

UNIVERSITATEA "ALEXANDRU IOAN CUZA" DIN IAȘI
FACULTATEA DE INFORMATICĂ



LUCRARE DE LICENȚĂ

Urban Sound Recognition

propusă de

Emanuel Andrei Dodu

Sesiunea: *Februarie, 2020*

Coordonator științific

Lect. dr. Anca Ignat

UNIVERSITATEA “ALEXANDRU IOAN CUZA” DIN IAȘI

FACULTATEA DE INFORMATICĂ

Urban Sound Recognition

Emanuel Andrei Dodu

Sesiunea: *Februarie, 2020*

Coordonator științific

Lect. dr. Anca Ignat

Avizat,

Îndrumător Lucrare de Licență

Titlul, Numele și prenumele _____

Data _____ Semnătura _____

DECLARAȚIE privind originalitatea conținutului lucrării de licență

Subsemnatul(a)

domiciliul în

născut(ă) la data de, identificat prin CNP,
absolvent(a) al(a) Universității „Alexandru Ioan Cuza” din Iași, Facultatea de
..... specializarea, promoția
....., declar pe propria răspundere, cunoscând consecințele falsului în
declarații în sensul art. 326 din Noul Cod Penal și dispozițiile Legii Educației Naționale nr.
1/2011 art.143 al. 4 și 5 referitoare la plagiat, că lucrarea de licență cu titlul:

_____elaborată sub îndrumarea dl. / d-na
_____, pe care urmează să o susțină în fața
comisiei este originală, îmi aparține și îmi asum conținutul său în întregime.

De asemenea, declar că sunt de acord ca lucrarea mea de licență să fie verificată prin
orice modalitate legală pentru confirmarea originalității, consimțind inclusiv la introducerea
conținutului său într-o bază de date în acest scop.

Am luat la cunoștință despre faptul că este interzisă comercializarea de lucrări
științifice în vederea facilitării falsificării de către cumpărător a calității de autor al unei
lucrări de licență, de diploma sau de disertație și în acest sens, declar pe proprie răspundere
că lucrarea de față nu a fost copiată ci reprezintă rodul cercetării pe care am întreprins-o.

Data azi,

Semnătură student

DECLARAȚIE DE CONSIMȚĂMÂNT

Prin prezenta declar că sunt de acord ca Lucrarea de licență cu titlul „*Titlul complet al lucrării*”, codul sursă al programelor și celelalte conținuturi (grafice, multimedia, date de test etc.) care însoțesc această lucrare să fie utilizate în cadrul Facultății de Informatică.

De asemenea, sunt de acord ca Facultatea de Informatică de la Universitatea „Alexandru Ioan Cuza” din Iași, să utilizeze, modifice, reproducă și să distribuie în scopuri necomerciale programele-calculator, format executabil și sursă, realizate de mine în cadrul prezentei lucrări de licență.

Iași, *data*

Absolvent *Emanuel Andrei Dodu*

(semnătura în original)

ACORD PRIVIND PROPRIETATEA DREPTULUI DE AUTOR

Facultatea de Informatică este de acord ca drepturile de autor asupra programelor-calculator, în format executabil și sursă, să aparțină autorului prezentei lucrări, *Prenume Nume*.

Încheierea acestui acord este necesară din următoarele motive:

[Se explică de ce este necesar un acord, se descriu originile resurselor utilizate în realizarea produsului-program (personal, tehnologii, fonduri) și aportul adus de fiecare resursă.]

Iași, *data*

Decan *Adrian Iftene*

(semnătura în original)

Absolvent *Emanuel Andrei Dodu*

(semnătura în original)

Cuprins

Motivație.....	2
Introducere.....	3
Contribuție	4
Descrierea problemei.....	5
Procesarea sunetelor	6
Filtre	8
Cadre	9
Transformarea Fourier	10
Filtrul Bank	10
Coeficienții Mel	11
Filtru Banks vs MFCCs	12
Rețele neuronale convoluționale	13
Soluție.....	16
Concluzie	21
Bibliografie	22

Motivație

Recunoașterea sunetelor este tehnologia bazată pe teoriile recunoașterii formelor și pe metodele de analiză a semnalelor audio. Tehnologiile de recunoaștere a sunetelor conțin în prealabil procesarea datelor, extragerea caracteristicilor și algoritmi de clasificare. Recunoașterea sunetelor poate clasifica vectorii de caracteristici. Vectorii de caracteristici sunt creați ca urmare a procesării preliminare a datelor și a codificării liniare predictive.

Tehnologiile de recunoaștere a sunetelor sunt folosite la:

- ❖ recunoașterea sunetelor muzicale
- ❖ recunoaștere vocală
- ❖ detectarea automată a alarmelor
- ❖ în domeniul medical pentru persoane cu probleme la sistemul auditiv.
- ❖ Soluții de securitate inteligente

În soluțiile de securitate și monitorizare tehnologiile de recunoaștere a sunetelor aduc o îmbunătățire la sistemele de detecție. Aceste metode ar putea fi utile pentru detectarea intruziunilor în locuri spații publice și private. Ar putea identifica sunete de pericol sau suferință. Poate identifica sunete ca spargerea sticlei, țipete și altele. Soluțiile bazate pe tehnologii de recunoaștere a sunetelor pot oferi ajutor persoanelor mai în vârstă sau persoanelor cu probleme de auz pentru a avea o viață mai normală.

Introducere

Acest proiect are ca scop crearea unei aplicații care folosește rețeaua neuronală convoluțională pentru a antrena modele ce sunt capabile să recunoască sunete date ca input pe baza categoriilor de sunete date pentru modelele antrenate. Aplicația permite utilizatorului să antreneze propriul model pe baza sunetelor date de el.

În continuare se prezintă capitolele cât și o scurtă descriere a fiecărui capitol cu scopul de a înțelege mai bine despre proiect.

1. **Descrierea problemei.** În primul capitol se va face o scurtă descriere a problemei și modul prin care aplicația o abordează, dar se va discuta și despre ideile principale care ajută la rezolvarea problemei.
2. **Procesarea sunetelor.** În acest capitol se va face o scurtă prezentare a modului cum se prelucrează sunetul și se vor enumera filtrele care se vor aplica pe sunet.
3. **Filtre.** În acest capitol se va lua fiecare filtru folosit și se va prezenta modul cum este folosit și ce efect are asupra sunetului sunet.
4. **Rețele neuronale.** Se va explica ce înseamnă rețeaua neuronală, se va preciza tipurile de rețele neuronale și se va descrie tipul de rețea neuronală folosită în proiect.
5. **Soluție.** Se prezintă arhitectura proiectului, tehnologiile folosite și modul cum programul propriu zis este folosit de client.
6. **Concluzie.** În acest capitol se va face un scurt rezumat despre metodele folosite în proiect cât și opinia personală despre rezultatul lucrării.

Contribuție

În acest program se dorește să se implementeze, antreneze și evalueze un model care este capabil să prezică sunetele dintr-un fișier primit ca input. Există multe tehnici de depistare pe imagine dar identificarea de sunete este un subiect care de abia crește în popularitate și puține companii lucrează în domeniul respectiv. Parțial procesarea sunetelor este asemănătoare cu procesarea imaginilor doar ca tipul de date al sunetelor poate varia de la vector la matrice. Pentru ca tehnicile de recunoaștere a sunetelor să poată fi folosite de un public mai larg e necesar o bază de date amănunțită pentru procesarea sunetelor.

