabile.

Până în momentul de faţă, metoda de clasificare RF a fost aplicată în domenii variate, precum modelare, predicţie şi sisteme de detecție a intruziunilor.

2.5 Metoda MFCC

Descriptorii MFCC [27] sunt cei mai populari descriptori acustici de nivel scăzut (LLDs), datorită eficienţii computaţionale şi robusteţii zgomotului. Această tehnică realizează o analiză spectrală bazată pe un filtru logaritmic triunghiular plasat în frecvenţă. Aceşti coeficienţi necorelaţi furnizează o reprezentare compactă a unui semnal, bazându-ne pe informaţia lor legată de frecvenţă.

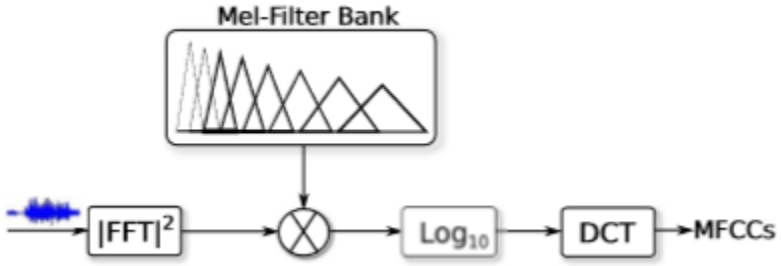


Figura 2.16 - Schema bloc a metode de detective MFCC [27]

Descriptorii MFCC s-au dovedit a fi foarte eficienţi. Calculul acestora include următoarele etape, ca în figura 2.16:

- se calculează transformată Fourier discretă, aceasta transformând semnalul în domeniul frecvență, iar spectrul de putere *P(f)* poate fi uşor obţinut;
- spectrul de putere *P(f)* este deviat de-a lungul axei sale de frecvență (în Hertzi) către axa de frecvenţă Mel *P(M)*, unde *M* reprezintă frecvenţa Mel. Pentru această operaţie se foloseşte ecuaţia de mai jos. Acest proces reflectă aproximativ percepţia audio a urechii umane.

  M(f) = 2595 logl0(1 + f/700)

- spectrul puterii rezultat este trecut prin filtrul trece bandă triunghiular, iar funcţia spectrului de putere *P(M)* devine $\theta(M)$. Acest proces de convoluţie va reduce semnificativ rezoluţia spectrală a $\theta(M)$ în comparaţie cu *P(M)*. Convoluţia discretă pune în evidenţă eşantioane din banda critică a spectrului de putere, sub forma $\theta(M_k)(k = 1, \dots, K)$ din ecuaţia de mai jos, unde Ω_k sunt spaţiate liniar pe scara Mel. Cele *K* ieşiri de forma *X(k)* = $\ln\big(\theta(M_k)\big)$ $(k = 1, \dots, K)$ sunt astfel obţinute. De asemenea, în ecuaţia de mai jos observăm o simulare a celor *K* filtre. În implementare, $\theta(M_k)$ este mai degrabă o medie decât o sumă.

$$\theta(M_k) = \sum_{M} P(M - M_k)\psi(M), \quad k = 1, \ldots, K$$

Figura 2.17 - Funcţia spectrului de putere trecută printr-un filtru trece bandă triunghiular [27]

- funcţia MFCC este calculată folosind ecuaţia 2.18, iar deseori $D \ll S$ din cauza abilităţii de compresie a funcţie MFCC.

$$\mathrm{MFCC}(d) = \sum_{k=1}^{K} X_k \cos\left[d(k - 0.5)\frac{\pi}{K}\right], \quad d = 1, \ldots, D$$

Figura 2.18 - Funcţia MFCC [27]

Capitolul 3

Metoda propusă de detecţie audio a drujbelor

3.1 Fundamente teoretice

3.1.1 Coeficienţi Haar

Coeficienţii de bază Haar [28] au fost folosiţi de-a lungul timpului în recunoaşteri versatile ale imaginilor şi ale sunetelor. Un parametru Haar este un simplu filtru diferenţial de cost computațional redus, ce necesită doar adunări şi scăderi pentru a extrage valoarea unei caracteristici. Original, această idee a fost introdusă în domeniul recunoaşterii de imagini, însă studiile au dovedit că parametrii Haar 1-D pot fi de asemenea aplicaţi în probleme de recunoaştere ale altor tipuri de semnale, precum clasificări verbale/non-verbale şi clasificări ale activităţilor umane. Prin aplicarea unui singur parametru Haar 1-D asupra unui semnal temporal obţinem un simplu filtru trece bandă, însă prin formarea unei combinaţii liniare de câţiva coeficienţi Haar, se creează un clasificator puternic, cu o capacitate ridicată de a recunoaşte cu acurateţe diverse forme sau tipuri de sunete. Acest tip de clasificator se dovedeşte versatil prin simplul fapt că putem schimba combinaţia liniară.

În studiile de recunoaşterea a formelor din cadrul semnalelor audio, forma spectrului este folosită în principal ca parametru de bază. De exemplu, semnalele vocale furnizează forme concave şi convexe aflate în benzi de frecvenţă între 400 Hz şi 4kHz, aşa cum se observă în figura 3.1:

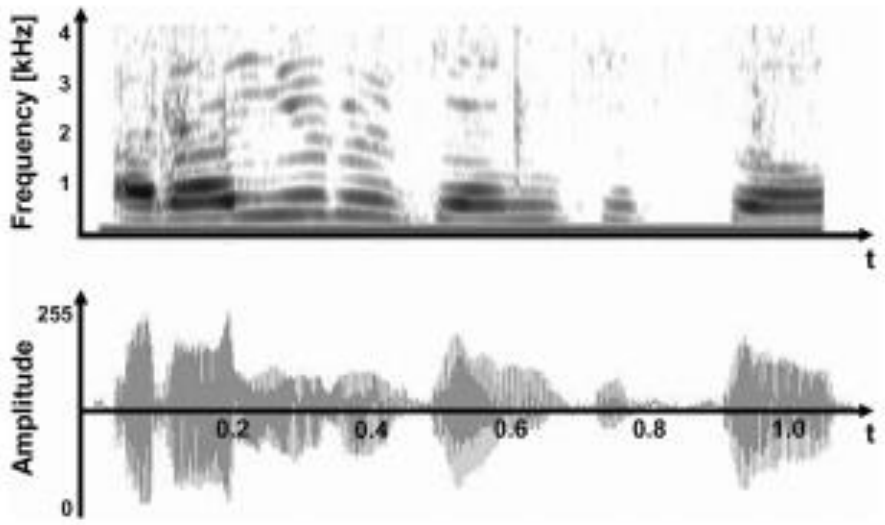


Figura 3.1 - Formă semnalului audio. Spectrul de frecvenţă al semnalului audio [28]

3.1.2 Spectograme

Spectrul [29] unui semnal pune în evidenţă conţinutul de frecvență a unei scurte secţiuni din cadrul formei de undă. Totuşi, în cazul în care dorim să observăm cum un spectru se modifică în timp în cadrul unei secţiuni de semnal audio/vocal, apare necesitatea unui alt tip de instrument spectral. Un astfel de instrument standard este spectograma, care reprezintă o diagramă bi-dimensională a frevenței în funcţie de timp, în care amplitudinea pentru fiecare frecvenţă este descrisă prin nuanţa de gri a punctului corespunzător din plan. Figura de mai jos reprezintă un exemplu de spectogramă în concordanţă cu forma de undă a cuvântului "stamp" din limba engleză (în traducere, timbru). Putem observa câteva elemente de bază: nuanţele mai închise de gri corespund cu energia de frecvenţă înaltă a fonemului "s", striaţiile verticale în fonemul "A" corespund cu faptul că această vocală caracterizează un semnal periodic, iar saltul brusc către zona albă din cadrul literei "m" corespunde cu finalizarea cuvântului prin intermediul literei "p".

