Facultatea de Automatică şi Calculatoare

Calculatoare și Tehnologia Informației

Universitatea Politehnica Timișoara

Soluție Machine Learning pentru Recunoașterea Emoțiilor în Vorbire

Proiect de diplomă

Student

Steleac Raul-Dacian

Conducător științific:

Dr. Ing. Ștefan HOLBAN

Timișoara

2020

Cuprins

1 Introducere

1.1 Prezentarea Problemei

Comunicarea este o capacitate esențială pentru specia umană, fiind cea mai naturală și principala modalitate de transmitere a informației într-un mod direct. Totuși pe lângă informația lingvistică o mare parte din informațiile prezente în conversațiile pe care le avem zilnic sunt ascunse în emoțiile cu care rostim și articulăm diferite cuvinte, silabe și chiar litere. Urechea umană este capabilă să determine și cele mai mici inflexiuni din vocile participanților la conversație pentru a reuși să capteze cât mai bine sensul acesteia. Astfel, este de așteptat că mașinile care urmează să facă parte de acum din aceste conversații să fie la fel de competente din această privință. Domeniul științific care se ocupă cu crearea unor modele de tip Machine Learning pentru determinarea emoțiilor dintr-un discurs se numește "Speech Emotion Recognition", sau SER.

Proiectul de diplomă propus de mine încearcă să creeze o soluție pentru recunoașterea emoției în vorbire și să ofere un mediu de utilizare propice printr-o interfață grafică care conține diferite funcționalități de configurare a modelului, extragere de statistici și înregistrare. Utilizatorul are astfel opțiunea de a antrena soluția SER propusă de mine cu diferite configurații de parametrii cât și de a testa modelul obținut pe semnale audio pre-înregistrate sau înregistrate pe loc.

1.1.1 Importanța informației emoționale

Informația emoțională este prezentă în orice conversație și reprezintă un punct de reper către sensul din spatele cuvintelor vorbitorilor. Emoțiile pot pe de o parte să adâncească sensul unor anumite cuvinte sau chiar să îl întoarcă pe dos dacă sunt folosite în ipostaze contradictorii. Sarcasmul este un exemplu bun de acest fenomen, unde prin schimbări de tonalitate sensul cuvintelor este intors pe dos. Un participant la conversație care nu poate să înțeleagă aceste concepte va fi puternic dezavantajat. Deoarece oamenii sunt mai mult sau mai puțin experți în determinarea acestor informații emoționale, dorim ca și viitoarea generație de vorbitori, agenții AI, să poată să realizeze această performanță.

Chiar și cele mai primitive specii puteau recunoaște tonuri care reprezintă emoții precum dragoste, frică sau enervare, Blaton, 1915 [16]. Plecând de la acest considerent, putem să ne întrebam cât de tare se foloseau strămoșii nostrii, specia *Homo sapiens*, de emoții pentru a comunica înainte de apariția limbajului modern. Levinson & Holler, 2014 [17] susțin convenția că limbajul a apărut cu aproximativ o sută de mii de ani după apariția speciei umane, *Homo sapiens*, care se estimează a fi acum circa trei sute de mii de ani. Astfel printr-un calcul rapid puitem să determinăm că a existat o perioadă de circa o sută de mii de ani în care strămoșii noștri, *Homo sapiens*, deși capabili să folosesaca un limbaj pentru comunicare, nu au făcut-o. Totuși specia *Homo sapiens* trăia în grupuri, iar comunicarea între indivizi era esențială. Christiansen & Kirby, 2003 [18] susțin că înainte de apariția limbajului câteva "pre-adaptări" au trebuit să apară în comunitățile descendenților familiei *Hominidae*. Principalul candidat propus pentru aceste "pre-adaptări" a fost abilitatea de a folosi așa numite "simboluri", sunete și gesturi bogate în emoții, pentru a crea primele și cele mai de bază forme de limbaj precum dorință, plăcere, dezgust, tristețe sau enervare. Putem să observăm astfel cum deși informația lingvistică nu exista încă în comunicarea speciei umane, informația emoțională a fost inclusă încă de la primele forme de interacțiuni sociale.

Cantitatea de informație din spatele emoțiilor pe care le folosim astăzi în limbajul modern

rămâne la fel de importantă că pe vremea strămoșilor noștri. Diferența apare doar în capacitatea oamenilor din prezent dea a combina emoțiile cu diferite cuvinte pentru a discuta concepte mult mai complexe decât cele existente pe vremea strămoșilor nostrii. De aceea, în prezent, studiul importanței emoției dintr-o conversație este extins și în domeniul calculatoarelor prin inteligență artificială.