Scopul acestei lucrări este de a crea o aplicație care este capabilă să creeze un model cu o acuratețe ridicată ce este în stare să identifice un sunet cât mai corect și poate fi folosită de persoane ce nu au un nivel de cunoaștere ridicat în învățare automată având o interfață prietenoasă, ușor de folosit și un timp rezonabil de antrenare a modelului. Modelul poate fi mereu îmbunătățit folosind tehnologii diferite sau procesarea sunetelor într-un mod diferit dar în mare parte contează dimensiunea datelor de antrenament deoarece cu cât dimensiunea datelor crește cu atât modelul nostru are o arie de verificare mai mare și poate oferi o acuratețe mai bună.

E important de precizat că recunoașterea sunetelor nu este tot una cu clasificarea unui grup de sunete puse la un loc. Dacă aplicația primește ca input un sunet compus va arăta procentele recunoscute de model în loc să arate cât la sută din fiecare sunet apare în sunetul compus. Prin utilizarea filtrelor putem observa dacă datele noastre sunt suficient de mari sau dacă o anumită categorie are o arie mică de acoperire ce poate duce la erori mari la categoria respectivă de sunet.

Descrierea problemei

Problema noastră are ca cerință găsirea unei soluții pentru recunoașterea unui sunet dat de utilizator printr-o metodă cât mai eficientă sau mai bine zis crearea unui model cât mai rapid cu o probabilitate de identificare a sunetului cât mai mare. Pentru oameni pare ceva banal acest lucru atâta timp cât persoana respectivă știe acel sunet dar dacă avem un sunet compus din care dorim să recunoaștem toate sunetele concrete(ex: într-un oraș vrem sa identificam sunetele de claxoane, animale, motoare, oameni) poate fi dificil pentru oameni sau sistemele de securitate au nevoie de o recunoaștere a sunetelor bune pentru a nu se declanșa când nu e necesar. Totuși pentru un computer e mai dificil de analizat un sunet pentru că el identifică sunetul respectiv ca o lista de trăsături unde trăsătura reprezintă o tonalitate si trebuie trecuta printr-o baza de date pentru a fi identificată. Pe scurt problema noastră cere o modalitate eficientă si rapidă de a identifica un sunet dat ca parametru in clasele din baza de date folosită pentru antrenarea metodei de identificare.

O metodă ar fi crearea unui model folosind o rețea neuronală convoluțională si a frecvenței Mel pentru procesarea sunetului și oferirea soluției. În continuare se va explica cum se modifică sunetul si cum se creează mediul pentru analiza sunetului si oferirea răspunsului pe care îl dorim.

Procesarea sunetelor

Începutul oricărui sistem de recunoaștere a sunetului este de a extrage trăsături adică trebuie să identificăm componentele din semnalul audio ce sunt utile pentru identificarea conținutului unic și înlăturarea informațiilor inutile, cum ar fi sunetele de fundal sau cu frecvență joasă care se aud la fel indiferent de cât de diferite sunt sunetele la frecvența mai înaltă. Scopul principal în a înțelege despre sunete este că sunetele generate sunt filtrate o anumită formă, la oameni sunetele sunt filtrate de tractul vocal incluzând gatul și dinții, aceste forme determină ce sunete apar.

Dacă putem determina cum sunt create formele putem crea o reprezentare exactă a fenomenului produs. Putem reprezenta sunetele ca un spectru de putere și treaba MFCCs(Coeficienți Cepstrali de Frecvență Mel) este de a oferi o reprezentare cât mai exactă. De fapt MFCCs-ul este cel mai utilizat în recunoașterea sunetelor și a fost introdus în 1980 de Davis și de Mermelstein. În afară de MFCCs mai sunt LPCs(Coeficientul de predicție linear) și LPCCs(Coeficienții Cepstrali de predicție linear) dar în acest proiect vom vorbi de MFCCs și de ce este bun pentru recunoașterea automată a sunetelor și cum trebuie implementată. Acum vom oferi niște pași urmând să explicăm în detaliu fiecare pas. Pentru început trebuie să încadrăm semnalul sunetului în cadre mai scurte, apoi pentru fiecare cadru trebuie calculată densitatea spectrală a spectrului de putere, aplicăm filtrul mel pe spectrul de putere și adunăm energiile fiecărui filtru, luăm algoritmul fiecărui filtru de energie, luăm DCT-ul(Transformarea cosinică discretă) algoritmilor din filtre și în final păstrăm coeficienții de care avem nevoie și eliminăm ce rămâne.

E important să înțelegem de ce avem nevoie de pașii enumerați mai sus. Vom începe să explicăm cum salvăm semnalul audio care este în schimbare chiar și pe perioade foarte scurte, avem nevoie să găsim o metodă pentru a salva datele. Ca soluție vom folosi un cadru semnalului audio și îl vom împărți în cadre mai mici de 20-30 milisecunde dar, trebuie să fim atenți pentru că dacă folosim cadre foarte scurte nu avem suficiente probe pentru a forma un spectru de date valid și nu putem lua nici o probă prea lungă pentru că semnalul s-ar schimba prea mult într-un cadru. E important să precizăm că trebuie să simulăm ce face corpul uman, de exemplu pasul 2 este inspirat de coclea umană(un organ al urechii) care vibrează la anumite intensități ținând cont de

sunetul care vine în contact cu acel organ. Periodograma noastră simulează același lucru, diferența este că coclea umană nu poate să diferențieze frecvențe apropiate și acest lucru se intensifică dacă frecvența este mai înaltă. Din această cauză trebuie să luăm periodogramele tuturor cadrelor să le însumăm și să vedem energia formată în regiuni de frecvență.

După ce am făcut toți 3 pași trebuie să le luăm logaritmul lor. Și acest lucru este copiat după corpul uman. Noi nu auzim intensitatea liniară și din această cauză să percepem un volum al unui sunet avem nevoie să depunem de 8 ori mai multă energie, deci o variație mare în energie poate să nu sune chiar așa de diferit dacă, de exemplu sunetul are volumul ridicat. Frecvența Mel este folosită pentru procesul de comprimare a energiei pentru a face procesul nostru mult mai asemănător auzului uman. Algoritmul ne ajută să folosim extragerea mediei spectrale, mai exact folosim o tehnică de normalizare.

Ultimul pas este să folosim DCT energiilor filterbank. Avem două motive pentru a folosi DCT. Primul motiv ar fi că energiile noastre filterbank sunt toate suprapuse, ele sunt suprapuse una cu alta. DCT-ul de corelează energiile, adică matricele covariate diagonale pot fi folosite în clasificatorul HMM. Coeficienții DCT reprezintă schimbări rapide în energiile filterbank și aceste schimbări rapide degradează performanța ASR-ului și prin urmare dacă nu luăm toți coeficienții obținem o mică îmbunătățire a ASR-ului.

Filtre

Vom începe cu fișierul audio și cum este reprezentat de digital. Semnalul este împărțit în timp și amplitudine. Timpul va fi măsurat în secunde iar amplitudinea în hertzi.