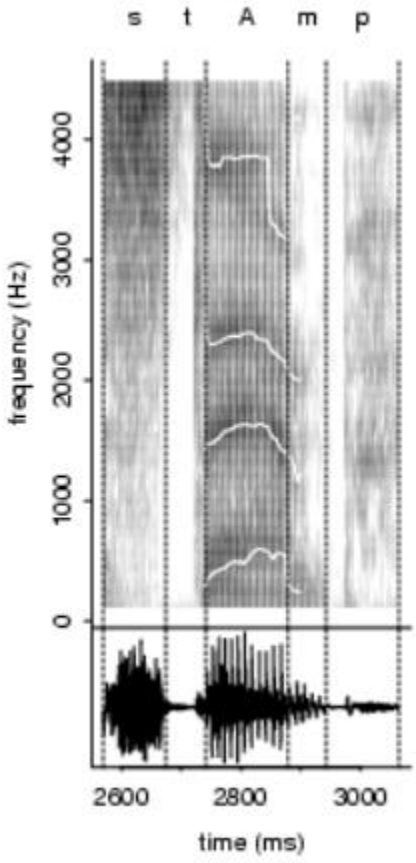


Figura 3.2 - Spectrograma cuvântului *stamp* [29]

Striaţiile verticale pentru vocala cuvântului *stamp* permit folosirea unei alte modalităţi de măsurare a frecvenţei fundamentale a semnalului în acest punct. În acest tip de spectrogramă (cunoscută sub numele de spectrogramă de bandă largă), fiecare striaţie verticală corespunde unui ciclu de deschidere-inchidere a glotei. Astfel, măsurarea timpului dintre striaţii ne indică perioada de oscilaţie, deci frecvenţa fundamentală a semnalului vocal în fiecare moment de timp.

3.2 Motivaţia metodei propuse

Această metodă îşi propună să diferenţieze sunetele produse de motofierăstrău faţă de o gamă largă de sunete cum ar fi sunetele ambientale de pădure, sunetele produse de motoare de capacitate mai mare decât cea a drujbei sau sunete produse de diferite animale. Această clasă de sunete negative va fi descrisă mai detaliat într-un capitol destinat special creării bazei de date în care vom exemplifica fiecare clasă de sunete şi în care vom specifica şi ponderea ocupată de fiecare clasă de sunet în raport cu întreaga bază de date. Pentru a realiza o clasificare cât mai bună între sunetele de drujbă şi sunetele de non-drujbă a fost necesară realizarea unui studiu exhaustiv asupra parametrilor extraşi din semnalul sonor, pentru a putea determina care dintre aceşti parametri are o capacitate mai bună de clasificare între aceste 2 clase de sunet menţionate anterior.

În urma acestui studiu aprofundat asupra parametrilor extraşi din semnalul sonor am constatat că există o diferenţă uşor de observat între sunetele produse de o drujbă care taie în mod activ şi o drujbă care funcţionează în gol. Această diferență o observăm în figura 3.3.

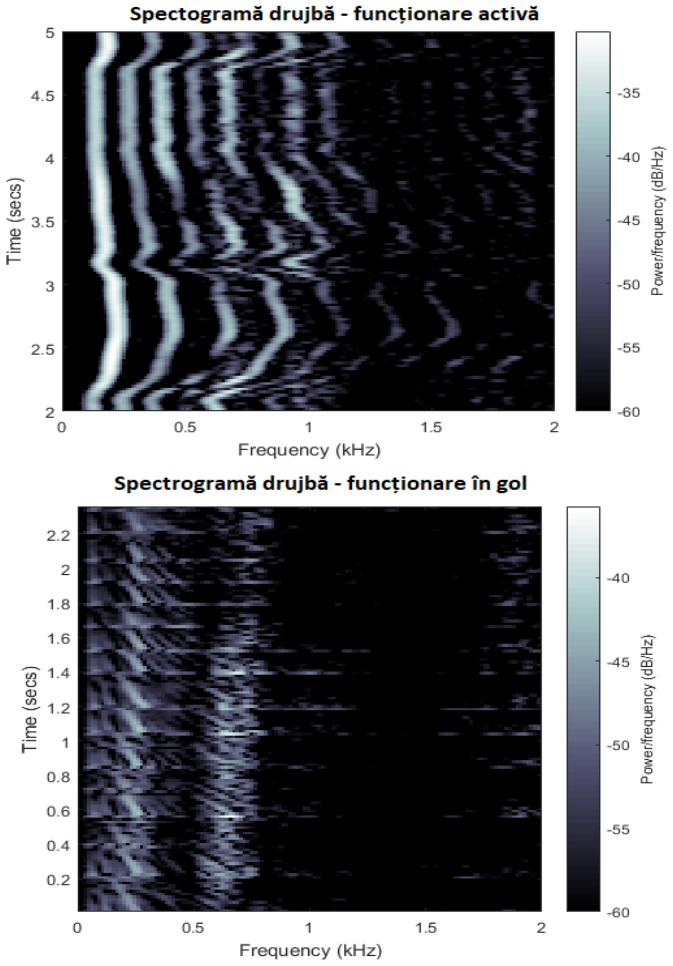


Figura 3.3 - Spectograme drujba - funcţionare active vs. funcţionare în gol

Funcţionarea activă este reprezentată de momentul în care drujba taie lemnul rezonând cu întreaga masa lemnoasă, iar funcţionarea în gol poate fi reprezentată atât de momentul în care este turat motorul de drujbă cat şi de momentul când acesta este lăsat liber, neturat.

Pentru a putea observa diferenţele dintre drujba care taie activ şi alte tipuri de sunete am analizat în continuare şi spectrograma sunetului ambiental de pădure din figura 3.4, dar şi spectrograma sunetului produs de vânt din figura 3.5.