1.1.2 Domeniul recunoașterii emoției in vorbire

Domeniul "Speech Emotion Recognition", sau SER, are ca scop final construirea unui model de tip Machine Learning care să primească de la intrare o înregistrare audio, o parte dintr-o conversație, și să genereze la ieșire o emoție, care să fie reprezentativă pentru acea înregistrare.

Recunoașterea emoțiilor în vorbire este o problemă care a stârnit curiozitatea adepților domeniului inteligenței artificiale de câteva decenii. Dellaert et al., 1996 [19] au deschis granițele acestui domeniu în 1996 cu primul articol științific care încearcă să abordeze acest subiect. Aceștia au încercat să clasifice patru tipuri de emoții prin folosirea unor date de intrare, așa numite "prozodice", precum tonalitatea, intensitatea, frecvența sau amplitudinea, folosind trei tipuri de modele de clasificare diferite "Maximum Likelihood Bayes classifier" (MLB), "Kernel Regression" (KR) și "K-nearest neighbors" (KNN). Această implementare este una reprezentativă pentru abordarea recunoașterii de emoții în vorbire, realizând o separare clară între cele două module arhitecturale principale: extragerea datelor și clasifiactorul care urmează să fie antrenat. Discrepanța dintre arhitecturile folosite în prezent și cea prezentată mai sus rămâne însă observabilă. Chiar dacă datele de intrare prozodice sunt încă folosite astăzi, creșterea drastică a puterii de procesare a dus la folosirea unor arhitecturi cu rețele neuronale adânci care adoptă ori mai multe tipuri de date de intrare ori direct semnalul audio neprocesat, dacă modelul realizează extragerea datelor printr-o manieră automată, "end-to-end models".

Diferențele arhitecturale sunt totuși un semn bun, fiind reprezentative pentru evoluția domeniului de cercetare. SER și-a păstrat popularitatea în ultimele două decenii, deținând un număr bogat de articole științifice pe această temă. Aceste articole aduc noi interpretări atât din punctul de vedere al extragerilor caracteristicilor semnalului audio, folosite ca date de intrare, cât și a modelului clasificator antrenat. Totuși, deși noi idei și arhitecturi contiună să apară anual, această tehnologie nu a reușit să atingă încă o acuratețe sau o generalitate destul de satisfăcătoare pentru a fi lansată pe piață.

O tehnologie înrudită a recunoașterii emoției în vorbire, "Speech Recognition" care încearcă să determine informația lingvistică dintr-o conversație, a reușit să revoluționeze interfețele de comunicare dintre om și mașină. Această tehnologie își găsește locul în majoritatea telefoanelor, calculatoarelor, mașinilor și chiar a unor echipamente din jurul casei. Alexa, Cortana și Siri sunt câteva nume pe care majoritatea persoanelor le cunosc fără să le asocieze cu o față sau o persoană. Acești agenți inteligenți obțin rezultate excepționale în capacitatea lor de a menține o conversație cu clienții și de a răspunde la anumite cerințe ale acestora. Cu toate acestea, algoritmii de "Speech Recognition" nu resuesc mereu să răspundă corect la afirmațiile utilizatorilor, deoarece nu iau în considerare și partea emoțională a dialogului. Pentru a obține o interfață de comunicare om-mașină completă, informația emoțională este esențială. Prin diferite intonații sensul cuvintelor poate fi schimbat complet, iar un algoritm care se focusează doar pe informația lingvistică va rămâne inflexibil la aceste intonații, generând rezultate eronate.

1.2 Motivația problemei

Recunoașterea emoției în vorbire reprezintă un subiect extrem de interesant atât din punct de vedere aplicativ cât și personal. Potențialul acestui domeniu este ridicat din cauza numărului mare de aplicații care pot beneficia de încorporarea unui astfel de sistem. Modurile în care un sistem SER poate fi utilizat sunt limitate doar de nivelul tehnologic curent și imaginația programatorilor.

1.2.1 Motivație aplicativă

Aplicațiile în care această tehnologie poate fi folosită în viitor sunt greu de estimat, deoarece orice interfață om-mașină care folosește dialogul ca modalitate de transmitere de informație poate beneficia prin includerea unui astfel de algoritm.