Mai jos este reprezentată o imagine cu sunetul respectiv:

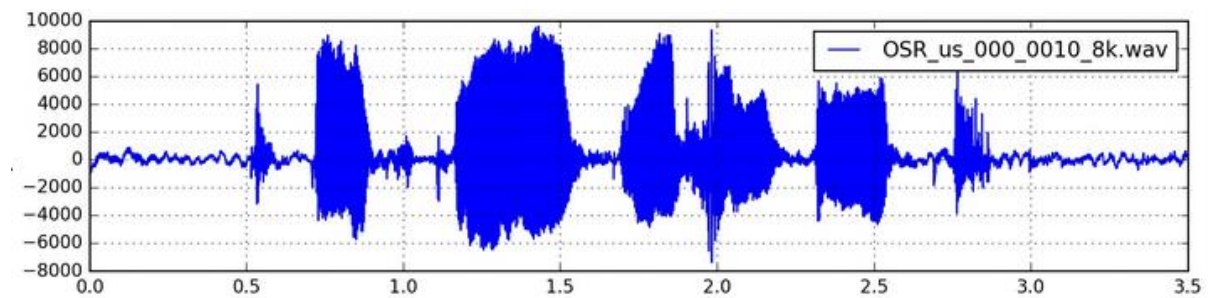


Fig.1

Pe acest tip de sunete vom aplica filtre și vom ajunge la datele care vor fi luate de rețeaua neuronală.

Pe acest sunet vom aplica o pre-accentuare care va avea 3 obiective:

- Balansarea spectrului de frecvență deoarece frecvențele mai înalte au o magnitudine mai mică în comparație cu cele mici
- Evitarea problemelor de aproximare când aplicăm transformarea Fourier
- Îmbunătățirea SNR-ului

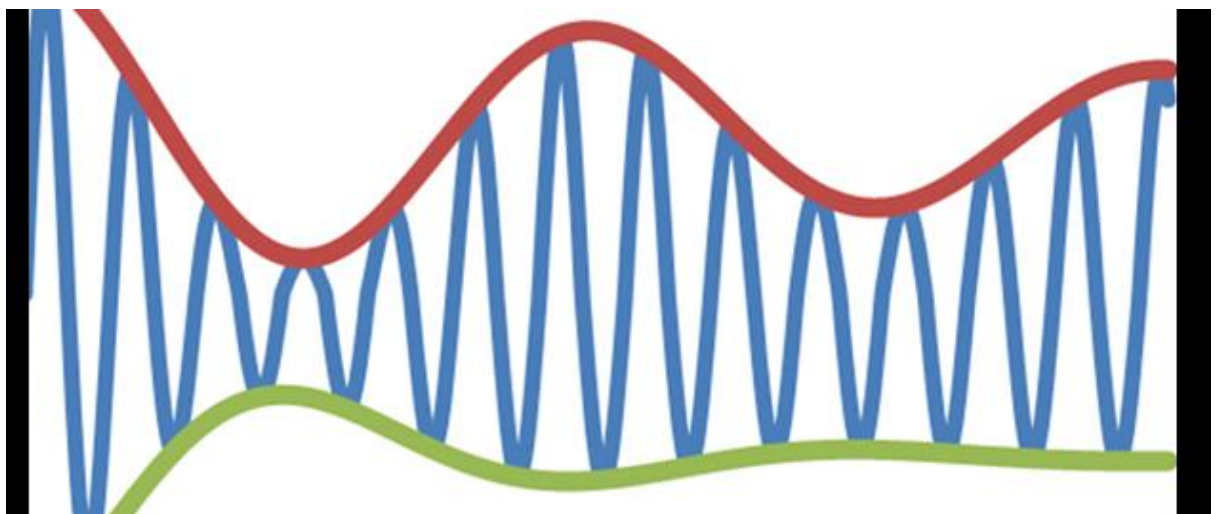


Fig.2

Practic noi eliminam frecvențele joase de o anumita valoare si vom folosi formula:

$$y(t)=x(t)-\alpha x(t-1)$$

Aceasta formulă va face normalizarea medie unde, x este semnalul iar, α este coeficientul aplicat pe semnal.

După aplicarea formulei pe semnal vom avea următoarea imagine:

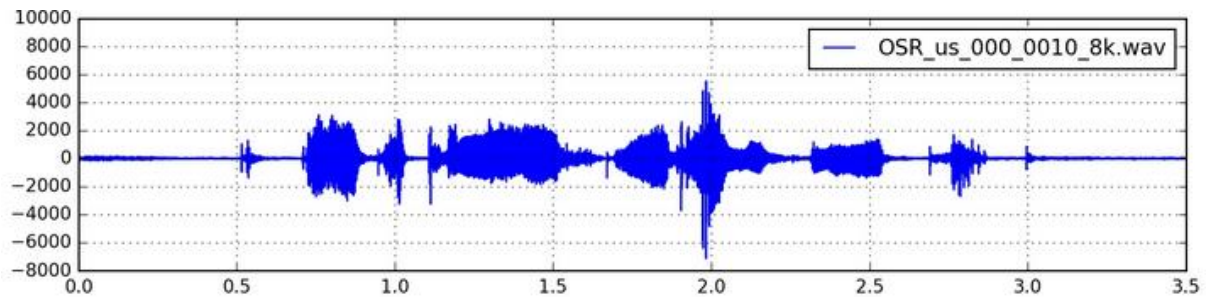


Fig.3

Cadre

După pre accentuare vom face o încadrare, adică vom împărți semnalul în cadre mai mici. Logica acestui lucru este faptul că frecvențele într-un semnal se schimbă rapid si nu se merită sa facem transformare Fourier pe tot sunetul pentru ca pierdem date importante deci, făcând transformarea Fourier pe cadre mai mici putem obține o aproximare bună a frecvenței prin concatenarea cadrelor adiacente.

După împărțirea semnalului in cadre aplicăm o funcție numită Hamming fiecărui cadru. Aplicăm aceasta funcție pentru a contraataca faptul ca transformarea Fourier are asumția ca informația e infinită și reducem pierderea spectrală.

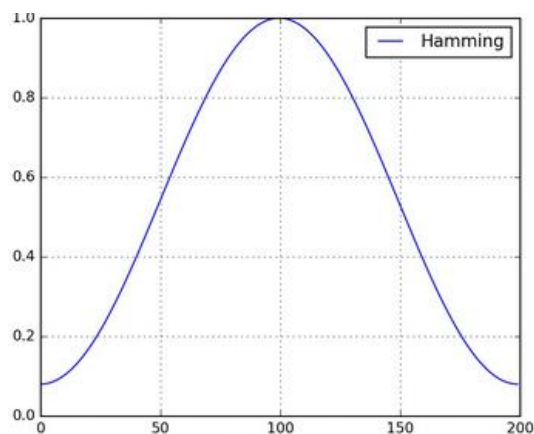


Fig.4

Transformarea Fourier

După normalizare putem aplica transformarea Fourier pe fiecare cadru pentru a calcula frecvența spectrului pe N-puncte unde de obicei N e 256 sau 512, se mai numește și transformare Fourier scurtă și vom folosi ecuația:

$$P = \frac{|FFT(x_i)|^2}{N}$$

Unde x_i este al i-lea cadru a semnalului x.