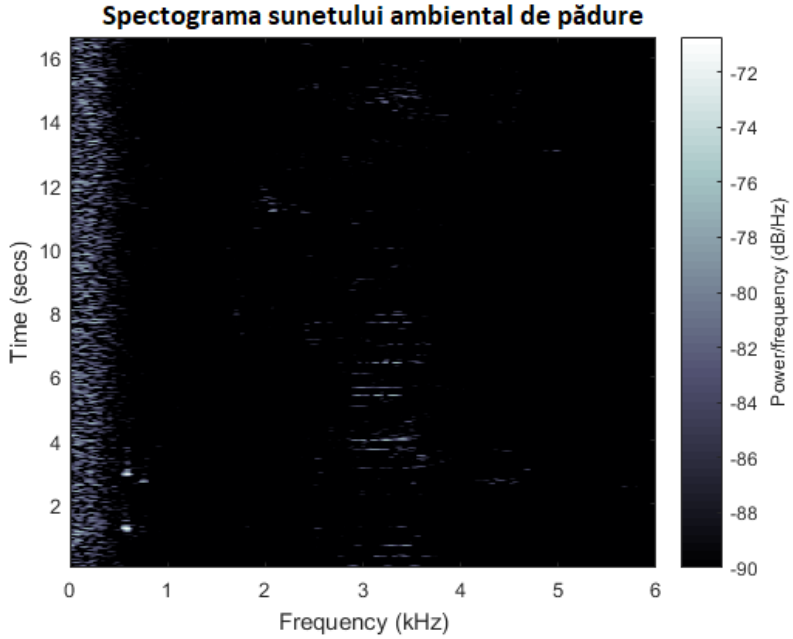


Figura 3.4 - Spectrograma sunetului ambiental de pădure

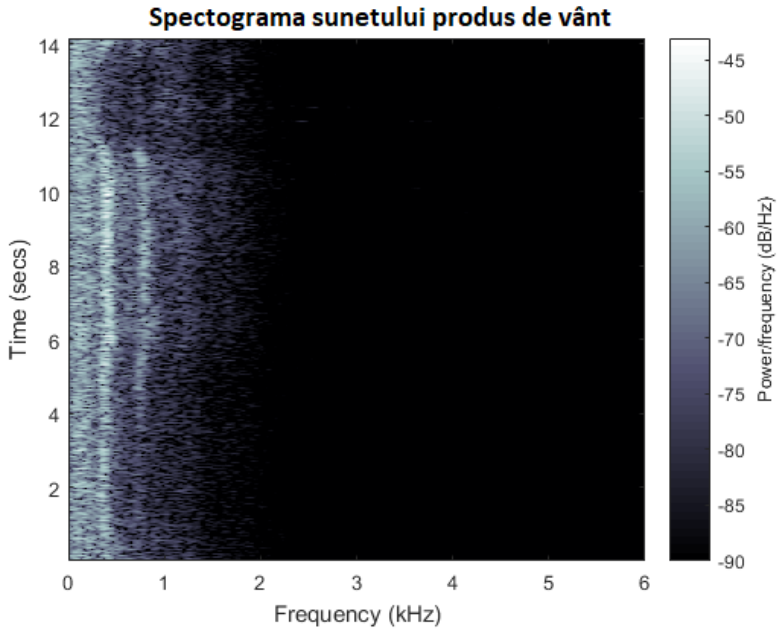


Figura 3.5 - Spectrograma sunetului produs de vânt

În urma acestei analize a multiplelor spectograme am decis că un criteriu bun de clasificare între sunetele de drujbă şi sunetele de non-drujbă ar fi reprezentat de prezenţa armonicelor specifice figurii 3.6. Prin urmare am considerat doar sunetele de drujbă care funcţionează activ ca fiind din clasa pozitivă, iar drujbele care funcţionează în gol au fost considerate ca aparţinând clasei negative, datorită faptului că nu prezintă caracteristicile spectrale potrivite detecţiei implementate.

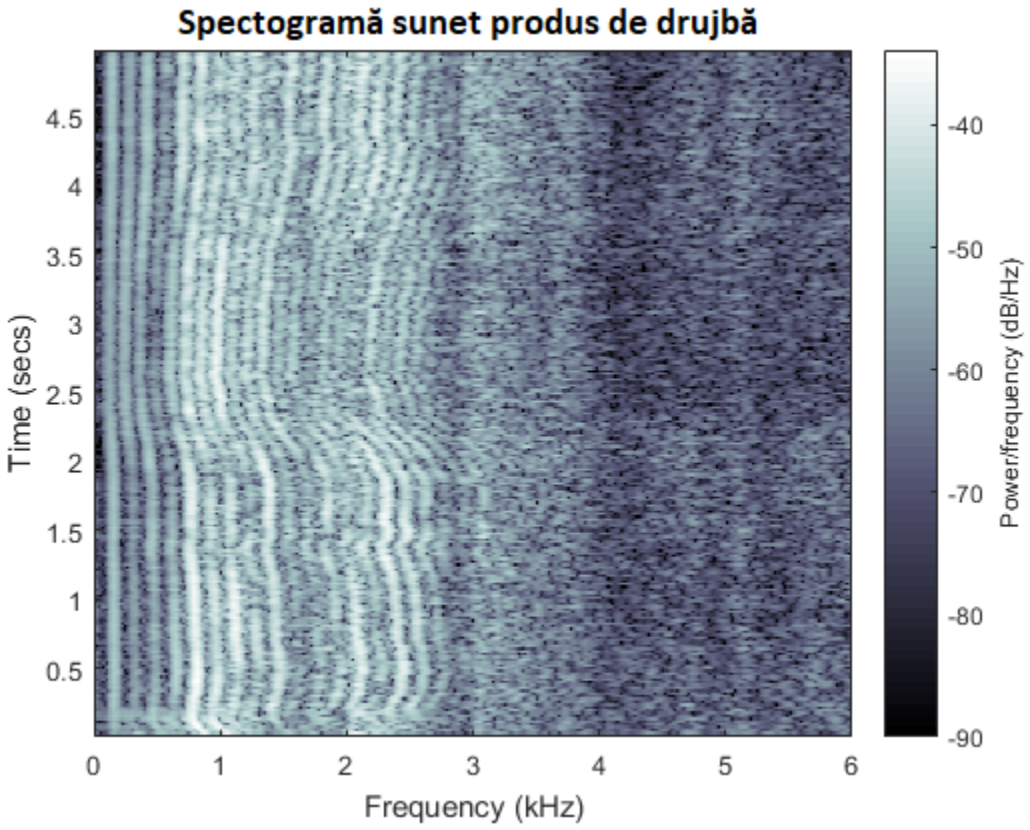


Figura 3.6 - Spectograma sunetului produs de drujbă

Pentru detectarea acestor caracteristici ale spectrogramei am utilizat un descriptor Haar capabil să genereze în urma multiplelor prelucrări un set de parametri care pot fi folosiţi cu uşurinţă în cadrul oricărui tip de clasificator. Un aspect foarte important în cadrul acestei metodei a fost alegerea unui interval de frecvenţă identic pentru toate semnalele , tocmai pentru a păstra un anumit factor de aspect al spectogramei. De asemenea, fiecare semnal a fost în ferestre de timp egale tocmai pentru a nu influenţa într-un mod negativ funcţia de extragere a coeficienţilor Haar.

3.3 Construirea bazei de date

Un pas important în proiectarea acestei metode a fost crearea unei baze de date adecvate. Setul meu de date are 2 clase principale: negative, care sunt sunete ce se găsesc în păduri şi o altă clasă care constă în sunete produse de drujbă. Pentru clasa negativă am încercat să am o gamă de sunete foarte variată. Pentru clasa de sunete considerate pozitive am mostre de la diferite tipuri de fierăstraie, la distanţe diferite faţă de dispozitivul de înregistrare. Numărul total de înregistrări este de 1397 din care 910 sunt negative şi 487 sunt pozitive.

O distribuţie mai precisă a bazei de date bazată pe sursa sunetului poate fi observată în tabelul 3.1.