Un bun exemplu este încorporarea unui algoritm SER în mecanismul de *"feed-back"* al unei companii. Principala modalitate prin care firmele din zilele noastre încearcă să capteze părerea publicului asupra unui produs este prin folosirea unor chestionare. Chiar dacă aceste chestionare au loc în scris sau telefonic aduc anumite limitări. În prima situație apare incertitudinea asupra onestității răspunsurilor oamenilor, iar în cea de a doua situație apare limitarea personalului disponibil care să asculte răspunsurile intervievaților în decursul chestionarului. Un sistem alcătuit dintr-un algoritm de "speech emotion recognition", împreună cu unul generic de "speech recognition", poate capta atât informația lingvistică cât și cea emoțională din răspunsurile la întrebările chestionarelor, observând atât cuvintele în sine cât și gradul de credibilitate bazat pe implicarea emoțională a participantului.

Un alt exemplu ar fi implicarea modelelor SER în tehnologiile care ne ușurează deja viața de zi cu zi. Agenții inteligenți și diferitele tipuri de roboți, ca de exemplu Huahu et al., 2010 [20], care apar tot mai des în apropierea oamenilor, pot găsi un mare avantaj în determinarea emoțiilor clienților când încearcă să răspundă cât mai exact la nevoile acestora. Spre exemplu un astfel de agent inteligent încorporat într-o mașină poate detecta dacă în timpul mersului șoferul este implicat într-o ceartă sau o discuție cu un puternic impact emoțional. Dacă acest lucru este adevărat, sistemul SER poate să-l îndrume pe șofer să oprească mașina până când discuția s-a terminat în scopul evitării unui accident din cauza lipsei de atenție. Alexa sau Siri, care sunt folosite la nivel global de mii de oameni în jurul casei, pot să încerce să ofere răspunsuri care să liniștească un client nervos sau să introducă mici glume pentu a încerca să înveselească un client trist.

Acești alogritmi ar putea fi folosiți și pentru a eficientiza educația. Prin introducerea unor receptoare de emoții, profesorii pot determina starea emoțională a studenților și pot extrage informații pentru îmbunătățirea calității orelor de curs. Spre exemplu profesorul poate folosi modelul de recunoaștere al emoțiilor pentru supravegherea continuă a interesului studenților sau pentru a crea strategii prin care să ridice moralul clasei când vine vorba de anumie subiecte predate care îi pot descuraja sau plictisi pe aceștia.

Alte exemple care merită menționate sunt folosirea acestor tipuri de algoritmi în: stații de call-center (Gupta & Rajput, 2007 [21]), jocuri video (Szwoch & Szwoch, 2015 [22]) sau evaluare psihologică (Lancker et al., 1989 [23]).

1.2.2 Motivație personală

Tema recunoașterii de emoții a început să mă intrige când mi-am pus problema construirii unui psiholog artificial. Deși crearea unui astfel de terapeut artificial este puțin probabilă, recunoașterea de emoții rămâne o problema cu potențialul de a fi rezolvată in viitor. În subiecte ca recunoașterea obiectelor, fețelor, și chiar a cuvintelor rostite s-a obținut o acuratețe destul de satisfăcătoare

pentru a fi introduse pe piață. Pentru domeniul recunoașterii de emoții în vorbire însă, acest lucru ramane inca o provocare.

Diferite cărți, filme sau seriale prezintă o multitudine de posibile utopii tehnologice care ar putea să devină realitate în următoarele decenii sau secole. Deși acestea sunt doar scenarii Sci-Fi, una dintre ideile comune este existența unei interfețe de comunicare verbală de la om la mașină aproape perfect identică, din punct de vedere calitativ, cu cea de la om la om. Sistemele de "Speech Recognition" deja existente oferă un bun exemplu prin succesul lor, care evidențiază importanța unei astfel de tehnologii, dar și a popularității ei în rândul publicului. SER încearcă să îmbunătățească aceste conversații oferind capacitatea mașinilor de a înțelege și emoțiile din spatele cuvintelor. Acest transfer de informație emoțională mi se pare extrem de interesant deoarece poate să ne ofere pe viitor capacitatea de a ne înțelege mai bine propriile emoții dar și de a crea agenți inteligenți care să se aproprie cât mai puternic de o inteligență de o generalitate asemănătoare cu a noastră.

Lipsa acestui domeniu pe piață, cât și potențialul pe care îl deține m-a motivat să aleg acest subiect pentru proiectul de diplomă. Deși implementarea pe care o propun obține rezultate asemănătoare cu unele din cele mai de success soluții găsite în diferite articole științifice, nu reușește să obțină încă o acuratețe și o generalitate destul de ridicată pentru a permite comercializarea acestor alogritmi. Soluția propusă de mine reprezintă o altă încercare de a aduce acest domeniu mai aproape de acel nivel necesar care îl va face viabil publicului.