Filtrul Bank

Pasul final în aplicarea filtrului Bank este să aplicăm filtre triangulare, mai exact 40 de filtre, pe o scală Mel a spectrului de putere pentru a extrage bandele de putere. Mel încearcă să elimine percepția non-liniară a urechii umane în perceperea sunetului fiind mult mai discriminator asupra frecvențelor joase și mai puțin discriminator asupra frecvențelor mai mari. Mai exact convertim între Hertz(f) și Mel(m) cu formulele următoare:

$$m = 2595 \log_{10} \left(1 + \frac{f}{700} \right)$$

$$f = 700 \left(10^{\frac{m}{2595}} - 1 \right)$$

Fiecare filtru din filtrul Banks e triunghiular având un răspuns crescând la 1 în centrul frecvenței și scăzând la 0 liniar până ajunge la centrul frecvenței sau la 2 filtre adiacente unde răspunsul e 0 ca în figura următoare:

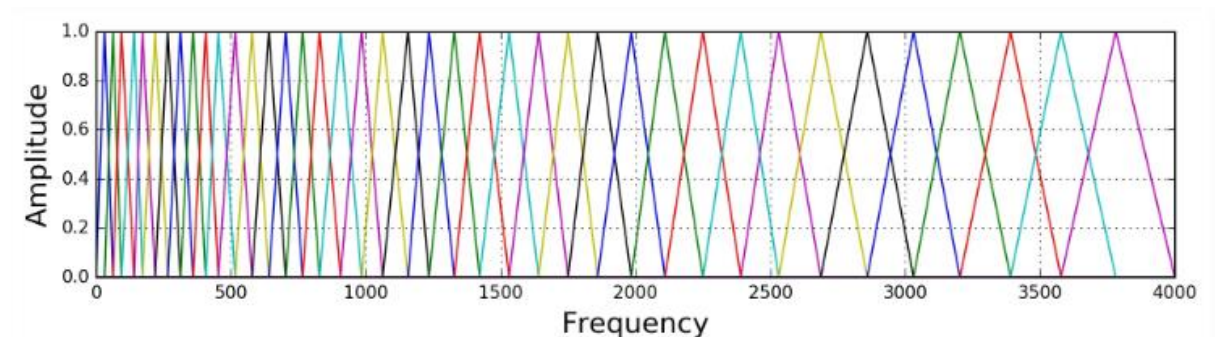


Fig.5

După ce aplicăm filtrul Banks pe spectru de putere a semnalului, obținem următoarea spectrogramă:

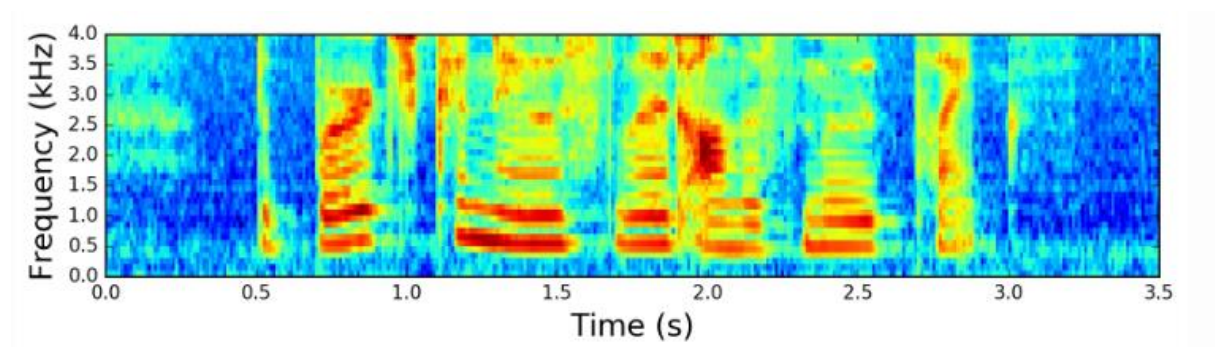


Fig.6

Coeficienții Mel

Putem vedea că filtrele Banks folosite în pasul anterior sunt foarte corelate, acest lucru poate fi problematic în anumiți algoritmi de machine learning. Prin urmare, trebuie să aplicăm Transformarea Discretă Cosinusoidală (DCT) pentru a decorela coeficienții filtrului Banks. Tipic pentru ASR, rezultatul coeficienților spectrali 2-13 sunt păstrate și restul este înlăturat. Motivul pentru înlăturare este faptul că restul coeficienților reprezintă schimbări rapide în filtrul Banks și aceste detalii fine nu contribuie la recunoașterea automată a sunetelor.

Aplicam si o liftare sinusoidala MFCCs pentru a sublinia MFCCs-ul mai mare ceea ce ajuta la recunoasterea sunetului in semnale mai ridicate.

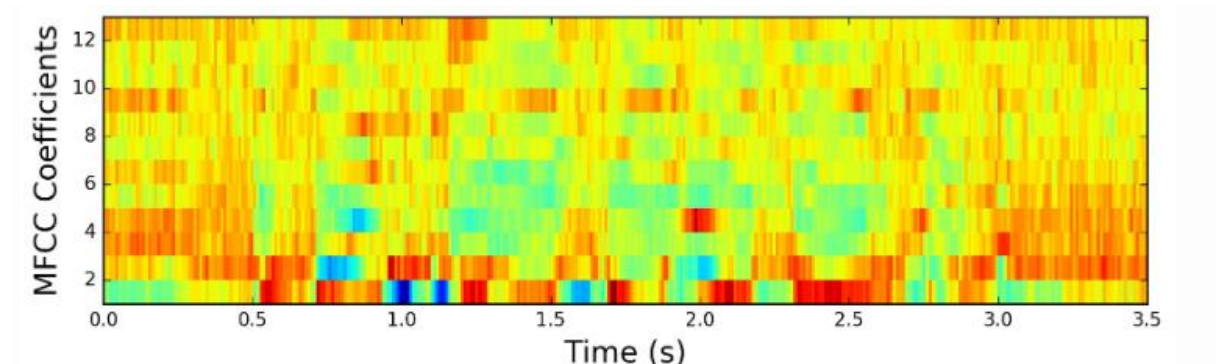


Fig.7

Pentru a balansa spectrograma si îmbunătăți Semnalul-la-Zgomot(SNR) putem simplu extrage coeficientul mediu din toate cadrele.