Tipul de sunet	**Numărul de înregistrări**	**Distribuţia bazei de date [%]**
Drujbă	487	34.86
Câine	40	2.86
Cocoş	40	2.86
Broască	40	2.86
Sunet ambiental de pădure	60	4.29
Pisică	40	2.86
Găină	40	2.86
Insecte	40	2.86
Oaie	40	2.86
Cioara	40	2.86
Voce Umană	64	4.58
Ploaie	40	2.86
Lemne Arzând	40	2.86
Greiere	40	2.86
Cerb	15	1.07
Păsări	40	2.86
Picături de apă	40	2.86
Vânt	60	4.29
Râu ce curge	40	2.86
Tunete	40	2.86
Paşi	40	2.86
Motor tractor	41	2.93

Tabelul 3.1 - Distribuţia bazei de date

3.4 Prezentarea metodei propuse

Metoda propusă prin intermediul acestei lucrări clasifică semnale audio fie în sunete de drujbă, fie în non-sunete de drujbă, pe baza proprietăţilor spectrale ale formei de undă. Caracteristica extrasă reprezintă de fapt un coeficient de similaritate relaţionat cu un spectru ideal care constă în urme de frecvenţă distribuite uniform. Au fost obţinute rezultate foarte bune având în vedere restricţiile legate de sunete de drujbă şi adăugarea unor paşi de rafinare a sunetelor. Întregul proces de clasificare este prezentat în diagrama bloc din figura 3.7, urmând ca în continuare să prezint pe larg blocurile de bază.

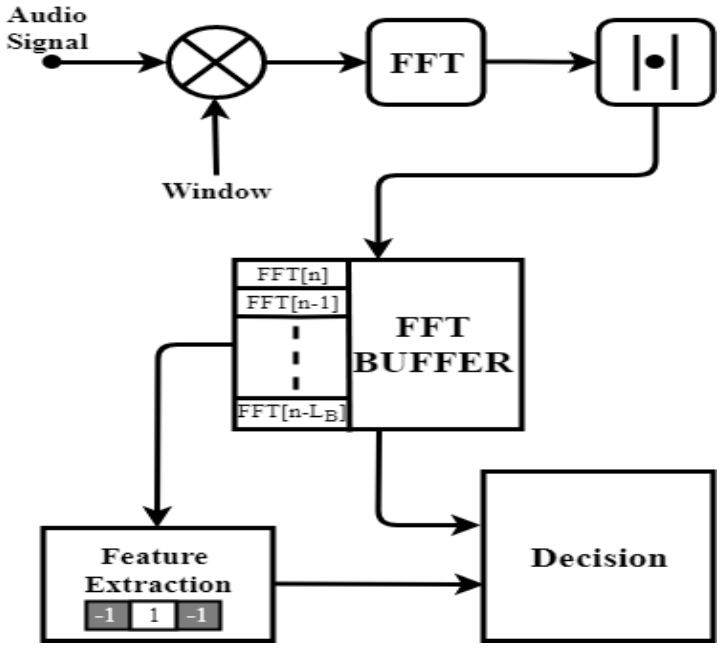


Figura 3.7 - Schema bloc a procesului de clasificare a semnalelor audio

Primul pas constă în procesarea pe ferestre a semnalului original înregistrat cu ajutorul unui microfon. În acest punct, alegerea unei lungimi inadecvate a ferestrei conduce la rezultate incorecte în cadrul blocului FFT şi la rezultate de clasificare eronate. Am decis să aleg o fereastră Hamming cu o lungime de aproximativ 30ms şi un procentaj de suprapunere de 50%.

Blocul FFT este folosit pentru a realiza transformarea Fourier a semnalului pentru fiecare fereastră în parte. În această lucrare am ales 1024 de puncte N_{FFT} pentru care am obţinut o estimare bună a spectrului. Următorul pas presupune calculul unei valori absolute FFT, iar rezultatul acestui calcul este livrat către următorul bloc unde este stocat şi plasat într-o coadă pentru procesări ulterioare.

Blocul de extragere a parametrilor Haar are rolul de a calcula o serie de coeficienţi care sunt principalele caracteristici utilizate în procedura de clasificare. Metodele care au la bază coeficienţi Haar au fost utilizate cu succes în *computer-vision* pentru recunoaşterea obiectelor [6, 7].

În această lucrare, îi folosim pentru a detecta modele specifice de frecvenţă în reprezentarea de sunete în domeniul frecvenţelor. Prin urmare, motivaţia pentru alegerea coeficienţilor Haar pentru a reprezenta spectrograma semnalelor de intrare stă în necesitatea de a detecta modelul caracteristic al armonicilor pentru o spectogramă a sunetului produs de un motofierăstrău ce taie o masă lemnoasă. După cum am observat în experimentele precizate anterior, acest model al armonicilor pentru spectograma unei drujbe are un grad înalt de stabilitate în timp.

Iniţializarea algoritmului constă în definirea măştii Haar folosită pentru determinarea coeficienţilor bazaţi pe Haar:

$$MASK = \begin{bmatrix} -1 & -1 & -1 & 1 & 1 & 1 & -1 & -1 & -1 \\ -1 & -1 & -1 & 1 & 1 & 1 & -1 & -1 & -1 \\ -1 & -1 & -1 & 1 & 1 & 1 & -1 & -1 & -1 \end{bmatrix}$$

Funcţia de extragere a caracteristicilor prezentată mai pe larg în [30] se aplică peste matricea M care conţine 3 blocuri FFT de semnale pentru ferestre consecutive:

$$H = \frac{\sum_{i=1}^{3} \sum_{j=1}^{9} [MASK * M(:, F_{in}:F_{in}+8)](i,j)}{max[MASK * M(:, F_{in}:F_{in}+8)]}$$

Prelucrarea în blocul de decizie cuprinde două etape descrise în detaliu în [30]. Prima etapă utilizează ca intrări caracteristicile Haar şi detectează vârfurile de frecvenţă. A două etapă face deosebirea între sunetele de drujbă şi alte tipuri de sunete cu proprietăţi spectrale similare.

Prima etapă implică analiza matricei caracteristice H şi determină criteriul de clasificare. Coeficientul H are o valoare mare atunci când sunt prezente vârfurile de frecvenţă specifice şi o valoare scăzută în lipsa vârfurilor. Pe baza acestei proprietăţi, am stabilit un prag cu cele mai bune rezultate de recuperare pentru setul de date de antrenament. Acest prag reprezintă criteriul de clasificare al primei etape.

A doua etapă a blocului de decizie reprezintă o procedură de rafinare a sunetelor care au un spectru similar cu cel de drujbă. Diferenţa principală dintre spectrogramele sunetelor de drujbă şi alte sunete cu modele spectrale similare cum ar fi oile, vacile, cerbii a fost că nivelul zgomotului de înaltă frecvenţă faţă de benzile inferioare de frecvenţă. Prin urmare, pentru a determina dacă un semnal aparţine clasei pozitive, am extras un alt coeficient care măsoară nivelul zgomotului de înaltă frecvenţă. În mod specific, acest coeficient este raportul dintre amplitudinea maximă pătrată şi media amplitudinilor pătratului în bandă de frecvenţă înaltă. Criteriul de clasificare pentru a doua etapă se realizează prin aplicarea unui prag peste coeficientul de zgomot. Acest al doilea prag a fost ales, de asemenea, pe baza celor mai bune rezultate de recuperare din setul de date de antrenare.

Astfel, un sunet este clasificat ca pozitiv dacă trece cu succes prin ambele etape de decizie.

3.5 Rezultate obţinute

Pentru o descriere cât mai exactă şi concisă a rezultatelor am ales diferite metrici cum ar fi acurateţea sau curba ROC. Pentru a evidenţia cât mai clar rezultatele şi perfomanţele de care este capabilă această metodă am ales să prezint în figura 3.8 acurateţea de clasificare ca fiind din clasa non-drujbă a fiecărui tip de sunet negativ din baza de date dar şi acurateţea clasificării sunetelor de drujbă ca aparţinând clasei de sunete pozitive. De asemenea, în figura 3.9 putem observa caracteristica curbei ROC pentru metoda propusă.