1.3 Obstacole in studiul SER

Principalele piedici care despart domeniul SER de majoritatea aplicațiilor de Machine Learning și ii încetinesc acestuia progesul sunt legate de dificultatea obținerii unui set de date de intrare satisfăcător comparativ cu complexitatea problemei și lipsa unor caracteristici de intrare care să fie reprezentative pentru detectarea emoției. Aceste două considerente au alcătuit în decursul ulitmelor două deceni obstacole serioase in studiul și dezvoltarea modelelor de recunoaștere de emoții deoarece implică necesitatea folosirii unor resurse costisitoare din punct de vedere financiar, temporal și uman.

1.3.1 Impactul bazelor de date

Bazele de date aferente recunoașterii de emoții în vorbire suferă atât din punct de vedere cantitativ cât și calitativ. Bjorn, 2018 [24] susține că o particularitate a acestui domeniu de cercetare este subiectivitatea și incertitudinea ridicată în construirea bazelor de date.

Există două tipuri principale de baze de date în domeniul SER în funcție de modul în care acestea sunt obținute: jucate (de actori) sau spontane, iar ambele modalități suferă de diferite dezavantaje.

Pe de o parte, majoritatea bazelor de date care există sunt alcătuite prin înregistrarea unor actori profesioniști, studenți la actorie sau chiar persoane care primiesc o anumită propoziție și încearcă să o rostească în cadrul unei anumite emoții. Din punct de vedere calitativ, devine destul de aparent cum aceste emoții pot fi exagerate, lucru care face ca clasificatorul obținut să fie superficial în cazul detectării emotiiolor reale. Pe lângă această problemă, obținerea bazelor de date implică și o perioadă de verificare și filtrare. Înregistrările obținute sunt tranmise mai departe unor persoane, care nu au participat în partea de înregistrare, pentru a le valida. Dacă în urma acestui proces rezultatul este emoția intenționată inițial atunci înregistrarea este declarată validă și va fi folosită pentru antrenare. Totuși, problema principala este că nici oamenii nu

reușesc să determine perfect emoția predominantă dintr-un discurs. Acest lucru afectează direct corectitudinea bazei de date și acuratețea modelului. Din punct de vedere cantitativ, în procesul de antrenare sunt astfel implicate destul de multe persoane. Acest lucru îngreunează obținerea unor seturi de date numeroase deoarece acest proces devine dificil din punct de vedere financiar cât și temporal.

Pe de altă parte, există seturi de date în care emoțiile nu sunt jucate de actori profesioniști, ci sunt extrase din înregistrări în care acestea apar în mod spontan. În alcătuirea acestora, se aleg părți din diferite talk show-uri, înregistrării din call-centere, discuții la radio, și alte surse similare, iar apoi se depistează și se extrag fragmentele bogate în emoție. Un exemplu de acest tip de bază de date este "Multimodal EmotionLines Dataset" (MELD) [25], în care s-au preluat părți din episoadele celebrului serial "Friends". Obținerea datelor prin aceste metode devine mai dificilă atât din punct de vedere legal cât și etic, iar problema din varianta precedentă în care emoția depistată depinde doar de percepția persoanei care clasifică înregistrarea se menține. Astfel posibilitatea apariției de erori nu este evitată.

Concluzia pe care o putem trage este că indiferent de varianta aleasă nu putem scăpa de incertitudinea adusă de discernământul uman în clasificarea datelor de intrare. Multe modele propuse susțin ideea folosiri înregistrărilor atât din prima cât și din a două categorie pentru a echilibra dezavantajele impuse de ambele.

Un alt obstacol întâmpinat de mine in decursul acestei lucrări de licență a fost obținerea unui set satisfăcător de baze de date pentru antrenarea modelului Machine Learning. Dificultatea inregistrării emoțiilor face ca unele din bazele de date existente să fie private și să necesite sume mari de bani pentru obținerea lor. Din acest motiv am fost limitat din privința datelor de intrare pe care le-am putut folosi.

1.3.2 Dificultatea extragerii informației emoționale

O altă mare dilemă cu care s-au confruntat multe articole științifice a fost determinarea unui set de caracteristici ale semnalului audio care să eficientizeze clasificarea emoției. Din punctul de vedere al extragerii informației emoționale momentan există două modalități principale: folosirea unor caracteristici obținute matematic prin formule predefinite (hand-crafted feaures) sau prin folosirea unor rețele neuronale care prin antrenare să găsească automat cele mai eficente informații din datele de intrare (end-to-end features).