Normalizarea Medie a MFCCs-ului:

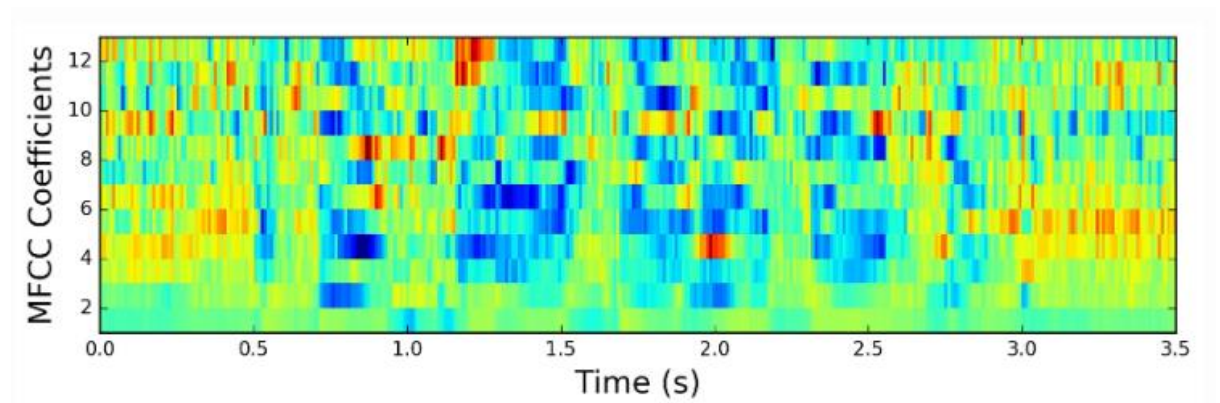


Fig.8

Filtru Banks vs MFCCs

Se va motiva implementarea Filtrului Banks si a MFCCs-ului. E interesant de observat ca toți pașii folosiți pentru a aplica filtrul Banks sunt inspirați recunoașterea sunetelor de către natura umana si de perceperea de sunete de către urechea umana. In contrariu pașii extra folosiți pentru MFCCs au fost motivați de către limitările unor

anumiți algoritmi de machine learning. DCT(Transformarea discreta cosinusoidală) e necesară pentru a decorela coeficienții filtrului Banks un proces care se mai numește și albire. În particular, MFCCs e foarte popular cu modelele de amestec gaussiane – modele ascunse Markov(GMMs-HMMs) sunt foarte populare împreună. În momentul respectiv MFCCs și GMMs-HMMs co-evoluiează și sunt căile standard de a face recunoașterea automată a sunetelor(ASR).

Cu avansarea învățării profunde în sistemul de sunet se încearcă să se descopere dacă MFCCs încă este calea corectă ținând cont de faptul că rețelele neuronale sunt mai puțin susceptibile la marile corelate input-uri și prin urmare DCT-ul e o transformare liniară, și de aceea e nedorită pentru că elimină niște informații care nu sunt liniare.

Este delicat să întrebăm dacă e necesar să folosim transformarea Fourier. Ținem cont de faptul că Fourier e o operație liniară. Poate fi benefic să ignorăm și să vedem ce se întâmplă dacă învățăm direct din semnal. Transformarea Fourier este o transformare dificilă și crește cantitatea de date și complexitatea modelului necesar atingerii aceluiași performanțe. În plus, folosind Transformarea Fourier scurta ne asigurăm de faptul că semnalul este staționar în intervale mici de timp și prin urmare liniaritatea transformării Fourier nu va impune o problemă critică.

Rețele neuronale convoluționale

Rețeaua neurală convoluțională este o clasă de mai multe rețele aplicate în mare parte pentru imagini dar dacă prelucrăm informațiile pe care le luăm și cream o reprezentare grafică a acestor informații putem folosi rețelele neuronale și pe sunete. Numele sugerează rețeaua neurală implică o operație matematică numită convoluție. Convoluția este o operație liniară specializată. Rețelele neuronale convoluționale sunt rețele simple care folosesc convoluția în locul înmulțirii matricei generale în cel puțin unul dintre straturi.

Ca aspect rețeaua neurală convoluțională este alcătuită dintr-un strat de input și un strat de output, dar și de straturi ascunse. Convoluția e o operație de înmulțire pe filtre(numite și kernel) cu o matrice de imagine pentru a extrage din el niște caracteristici determinate. Mai concret noi folosim filtrul convoluțional să filtrăm

imaginea si sa arătam numai ce contează. E de precizat ca kernel-ul rămâne pe matricea inițială fără a se supraîncărca.

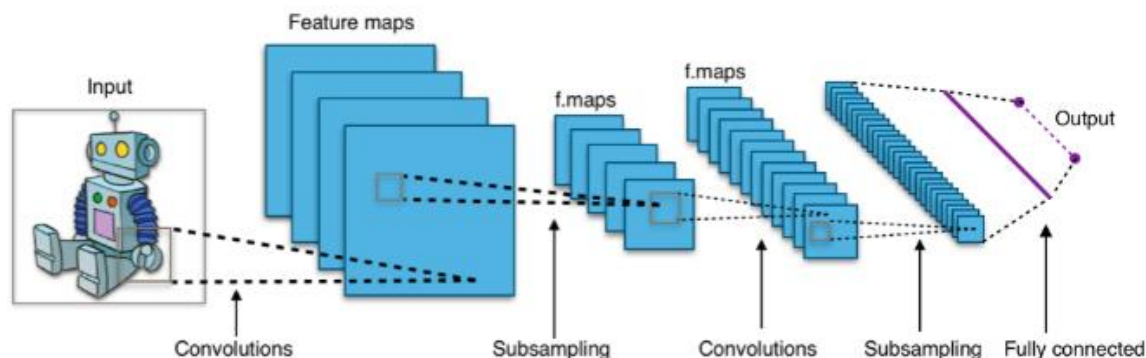


Fig.9

Straturile convoluționale aplica filtre sau kernel-uri pe imaginea data ca input, obținem o hartă cu caracteristici a imaginii respective. Fiecare filtru are un task simplu si precis de îndeplinit. Deci pentru a rezolva problemele noastre de clasificare între sunete trebuie să folosim mai multe filtre care ne vor ajuta să atingem acest lucru. Prin combinarea caracteristicilor evidențiate de filtre; cum ar valoarea frecvențelor, modelul nostru va fi în stare să antreneze modele care vor diferenția clasele de sunete diferite și cunoscute. Trebuie să alegem un număr de convoluții de făcut, pentru că numărul de filtre folosit, știind că mai multe filtre avem mai multe caracteristici extragem pentru clasificator, parametrii vor fi mult mai numeroși pentru model să învețe dar performanța va fi mărită, adică acuratețea mărită. După acest lucru trebuie să decidem umplerea sau nu și ce funcție de activare să folosim.

Într-o imagine, este o puternică corelare între pixeli. Mai exact dacă un pixel în imagine e roșu este foarte probabil ca și patru cei mai apropiați pixeli să fie tot roșii. Deci putem să ne folosim de acest lucru să reducem dimensiunea imaginii prin păstrarea unei reprezentări locale per bloc local, acest lucru se cheamă umplere. Deci vom lua pixel-ul cu intensitatea maximă în bloc(umplerea maximă) și mediem intensitatea pixelilor în bloc(umplere medie) și o păstrăm ca o reprezentare a acelei locații.

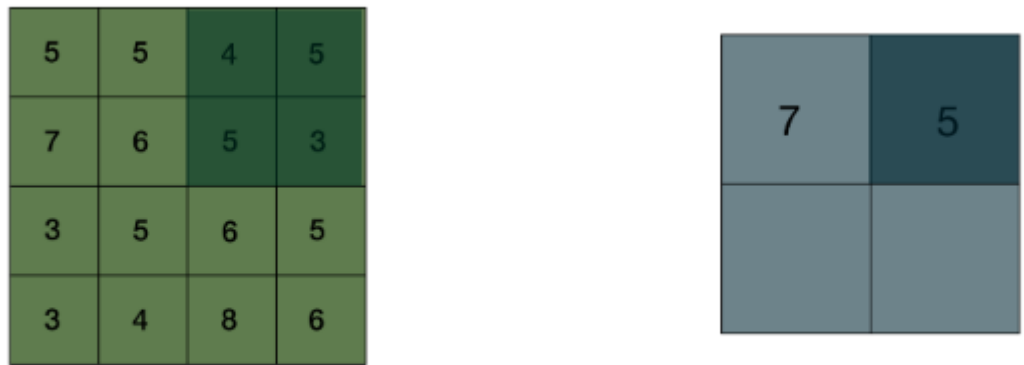


Fig.10

După cum putem observa, reunim jumătăți pentru fiecare dimensiune. Făcând această operație degradăm puternic imaginea inițială reprezentând blocul de pixeli doar cu unul, imaginea de output este mai puțin clară, dar conține caracteristicile principale ale imaginii originale.