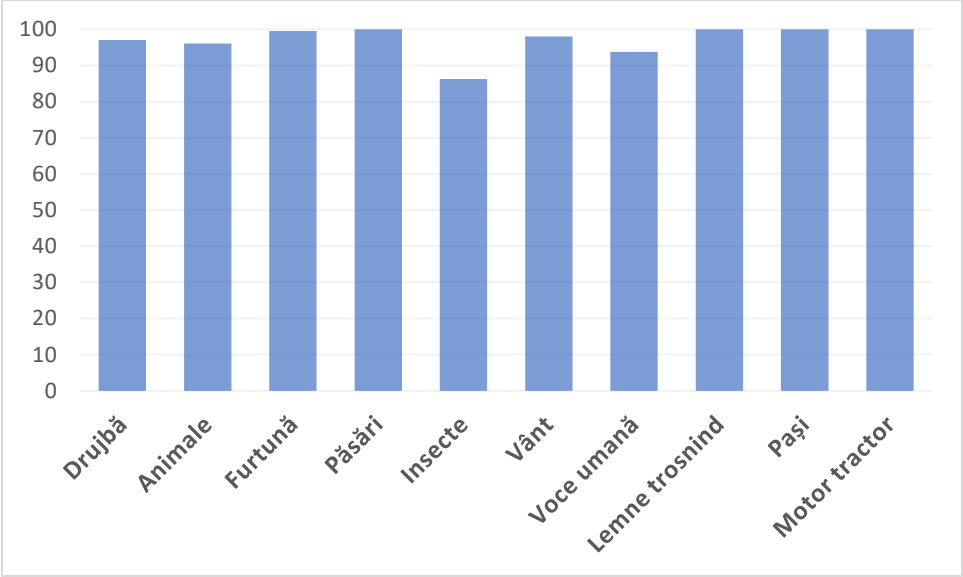


Figura 3.8 - Performanţele metodei propuse

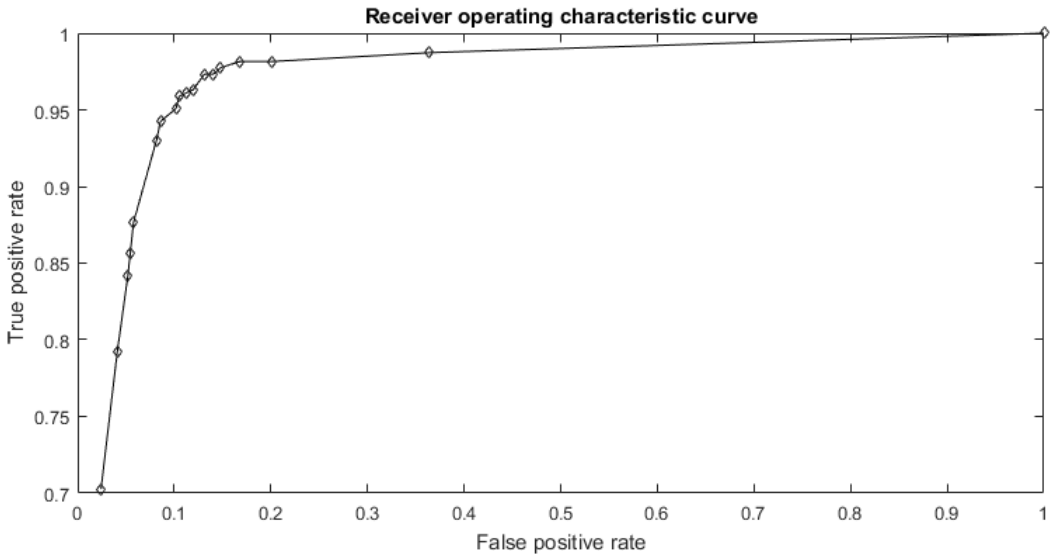


Figura 3.9 - Curba ROC a metodei propuse

3.6 Implementarea metodei propuse pe un sistem *Embedded*

Pentru implementarea şi testarea metodei propuse ce are la bază coeficienţi Haar am utilizat platforma *embedded* Raspberry, iar ca suport *software* am utilizat sistemul de operare Raspbian ce are la bază o distribuţie Debian.

În cadrul lucrării de faţă a fost implementat în limbajul de programare C++ modulul de extragere a coeficienţilor de tip Haar dar şi modulul de decizie. Acest lucru a fost realizat pentru a testa performanţa metodei propuse în cadrul unui sistem *embedded* dar şi pentru a putea demonstra faptul că procesarea *live* a unui semnal audio folosind această metodă este posibilă pe o astfel de platformă.

S-a observat că durata medie de procesare a unui minut de înregistrare durează aproximativ 40 de secunde, prin urmare se observă că prin utilizarea unui buffer de aproximativ 1 minut puteam dezvolta un sistem *live* de supraveghere a pădurilor deoarece raportul între durata necesară procesării şi durata înregistrării este de aproximativ 68%.

Codul aferent acestei implementări este ataşat în anexa 1, în cadrul funcției **get_features.c.**

Capitolul 4

Comparaţie între metoda propusă şi metodele din starea artei

În cadrul comparării celor 5 metode prezentate în paragrafele anterioare, putem folosi diverse metrici. Figura 4.1 ilustrează comparaţia unor rezultate utilizând metrici comune pentru clasificarea TPR, TNR, FPR şi FNR.

$$Overall\ Accuracy = \frac{TP+TN}{N} \times 100$$

unde TP reprezintă numărul de exemple positive ce sunt clasificate corect, TN este numărul de exemple negative clasificate corespunzător, iar N este numărul total de exemple de test. Curbele ROC le observăm în figura 4.2.