În cazul în care se folosesc coeficienți obținuți prin formule matematice generice ca "Mel-frequency cepstrum coefficiants", "Roll-off coeficiants", "detas and delta deltas" etc., detaliați la 3.3, nu s-a găsit un set de caracteristici de acest tip care să fie considerate ideale pentru obținerea informației emoționale. Coeficienții înșirați mai sus sunt preluați din "Speech Recognition" pentru că reprezintă caracteristicile necesare identificării informației lingvistice. Totuși, nu s-a demonstrat că aceștia pot fi la fel de benefici și în cazul determinării emoțiilor, lucru care face că majoritatea studiilor în SER să folosească seturi de caracteristici de intrare diferite.

Coeficienții extrași prin rețele neuronale profunde, detaliați la 3.4, sunt considerați a fi mai subiectivi sarcinii de detectare a emoției in vorbire, deoarece fac parte din procesul de antrenare al clasificatorului. Datorită faptului ca nu putem înțelege sau replica calculele realizate în diferitele rețele neuronale folosite, nu putem determina ce semnifică rezultatul fiecărui nivel din rețea, cu atât mai puțin a fiecărui nod. Astfel nu putem să facem o inferență directă pe coeficienții obținuți.

Ambele variante au reușit să producă rezultate performante, iar multe studii s-au realizat în găsire soluției celei mai eficiente în ambele situații. Cu toate a acestea cele două nu reușesc să rezolve problema inițială, adică găsirea unui set de caracteristici reprezentative pentru emoția din înregistrările audio.

Structura lucrării

Studiul recunoașterii emoției in vorbire este un domeniu de cercetare în continuă creștere și are ca scop final obținerea unui model capabil să determine, să înțeleagă și să răspundă la diferitele emoții prezentate de utilizatorul uman. Deși natura problemei implică diferite dificultăți când vine vorba de gestionarea bazelor de date și extragerea informațiilor relevante din semnalul audio, aceste probleme pot fi rezolvate prin aplicarea diferitelor tehnici prezente în lumea inteligenței artificiale de astăzi. În acest mod, detectarea emoțiilor din vorbire rămâne un domeniu de studiu care are potențialul să aducă îmbunătățiri puternice în interfețele de comunicare om-mașină din viitorul apropiat.

Structura capitolelor care urmează să detalieze soluția propusă în această lucrare de diplomă atât pentru sitemul de recunoaștere a emoțiilor în vorbire cât și pentru interfața grafică este următoarea:

- Capitolul 2 - *Analiza stadiului actual în domeniul problemei*, descrie componentele necesare pentru alcătuirea unui sistem SER, trei exemple de arhitecturi de succes din domeniu și o scurtă prezentare a soluției propuse.

- Capitolul 3 - *Bazele teoretice*, prezintă conceptele teoretice care stau la baza arhitecturii sistemului SER, incluzând metodele folosite pentru extragerea caracteristicilor de intrare și componentele modelului clasificator.

- Capitolul 4 - *Descrierea implementării*, detaliază tehnologiile folosite, descrierea secvențelor de cod care constituie componentele principale ale lucrării și utilizarea interfeței grafice.

- Capitolul 5 - *Rezultate și experimente*, enumerară și descrie diferite configurații experimentale încercate și rezultatele obținute comparativ cu alte soluții din domeniu.

- Capitolul 6 - *Concluzii*, prezintă un sumar al lucrării de diplomă împreună cu o listă de posibile imbunatatiri viitoare ale soluției de recunoaștere a emoțiilor în vorbire propuse.

2 Analiza stadiului actual în domeniul problemei

Definim un sitem SER ca o colecție de metodologii care procesează semnalele audio aferente unui discurs pentru a detecta emoția încorporată în ele. Ca orice altă problemă de clasificare, un sistem SER trebuie să îndeplinească un anumit set de pași, care odată organizați cronologic constituie modelul Machine Learning propus.

Orice sistem SER necesită un clasificator, o entitate care reprezintă metoda de învățare supervizată. Un astfel de sistem supervizat implică folosirea unor date catalogate. În cadrul recunoaterii de emoții în vorbire, datele de intrare sunt semnalele audio cu emoțiile incorporate. Cu toate acestea, semnalul audio neprocesat nu reprezintă o formă care sa faciliteze detectarea acelor emoții. Prin aplicarea a diferite tehnici, informația emoținala este extrasă din semnalul audio și oferită într-o nouă formă, mai eficientă, clasificatorului. Înainte ca aceste caracteristici emoționale să poată fi extrase, semnalele trebuie să treacă și printr-un stagiu de preprocesare.