Soluție

La acest capitol se va prezenta aplicație si modul cum aplicația va manipula input-ul după informațiile de la capitolele anterioare. Trebuie sa precizam datele necesare pentru funcționarea alocăției noastre. La început avem nevoie sa creăm modelul folosind rețeaua neuronală ca sa putem recunoaște un sunet, deci avem nevoie de date de antrenament, cu cat mai multe cu atât mai precis va deveni modelul nostru. Pe lângă sunete avem nevoie si de un Excel care va conține numele fișierelor si clasa lor de sunete.



Fig.11

După acest lucru trebuie sa ne asiguram ca sunetele sunt in formatul corect, mai exact aplicația are nevoie de sunete de 16 biți adâncime si folosind opțiunea Curățare va lua fiecare fișier din folderul train si îl va converti si muta in folder-ul clean. Pe noile sunete se poate aplica obține Neural care va lua sunetele, va aplica filtrele si după se va antrena modelul, urmând ca apoi sa selectam un sunet sa îl ascultam(opțional) folosind Browser si cu Play vedem rezultatul dorit. Prin rezultatul dorit înțelegem categoria de sunete din care face parte sunetul dat ca input si cat la suta in ce clasa a fost încadrat sunetul respectiv. Avem un exemplu mai jos.

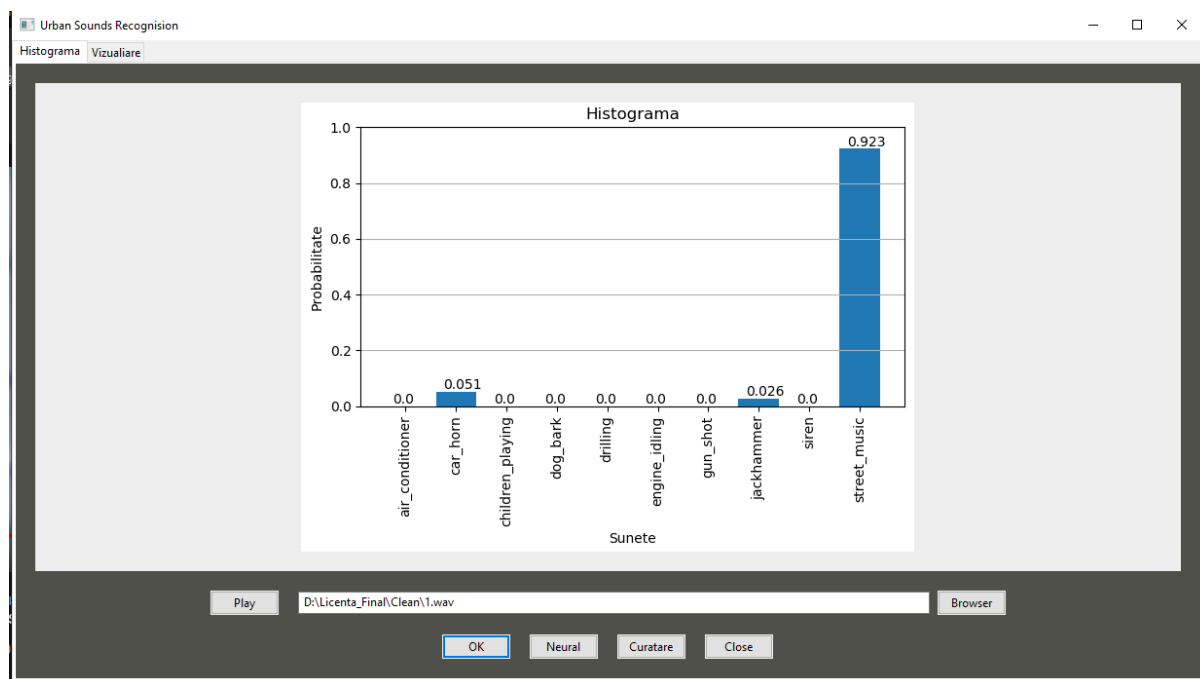


Fig.12

In exemplul de mai sus vedem ca sunetul nostru a fost încadrat in categoria muzica de strada si ca sa verificam daca acest lucru este adevărat cu opțiunea Play prin care sunetul va fi redat si auzit de noi.

La tag-ul Vizualizare putem vedea diagrama sunetelor folosite pentru modelul antrenat de rețeaua noastră si mai putem observa si filtrele aplicate pe sunete pentru a decide daca mai trebuie sau nu sa adăugam sunete pentru anumite clase.

In imaginea de mai jos se afișează diagrama cu procente pentru fiecare clasa folosita in model:

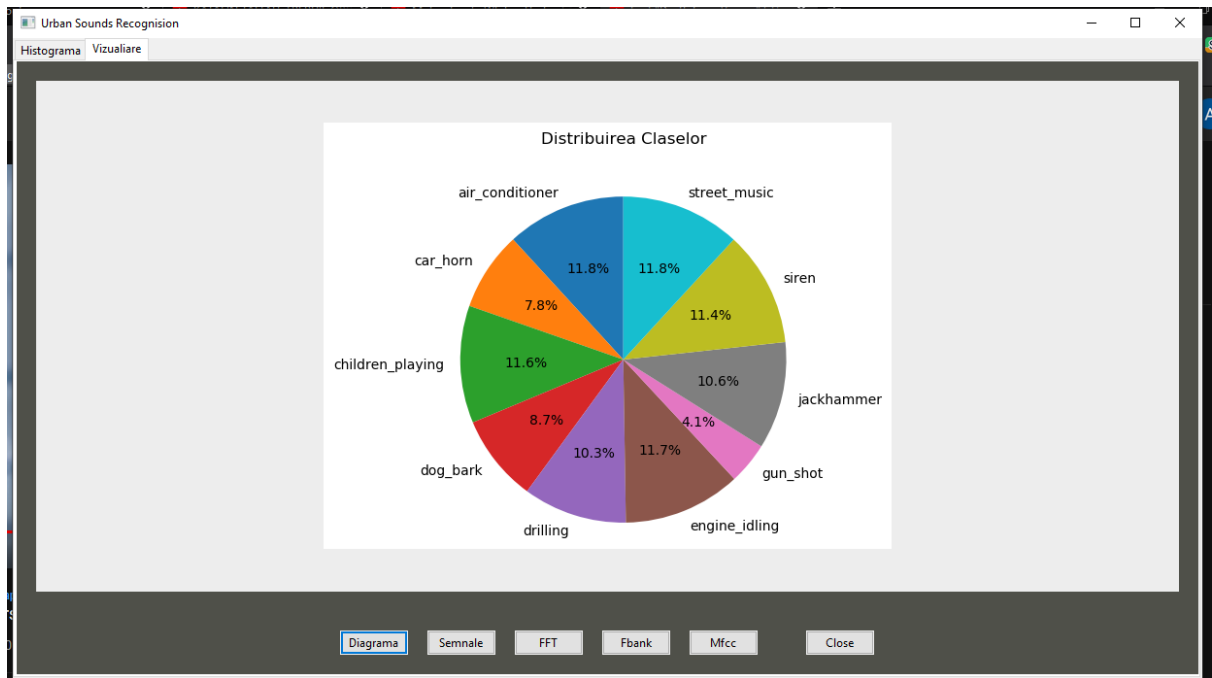


Fig.13

Avem clasele si cat la suta din sunetele folosite pentru rețea aparțin fiecărei clase(Distribuirea Claselor).Putem apela aceasta diagrama folosind opțiunea Diagrama.