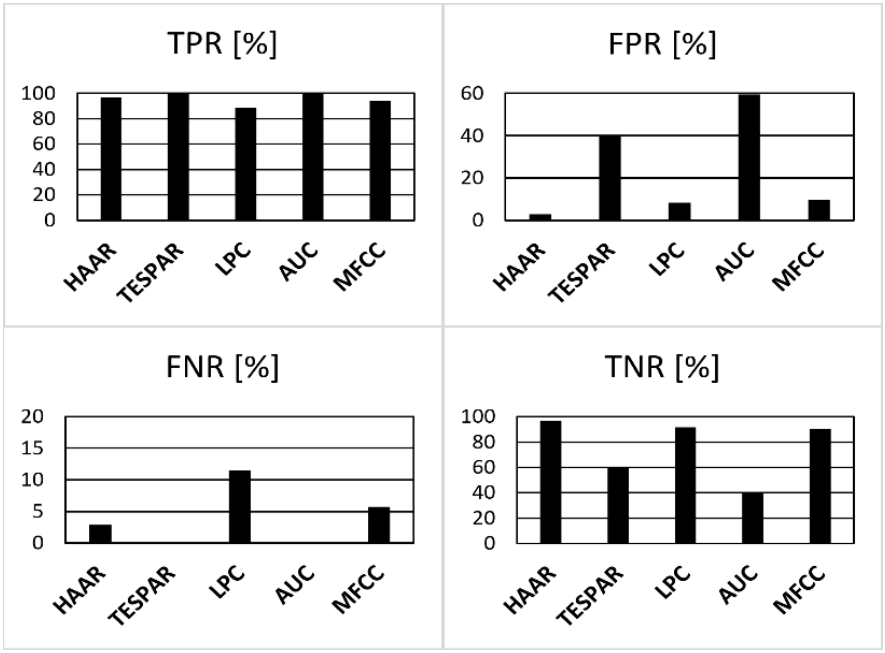


Figura 4.1 - Acurateţea generală a metodelor comparate

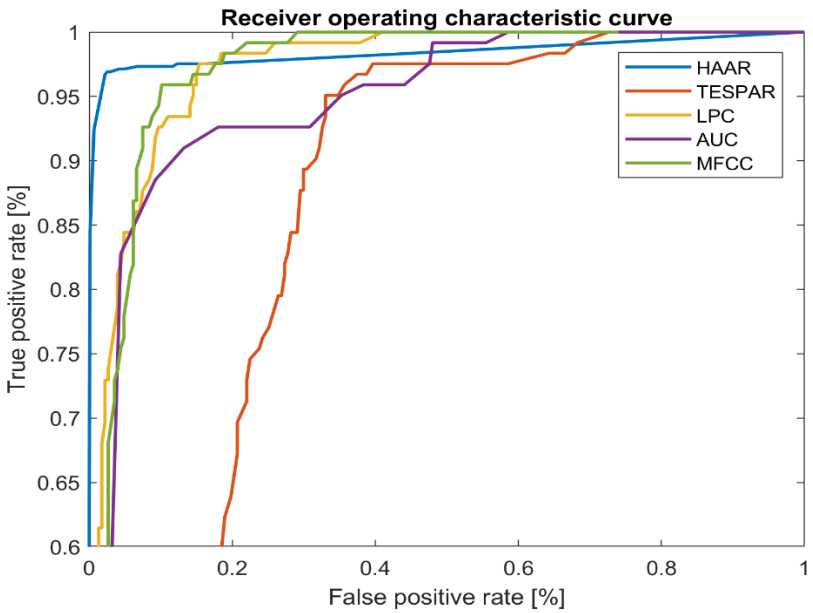


Figura 4.2 - Curbele ROC aferente metodelor comparate

Caracteristicile din domeniul timp, precum cele extrase din funcţia de autocorelaţie şi TESPAR sunt foarte sensibile la SNR (signal to noise ratio). Aceste metode au un TPR şi un FPR înalt, deci şi o acurateţe slabă. În cazul metodei TESPAR, rata înaltă a trecerilor prin 0 într-un mediu zgomotos conduce la clasificări eronate. În cazul caracteristicilor extrase din funcţia de autocorelaţie, acestea nu reuşesc să ofere o descriere clară a semnalului deoarece parametrul STI pune în evidenţă energia semnalului care nu este considerat un criteriu puternic de discriminare între diferitele tipuri de sunete.

Marele dezavantaj al utilizării coeficienţilor de predicţie liniară este faptul că oferă o reprezentare lipsită de zgomot când sunt aplicaţi pe o fereastră largă a unui semnal staţionar zgomotos, dar acest lucru vine totuşi cu un cost semnificativ, acela că se pierd informaţii discriminative la utilizarea coeficienţilor pentru semnale nestaționare în totalitate.

Parametrii domeniului spectral se dovedesc a avea o performanţă mai bună, considerând faptul că parametrii MFCC furnizează o aproximare spectrală bună. În acest caz, faptul că semnalele nu sunt staţionare de-a lungul unor perioade lungi de timp poate conduce la pierderea semnificativă a unor informaţii discriminative.

Concluzii

Metoda propusă prin intermediul acestei lucrări prezintă rezultate foarte bune, comparative cu cele ale metodelor convenţionale folosite în detecţia şi clasificarea sunetelor. Utilizarea parametrilor Haar, cunoscuţi din domeniul procesării de imagini, s-a dovedit o alegere foarte bună pentru extragerea caracteristicilor unui semnal audio, iar acest aspect se observă imediat în cadrul spectogramelor. Consider că metoda propusă în cadrul lucrării de faţă poate duce la construirea unei sistem eficient de detecţie a despăduririlor ilegale.

Performanța obținută prin metoda propusă este comparată cu alte metode din literatură în termeni de cinci valori (adică precizia generală, TPR, TNR, FPR, FNR) și curbele ROC. Toate experimentele au fost realizate pe o bază de date cu 1397 sunete împărțite în 487 exemple pozitive și 910 exemple negative. Rezultatele raportate demonstrează eficacitatea metodei propuse pentru a discrimina sunetele produse de lanț. Acest lucru se datorează în principal faptului că algoritmul propus consideră că semnalul sonor al lanțului de drujba este staționar pentru o anumită perioadă de timp.

Bibliografie

[1] Shishalov, I.S. and Gromazin, O.A. and Solovyev, Y.S. and Romanenko, A.V. and Esin, I.V., "Forest fire video monitoring system and method", US Patent 9,686,513, Jun. 20, 2017

[2] Lucian Petrică, Gheorghe Ştefan, "Energy-Efficient WSN Architecture for Illegal Deforestation Detection", International Journal of Sensors and Sensor Networks. Vol. 3, No. 3, pp. 24-30, 2015

[3] L. Todor, V. Zoicaş, L. Grama, C. Rusu, "The SPG (Signal Processing Group) Sound Database," Novice Insights în Electronics, Communications and Information Technology, Issue 15, 2014

[4] D. F. Specht, "Generation of Polynomial Discriminant Functions for Pattern Recognition," *IEEE Transactions on Electronic Computers*, vol. 16, pp. 308-319 (1967 June).

[5] F. Rosenblatt, "The Perceptron: A Probabilistic Model for Information Storage and Organization în the Brain," The Perceptron:AProbabilistic Model for Information Storage and Organization în the Brain," *Psychological Review*, vol. 65, pp. 386-408 (1958 Nov.).

[6] V. N. Vapnik, *The Nature of Statistical Learning Theory* (Springer-Verlag, 1995).

[7] G. Tzanetakis and P. Cook, "Musical Genre Classification of Audio Signals," *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 5, pp. 293-302 (2002 July).

[8] K. Fukunaga, Introduction to Statistical Pattern Recognition. Academic Press, 1990

[9] L. Devroye, L. Gy¨orfi, and G. Lugosi, A Probabilistic Theory of Pattern Recognition. Springer-Verlag, 1996.

[10] G. H. Golub and C. F. Van Loan, Matrix Computations. John Hopkins, 1989

[11] http://www.helsinki.fi/~ahonkela/dippa/node41.html

[12] http://www.saedsayad.com/support_vector_machine.htm

[13] http://mlg.eng.cam.ac.uk/zoubin/papers/ijprai.pdf

[14] R. A. King and T. C. Phipps, "Shannon, TESPAR and approximation strategies," Computers & Security, Volume 18, Issue 5, pp. 445-453, 1999.

[15] Lightweight Acoustic Detection of Logging în Wireless Sensor Networks

[16] Papán, Jozef, Matús Jurecka, and Jana Púchyová. "WSN for Forest Monitoring to Prevent Illegal Logging." în Proceedings of the Federated Conference on Computer Science and Information Systems, pp. 809-812. 2012.

[17] Freund, Y., Mason, L.: The alternating decision tree learning algorithm. Proceeding of the Sixteenth International Conference on Machine Learning, Bled, Slovenia, (1999) 124-133.

[18] Haijian Shi (2007). Best-first decision tree learning. Hamilton, NZ.

[19] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, Ian H. Witten (2009); The WEKA Data Mining Software: An Update; SIGKDD Explorations, Volume 11, Issue 1.

[20] Ross Quinlan (1993). C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, San Mateo, CĂ.