În continuare voi prezenta pașii necesari, în ordinea lor cronologică, și voi menționa câteva dintre configurațiile alese de dezvoltatori pentru rezolvarea recunoașterii de emoții în vorbire.

2.1 Tipologii arhitecturale in SER

2.1.1 Preprocesarea datelor de intrare

Preprocesarea datelor este primul pas în construirea majorității modelelor Machine Learning. În "Speech Emotion Recognition", preprocesarea datelor este vitală deoarece poate elimina multe din dezavantajele existente în bazele de date din această ramură a inteligenței artificiale.

Semnalul brut trece în prima faza printr-un proces de partiționare în segmente de lungime fixă. Această partiționare este avantajosă pentru algoritmii SER deoarece permite determinarea relațiilor temporale din interiorul înregistrării (fiecare segment, "frame", fiind considerat un punct pe axa temporală). Următorul pas în procesul de preprocesare este aplicarea unor funcții fereastră, detaliate la 4.1.2, pe fiecare segment. Utilizarea funcțiilor fereastră are ca scop reducerea pierderii de informații după aplicarea transformărilor Fourier care apare din cauza discontinuității de la marginea segmentelor. Transformatele Fourier sunt folosite pentru a extrage coeficenții folosiți ca caracteristici de intrare, deoarece acestea convertesc un semnal audio din spațiul timp în spațiul frecvență și ne permit să observăm diferite proprietăți specifice ale semnalului, care rămân "ascunse" inițial în domeniul temporal.

Cei doi pași prezentați anterior sunt necesari pentru a aduce semnalul audio intro formă care face antrenarea posibilă. Din acest motiv, aceștia sunt prezenți în orice model care folosește semnalul sonor ca date de intrare.

În continuarea fazei de preprocsare diferite implementări a modelelor SER optează să folsoeasca diferite tehnici care aduc avantaje serioase în faza de antrenare. Câteva din principalele tehnici folosite sunt:

- Normalizare per vorbitor
- Normalizare în funcție de sex
- Normalizare per baze de date
- Algoritmi de reducere a zgomotelor
- Algoritmi pentru identificarea segmentelor ce conțin o voce umană

- Reducerea dimensionalității

Alegerea acestor tehnici este complet subiectivă fiecărei implementări, iar avantajele aduse sunt cântărite în comformitate cu tipul de clasificator folosit. De exemplu, normalizarea per vorbitor reduce impactul diferențelor legate de tonalitatea vocii sau a microfonolui folosit de fiecare vorbitor. Acest tip de normalizare a înregistrat deja un succes într-un sistem SER detaliat în Bjorn et al., 2010 [27].

2.1.2 Extragerea datelor de intrare

Extragerea caracteristicilor semnalului audio reprezintă un aspect de mare importanță în domeniul recunoașterii emoțiilor în vorbire. Obținerea unui set de caracteristici care să curpindă informația emoțională cât mai precis are un impact considerabil asupra acurateții modelului clasificator. Diferite configurații de aceste seturi de date au fost propuse pentru sistemele SER, dar, cum am menționat și în sub-capitolul 1.3.2, nu s-a ajuns la un consens care să faciliteze recunoașterea emoțiilor.

În total există patru tipuri de caracteristici care pot fi extrase din semnalul audio, dar majoritatea articolelor științifice din SER se concentrează pe cele prozodice și spectrale.

Oamenii se folosesc de durată, intonație și intensitate pentru a crea diferitele secvențe sonore atunci când rostesc un discurs. Încorporarea acestor prozodii induce caracterul natural în convorbirile noastre. Koolagudi et al., 2012 [26] susțin că în literatura stințifică, caracteristicile prozodice precum energia, durata, amplitudinea și derivatele acestora sunt considerate a fi puternic corelate cu emoțiile [19, 28, 29]. Caracteristici ca minimul, maximul, media, variația, lungimea și deviația standard a energiei semnalului audio, și funcții similare ale amplitudinii sunt folosite astfel ca surse de informații prozodice în majoritatea sistemelor SER.

Când un sunet este produs de un om, acesta trece prin tractul vocal și este puternic influențat de forma acestuia. Caracteristicile acestui tract vocal sunt foarte bine ilustrate în domeniul frecvență. Pentru a profita de aceste informații se folosesc caracteristicile specializate pe extragerea informației din domeniul frecvență, denumite spectrale. Acest tip de caracteristici sunt obținute prin folosirea celebrelor transformate Fourier. Exemple ale unoara din aceste tipuri de caracteristici folosite în recunoașterea emoției în vorbire sunt: Mel Frequency Cepstral Coefficients (MFCC), Linear Prediction Cepstrum Coefficients (LPCC), Gammatone Frequency Cepstral Coefficients (GFCC) etc.