La opțiunea Semnale se va afișa semnalele reprezentate grafic pentru fiecare clasa.

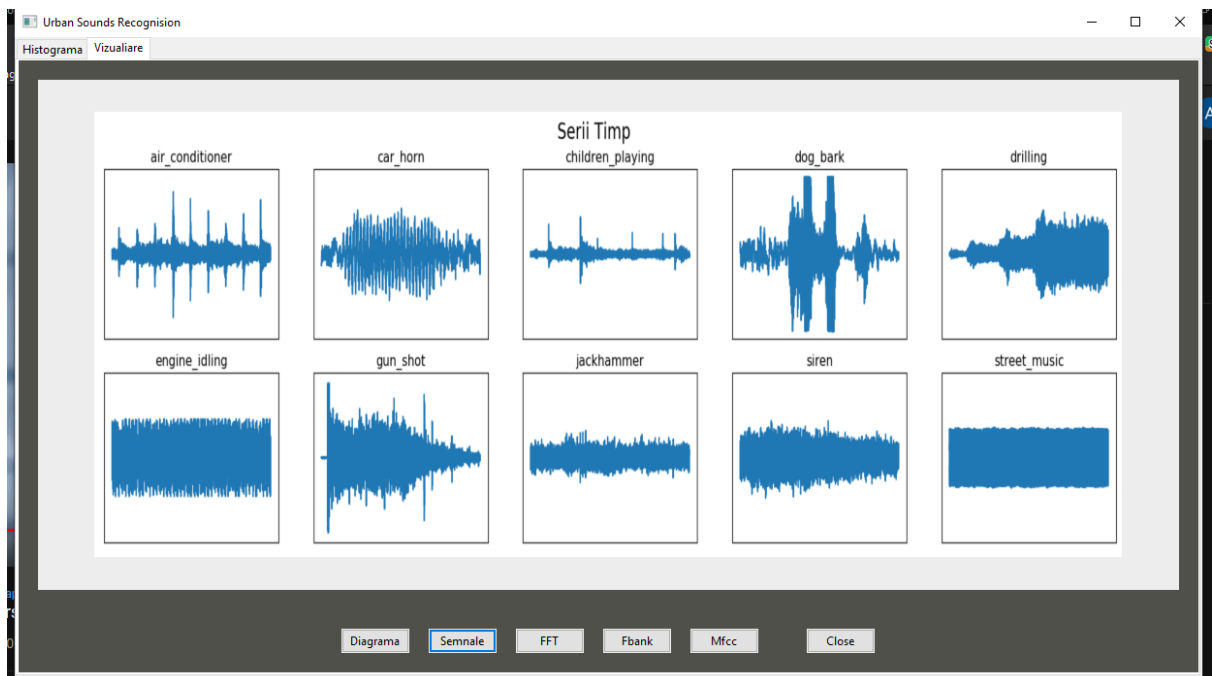


Fig.14

Ideea este ca filtrele le aplicam in ordine. După ce avem semnalele aplicam transformarea Fourier, FFT, Fbanks si Mfcc.

In imaginea de mai jos se va afișa Transformarea Fourier pe semnalele din imaginea anterioara:

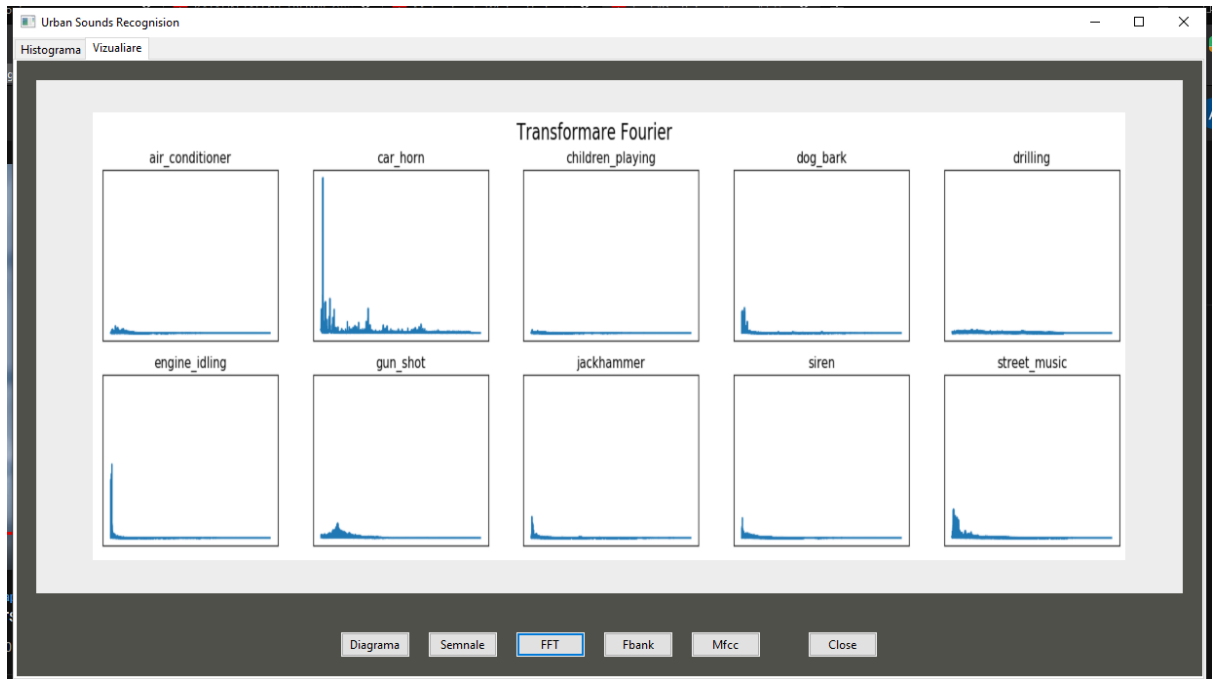


Fig.15

După ce am aplicat transformarea Fourier vom folosi filtrul Banks pe el:

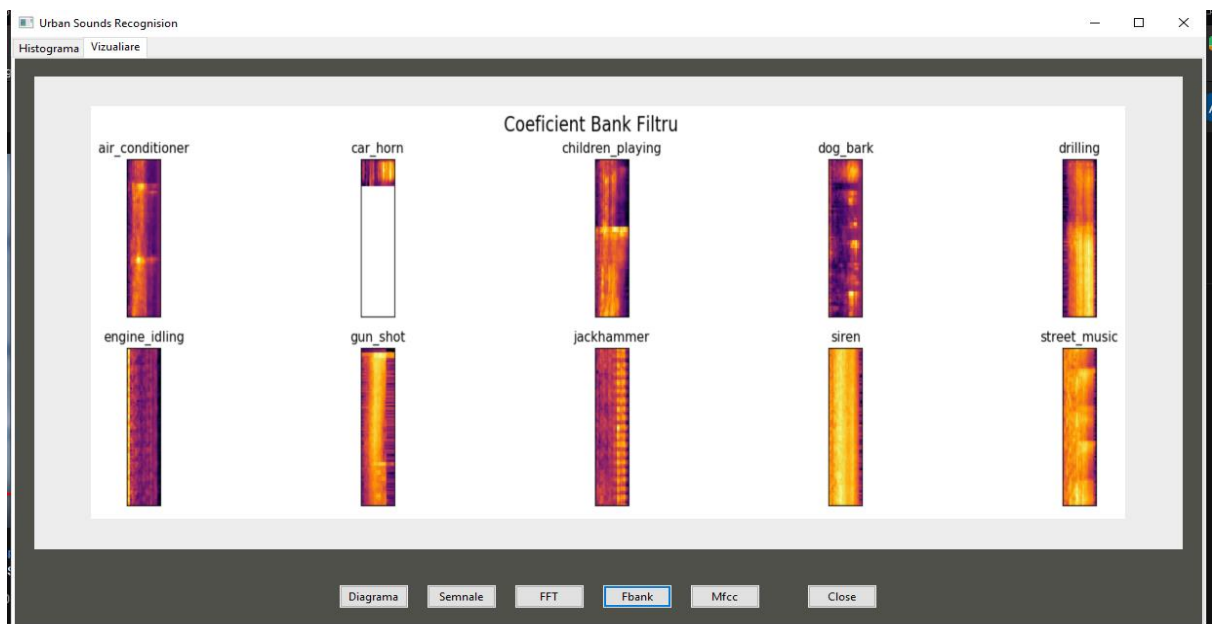


Fig.16

După ce am aplicat filtrul Banks se observa ca sunetele au o reprezentare grafică foarte diferita si datorita acestui lucru putem folosi rețeaua neuronală pe rezultatul obținut, dar datorita faptului ca filtrul Banks oferă limitări la predicție si la algoritmi pe care putem sa-i folosi vom interveni aplicând filtrul Mel si rezultatul obținut va fi folosit pentru Rețea.

Mai jos avem ultimul filtru aplicat folosind opțiunea Mel:

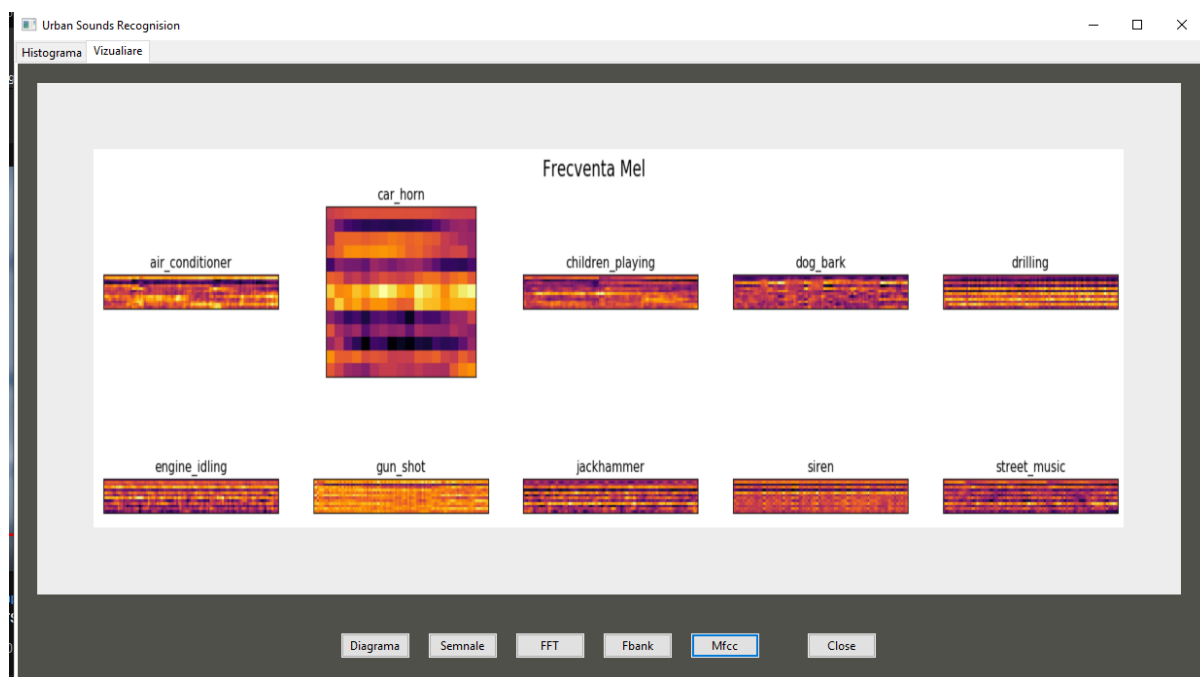


Fig.19

Acest filtru va fi folosit pentru a antrena modelul ce va fi folosit in recunoașterea sunetelor date ca input. E de precizat că aceste imagini sunt reprezentarea grafică a sunetelor din fiecare clasă si ele vor fi folosite pentru rețeaua noastră neuronală. Cu aceste imagini putem observa daca avem suficiente sunete pentru o clasă și la ce clasă vom putea întâlni dificultăți.

Concluzie

Sa prezentat cum un sunet poate fi preluat in domeniul digital și care sunt pașii de prelucrare a sunetului pentru a putea fi folosit in procese mai complexe. In cazul nostru sunetele au fost folosite in rețele neuronale cu scopul de a se crea o aplicație care poate imita omul in recunoașterea sunetelor, uneori chiar mai eficient decât o persoana la recunoașterea sunetelor neclare si zgomotoase.

Părerea mea este ca rețelele neuronale imită foarte mult creierul si in toate aplicațiile folosite trebuie sa imităm cât mai mult corpul uman si după acest pas să îl reprezentăm digital. Daca reușim sa facem acest lucru putem crea rețele care pot întrece creierul uman. Aplicația mea a fost concepută pentru ca un utilizator simplu sa testeze modelul si să poată crea modele care pot recunoaște sunete dar, modelele antrenate pot fi folosite si pe aplicații mai productive cum ar fi: alarme de securitate sau dispozitive de auz pentru persoanele in vârstă.

Bibliografie

- Christopher M. Bishop, Pattern Recognition and Machine Learning:
<http://practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs/>
- Haytham Fayek, Speech Processing for Machine Learning:
<https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html>
- Discrete Cosine Transformations:
<http://datagenetics.com/blog/november32012/index.html>
- Sound Data: <https://www.kaggle.com/papeloto/urban-sound-feature-extraction-knn/data>
- seth814, Audio-Classification: <https://github.com/seth814/Audio-Classification>
- WX Python Documentation: <http://zetcode.com/wxpython/>
- Keras : <https://keras.io/>
- Tensorflow Documentation: <https://github.com/tensorflow/docs>