[21] Geoff Webb: Decision Tree Grafting From the All-Tests-But-One Partition. În: , San Francisco, CĂ, 1999.

[22] Leo Breiman (2001). Random Forests. Machine Learning. 45(1):5-32.

[23] Leo Breiman, Jerome H. Friedman, Richard A. Olshen, Charles J. Stone (1984). Classification and Regression Trees. Wadsworth International Group, Belmont, California.
[24] Acoustic Detection of Human Activities în Natural Environments
[25] D. Wang, and G. J. Brown, Computational Auditory Scene Analysis: Principles, Algorithms and Applications (Wiley-Blackwell, Oxford, 2006).
[26] L. Grama and C. Rusu, "Audio signal classification using Linear Predictive Coding and Random Forests," 2017 International Conference on Speech Technology and Human-Computer Dialogue (SpeD), Bucharest, pp. 1-9, 2017.
[27] J. G. Colonna, B. Gatto, E. M. D. Santos and E. F. Nakamura, "A Framework for Chainsaw Detection Using One-Class Kernel and Wireless Acoustic Sensor Networks into the Amazon Rainforest," 2016 17th IEEE International Conference on Mobile Dată Management (MDM), Porto, pp. 34-36, 2016.
[28] Nishimura D., Kuroda, T., 2010, Versatile Recognition Using Haar-Like Feature and Cascade Classifier, IEEE Sensors Journal,10, 5, 942-961.
[29] http://web.science.mq.edu.au/~cassidy/comp449/html/ch03s05.html
[30] Andrei Gaiţă, Georgian Nicolae, Anamaria Rădoi, Corneliu Burileanu, "Chainsaw sound detection based on spectral Haar coefficients", în the Proceedings of 60th Internaţional Symposium "ELMAR 2018", 16-19 September

Anexa 1

- **get_feature.c**

```
#include <string.h>
#include "get_features.h"
#include "sum.h"

double[10] get_features(double fft1[9], double fft2[9], double fft3[9])
{
  double refined_features[100];
  double haar_features[100];
  int k;
  int i;
  int j;
  int i0;
  int i1;
  double y;
  int b_k;
  double MASK[27];
  double dv0[9];
  static const signed char b_MASK[27] = { -1, -1, -1, -1, -1, -1, -1, -1, -1,
1,
    1, 1, 1, 1, 1, 1, 1, 1, -1, -1, -1, -1, -1, -1, -1, -1, -1 };

  double c_MASK[27];
  double d_MASK[27];
  double varargin_1[5];
  double e_MASK[27];
  int idx;
  boolean_T exitg1;
  memset(&refined_features[0], 0, 100U * sizeof(double));
  memset(&haar_features[0], 0, 100U * sizeof(double));

  k = 0;
  refined_features[0] = 0.0;

  for (i = 0; i < 12; i++) {
    i0 = (7 + i) << 1;
    i1 = (int)((93.0 + ((double)(i0 - i) - 7.0)) / (double)i0);
    for (j = 0; j < i1; j++) {
            for (b_k = 0; b_k < 27; b_k++) {
            MASK[b_k] = b_MASK[b_k];
            }

            sum(MASK, dv0);
            y = b_sum(dv0);
            for (b_k = 0; b_k < 27; b_k++) {
            MASK[b_k] = b_MASK[b_k];
            }

            for (b_k = 0; b_k < 27; b_k++) {
            c_MASK[b_k] = b_MASK[b_k];
            }
```

```
            for (b_k = 0; b_k < 27; b_k++) {
            d_MASK[b_k] = b_MASK[b_k];
            }

            for (b_k = 0; b_k < 27; b_k++) {
            e_MASK[b_k] = b_MASK[b_k];
            }

            varargin_1[0] = y;
            sum(MASK, fft1);
            varargin_1[1] = b_sum(fft1);
            sum(c_MASK, fft2);
            varargin_1[2] = b_sum(fft2);
            sum(d_MASK, fft3);
            varargin_1[3] = b_sum(fft3);


            idx = 0;
            b_k = 2;
            exitg1 = false;
            while ((!exitg1) && (b_k < 6)) {
                  idx = b_k;
                  exitg1 = true;
            }


      if (idx != 0) {
        y = varargin_1[idx - 1];
        while (idx + 1 < 6) {
          if (y < varargin_1[idx]) {
            y = varargin_1[idx];
          }

          idx++;
        }
      }

      haar_features[(int)floor(((double)(i + j * i0) + 7.0) / (2.0 * (7.0 +
        (double)i)))] = y;
    }

    y = haar_features[0];
    for (b_k = 0; b_k < 99; b_k++) {
      y += haar_features[b_k + 1];
    }

    refined_features[k] = y / 100.0;
    k++;
  }

  for (j = 0; j < 10; j++) {
    features[j] = refined_features[10 * j];
    for (i = 0; i < 9; i++) {
      y = features[j];
      if ((refined_features[(i + 10 * j) + 1]) > (features[j]
           || (features[j] < refined_features[(i + 10 * j) + 1]))) {
        y = refined_features[(i + 10 * j) + 1];
```

```
      }

      features[j] = y;
    }
  }
  return features;
}
```

- **sum.c**

```
#include "rt_nonfinite.h"
#include "get_features.h"
#include "sum.h"


double b_sum(const double x[9])
{
  double y;
  int k;
  y = x[0];
  for (k = 0; k < 8; k++) {
    y += x[k + 1];
  }

  return y;
}

void sum(const double x[27], double y[9])
{
  int i;
  int xpageoffset;
  int k;
  for (i = 0; i < 9; i++) {
    xpageoffset = i * 3;
    y[i] = x[xpageoffset];
    for (k = 0; k < 2; k++) {
      y[i] += x[(xpageoffset + k) + 1];
    }
  }
}
```

- **get_feature.h**

```
#ifndef GET_FEATURES_H
#define GET_FEATURES_H

#include <stddef.h>
#include <stdlib.h>
#include "get_features_types.h"

extern double[10] get_features(double fft1[9],double fft1[9],double fft1[9]);

#endif
```