Tehnica de extragere de informații care folosește formule matematice pentru determinarea caracteristicilor prezentate mai sus se numește în domeniul științific "hand-crafting". Deși această tehnică a obținut rezultate destul de satisfăcătoare în ultiemele decenii, din cauza dezavantajelor prezentate în sub-caiptolul 1.3.2, multe implementări mai noi ale sistemelor SER încearcă să realizeze extragerea caracteristicilor de intrare într-o manieră automată.

"End-to-end modeles" se referă la o tehnică de automatizare completă a modelelor Machine Learning prin care inclusiv extragerea datelor este obținută prin antrenare. În SER acest lucru se realizează de obicei prin extragerea spectogramei Mel, 3.4.1, din sunetul brut și aplicarea unei rețele neuronale convoluționale, 3.4.3, cu un număr arbitrar de nivele [30, 31]. Aceste nivele interpretează spectograma ca o imagine generică și își adaptează filtrele pentru a extrage caracteristicile considerate importante din aceasta. Prin folosirea acestui tip de extragere de caracteristici, modelul SER poate identifica singur în timpul antrenării ce informații din semnalul audio sunt cu adevărat importante în cazul recunoașterii emoțiilor. Adoptarea acestei tehnici a fost benefică în cazul multor soluții din SER [30, 31, 32, 33, 34, 35], și este folosită și in implementarea propusă în această lucrare de licență.

2.1.3 Clasificatorul

Un alogritm de clasificare necesită un set date de intrare X, un set de clase de ieșire Y, și o funcție care realizează maparea lui X la Y în forma următoare $f(X) = Y$. Scopul clasificatorului este de a crea o aproximare a funcției f bazată pe perechile de antrenare (x_i, y_i) care să faciliteze predicția corecta în cazul unor noi date de intrare.

Procesul de alegerae a unui model de clasificare în domeniul recunoașterii de emoții în vorbire, la fel ca în cazul majorității problemelor Machine Learning complexe, nu prezintă o soluție general valabilă. Studiile pe această temă aleg un astfel de alogritm printr-o manieră empirică. Cu toate acestea, natura problemei face ca un anumit set de algoritmi de clasificare să fie mai avantajoși.

Cele mai folosiți algoritmi de clasificare în domeniu SER sunt: Hidden Marko Model (HMM), Gaussian Mixture Model (GMM), Support Vector Machines (SVM) și diferite tipuri de rețelele neuronale artificiale că rețele convoluționale și recurente. Pe lângă acestea au mai fost folosite și alte tehnici ca: Arbori decizionali (DT), k-Nearest Neighbor (k-NN). k-means și Naive Bayes. Pentru a obține o acuratețe cât mai ridicată s-a optat și spre utilizarea unor modele alcătuite prin combinarea mai multor algoritmi de clasificare, Mehmet et al., 2020 [36].

2.1.4 Tehnici de îmbunătățire a clasificării

Deși multe rezultate bune au fost obținute în SER prin folosirea doar a pașilor enumerați mai sus, în multe studii s-a demonstrat o îmbunătățire a acestor rezultate prin folosirea anumitor tehnici specifice din domeniul Machine Learning.

Una din aceste tehnici este folosirea unui *mecanism de atenție*. Mecanismul de atenție are ca scop să focalizeze atenția modelului pe segmentele bogate în informații emoționale. În cazul SER, mecanismul de atenție este folosit pentru a determina segmentele din semnalul sonor care conțin un grad de informație emoțională ridicată și a mari influența acestora în decizia clasificatorului. Acest mecanism este alcătuit dintr-un număr de ponderi antrenate în procesul de învățare, care se aplică direct pe ieșirile rețelelor neuronale având efectul prezentat anterior. Rezultatele benefice obținute în urma aplicării au fost observate în studiile: Misramadi et al., 2017 [37], Zhang et al., 2019 [32] .