- **run_haar_detection.m**

```
clear all; close all;

%% Define paths
negativePath = '../../dataset-original/negative/';
positivePath = '../../dataset-original/pozitive/';
negativeDir = dir(negativePath);
positiveDir = dir(positivePath);
decision_window = 1400;
threshold = 200;
iteration = 1;
count = 1;

%drujba remove samples at iteration = 1
%negative remove samples at iteration = 10

% for iteration = 1:2
%   for threshold = 40:2:60
    %decision_window = 160 - iteration * 40;
    %threshold = 31.5 - i/5;
    tic
    %% Positive
    structure_size = length(positiveDir)-2;
    bundle_pos(structure_size) = struct('name','','precision', 0.0);

    % configura pool for multi-threading
    myCluster = parcluster('local');
    myCluster.NumWorkers = 8;  % 'Modified' property now TRUE
    saveProfile(myCluster);    % 'local' profile now updated,
                               % 'Modified' property now FALSE
    parpool(myCluster)
    for index_bundle=1:length(positiveDir)-2
        bundle_name = positiveDir(index_bundle+2).name;
        bundle_files = dir(strcat(positivePath,bundle_name,'/*'));
        bundle_files = bundle_files(3:length(bundle_files));
        bundle_pos(index_bundle).name = bundle_name;
        counter_bundle_positive = 0;
        counter_fail = 1;
        bundle_pos(index_bundle).failed_records = [];
        parfor index_file = 1:length(bundle_files)
            full_path = strcat(positivePath,bundle_name,'/',...
                bundle_files(index_file).name);
            [y,Fs] = audioread(full_path);
            is_chainsaw = decision_tree(y,Fs,decision_window,threshold);
            if is_chainsaw
                counter_bundle_positive = counter_bundle_positive + 1;
            else
                failed_records{index_file} = bundle_files(index_file).name;
                counter_fail = counter_fail +1;
            end
        end
        counter_failed_formated = 1;
        if counter_fail ~= 1
            for j=1:length(failed_records)
                if length(failed_records{j}) ~= 0
```

```
                    failed_records_formated{counter_failed_formated} =
failed_records{j};
                    counter_failed_formated = counter_failed_formated + 1;
                end
            end
            bundle_pos(index_bundle).failed_records =
failed_records_formated;
            bundle_precision = counter_bundle_positive/length(bundle_files);
            bundle_pos(index_bundle).total_nr_files = length(bundle_files);
            bundle_pos(index_bundle).correct_classfied_nr_files =
counter_bundle_positive;
            bundle_pos(index_bundle).precision = bundle_precision;
        else
            bundle_pos(index_bundle).precision = 100;
            bundle_pos(index_bundle).total_nr_files = length(bundle_files);
            bundle_pos(index_bundle).correct_classfied_nr_files =
counter_bundle_positive;
        end
    end

    %find global mean
    corect_nr = 0;
    total_nr = 0;
    for i=1:length(bundle_pos)
        total_nr = total_nr + bundle_pos(i).total_nr_files;
        corect_nr = corect_nr + bundle_pos(i).correct_classfied_nr_files;
    end

    bundle_pos_mean(count) = corect_nr/total_nr
    delete(gcp('nocreate'))

    %% Negative

    structure_size = length(negativeDir)-2;
    bundle_neg(structure_size) =
struct('name','','failed_records',[],'precision', 0.0);

    % configura pool for multi-threading
    myCluster = parcluster('local');
    myCluster.NumWorkers = 8;  % 'Modified' property now TRUE
    saveProfile(myCluster);    % 'local' profile now updated,
                               % 'Modified' property now FALSE
    parpool(myCluster)
    for index_bundle=1:length(negativeDir)-2
        bundle_name = negativeDir(index_bundle+2).name;
        bundle_files = dir(strcat(negativePath,bundle_name,'/*'));
        bundle_files = bundle_files(3:length(bundle_files));
        bundle_neg(index_bundle).name = bundle_name;
        counter_bundle_positive = 0;
        counter_fail = 1;
        bundle_neg(index_bundle).failed_records = [];
        parfor index_file = 1:length(bundle_files)
            full_path = strcat(negativePath,bundle_name,'/',...
                bundle_files(index_file).name);
            [y,Fs] = audioread(full_path);
            is_chainsaw = decision_tree(y,Fs,decision_window,threshold);
```

```
            if is_chainsaw == false
                counter_bundle_positive = counter_bundle_positive + 1;
            else
                failed_records{index_file} = bundle_files(index_file).name;
                counter_fail = counter_fail +1;
            end
        end
        counter_failed_formated = 1;
        for j=1:length(failed_records)
            if length(failed_records{j}) ~= 0
                failed_records_formated{counter_failed_formated} =
failed_records{j};
                counter_failed_formated = counter_failed_formated + 1;
            end
        end
        bundle_neg(index_bundle).failed_records = failed_records_formated;
        bundle_precision = counter_bundle_positive/length(bundle_files);
        bundle_neg(index_bundle).total_nr_files = length(bundle_files);
        bundle_neg(index_bundle).correct_classfied_nr_files =
counter_bundle_positive;
        bundle_neg(index_bundle).precision = bundle_precision;
    end

    %find global mean
    corect_nr = 0;
    total_nr = 0;
    for i=1:length(bundle_neg)
        total_nr = total_nr + bundle_neg(i).total_nr_files;
        corect_nr = corect_nr + bundle_neg(i).correct_classfied_nr_files;
    end

    bundle_neg_mean(count) = corect_nr/total_nr
    %command for closing pool
    delete(gcp('nocreate'))
    %clear bundle_neg bundle_pos
    toc
     count = count +1;
%   end
% end

figure(1)
plot(bundle_pos_mean),title('Precision for Positive Dataset over
Iterations'),...
    xlabel('Iteration'),ylabel('Precision')
figure(2)
plot(bundle_neg_mean),title('Precision for Negative Dataset over
Iterations'),...
    xlabel('Iteration'),ylabel('Precision')
```

- **decision-tree.m**

```
function [ decision_result ] = decision_tree( y, Fs, decision_window ,
threshold)
    N_fft = 1024;
    decision_result = false;

    [P,Q] = rat(1.4733e4/Fs);
    ynew = resample(y,P,Q);


    Fs_resample = 16000;
    window_samples = 442;
    %this is made to consider only frquency smaller than Fs/2
    freq_point_limit = floor(window_samples/2);
    k=1;

    for i = window_samples/2:window_samples/2:size(ynew,1)-window_samples/2
        % ynew is windowed
        s = ynew(i-window_samples/2+1:i+window_samples/2);
        if size(s,1) == window_samples
            s=s';
        end
        % S is actually the spectrogram
        S(k,:) = (real(abs(fft(s'.*hamming(window_samples),N_fft))));
        % the if is to take in account 3 consecutive windows S(i-2:i,...)
        if k>3
           ft(i-window_samples/2+1) = get_features(S(k-
2:k,1:freq_point_limit));
        end
        k=k+1;
    end


    % decision is made here
    for i = 1:decision_window:length(ft)-decision_window
        decision_mean = mean(ft(i:i+decision_window));
        if(decision_mean>0)
            %at 5kHz window of 500Hz
            floor(2*(i+decision_window)/window_samples);
            window_5khz = S(floor(2*i/window_samples+1):
floor(2*(i+decision_window)/window_samples+1),...

(floor((5000/Fs_resample)*N_fft)):floor((5500/Fs_resample)*N_fft));
            prag = max(max(S))/mean(mean(window_5khz));
            % prag chose through observation, needs to be optimised
            if prag < threshold
                decision_result = true;
            end
            break;
        end
    end

end
```

- **get_features.m**

```
function [ features ] = get_features(  )

    S = ones(3,9);
    refined_features=zeros(10);
    haar_features=zeros(10);
    MASK = [ -1 -1 -1 1 1 1 -1 -1 -1; ...
             -1 -1 -1 1 1 1 -1 -1 -1; ...
             -1 -1 -1 1 1 1 -1 -1 -1];
    k=1;
    refined_features(1)=0;

    for i= 7 : 18
        for j = i:2*i:100-7
           haar_features(floor(j/(2*i))+1) = max ([...
               sum(sum(MASK .* S(:,j-6:j+2)))/max(max(S(:,j-6:j+2)))...
               sum(sum(MASK .* S(:,j-5:j+3)))/max(max(S(:,j-5:j+3)))...
               sum(sum(MASK .* S(:,j-4:j+4)))/max(max(S(:,j-4:j+4)))...
               sum(sum(MASK .* S(:,j-3:j+5)))/max(max(S(:,j-3:j+5)))...
               sum(sum(MASK .* S(:,j-2:j+6)))/max(max(S(:,j-2:j+6)))] );
        end
            refined_features(k) = mean(haar_features(:));
            k=k+1;
    end
    features = max(refined_features);
end
```