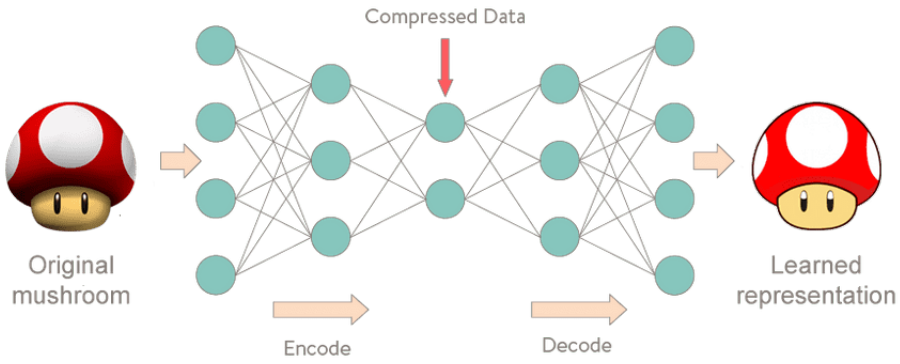


Figura 2.1: Exemplu de arhitectură auto-encoder. În partea stângă se poate observa imaginea inițială iar în partea dreaptă varianta compresata a acesteia obținută prin aplicarea auto-encoder-ului. Rosebrock, 2020 [38]

O altă tehnică este folosirea unor tipuri de rețele neuronale specifice, folosite pentru procesarea datelor de intrare sau chiar crearea unora noi. Aceste rețele neuronale sunt numite *autoencoders*. Autoenecoder-ele sunt alcătuite din minim trei nivele. Diferența față de rețele neuronale apare în faptul că dimensiunea intrărilor și ieșirilor este egală, în timp ce nivelele

"ascunse", din interiorul rețelei, au dimensiuni mai mici. Astfel autoencoder-ele sunt altacuite din două părți: "encoder" și "decoder". Encoder-ul compreseaza datele cu scopul de a obține o varianta cât mai eficientă în care informațiile principale sunt încă păstrate. În schimb, decoder-ul are ca scop aducerea acestei forme compresate la o formă cât mai apropriată de cea inițială. Datele care trec prin această rețea sunt filtrate pentru a păstra doar informația complet necesară, Fig.2.1. Prin modificări ușoare în arhitectură se pot obține funcționalități complet noi, ca de exemplu "Denoising Autoencoders" (DAE), care după aplicarea unui zgomot la datele de intrare au ca scop să determine ponderile necesare pentru extragerea acelui zgomot și readucerea intrărilor la o formă cât mai apropiată de cea "curată". În SER mai multe tipuri de autoencoder-e au fost folosite în încercarea de a mări acuratețea sistemului: Denoising Autoencoders (DAE) Chao et al., 2014 [39], Adaptive Denoising Autoencoders (ADAE) Deng et al., 2014 [40], sparse autoencoder (SAE) Deng et al., 2013 [41], adversarial autoencoder (AAE).

Alte tehnici folosite sunt:

- "Multitask Learning", unde din cauza similitudinii dintre anumie sarcini părți dintr-un clasificator pot fi antrenate pe mai multe sarcini mărind astfel generalitatea modelului.
- "Transfer Learning", Prin această tehnică s-a încercat depășirea dezavantajului legat de lipsa bazelor de date suficiente. Astel diferite implementări se folsoesc de părți din alte modele care au fost pre-antrenate pe probleme similare ca "Speech Recognition" înainte de a începe antrenarea modelului pe cele specifice SER.
- "Voice Detection", Acest algoritm este folsoit pentru exluderea segmentelor care nu conțin vocea umană, pentru a reduce posibilele erori aduse de zonele lipsite de informație emoțională.

2.2 Prezentarea unor implementări din SER

Cum am menționat și în capitolul precedent, "Speech Emotion Recognition" nu a ajuns în punctul în care poate fi pus pe piață, astfel am decis să fac o comparație teoretică incercând să prezint alte moduri de implementare prezente în câteva articole de cercetare. În continuare voi prezenta trei arhitecturi de sisteme din recunoașterea a emoției în vorbire, care susțin câteva din principalele idei pe care și eu mi-am bazat modelul. Deși prezintă unele similarități, acestea nu pot fi comparate in mod perfect, deoarece folosesc atât baze de date diferite cât și caracteristici de intrare diferite. Deoarece nu există un mod consacrat de a construi un model SER, avantajele și dezavantajele dintre diferitele implementări devin dificil de identificat.

2.2.1 A Cross-corpus Study on Speech Emotion Recognition

Milner et al.(2019) în articolul de cercetare "A Cross-corpus Study on Speech Emotion Recognition" [42] folosesc un model antrenat pe mai multe baze de date constituite din înregistrări în aceiași limbă, engleză, cu voci din aceeași grupă de vârstă, adulți. Acest articol încearcă să determine beneficiile folosirii unor emoții jucate de actori profesioniști în combinație cu unele naturale. Cu atât mai mult, studiul își propune să prezinte și avantajele folosirii conceptului de "multi-task learning" unde părți din același model sunt antrenate pe diferite sarcini asemănătoare pentru a mării eficiența antrenării pe același set de date de intrare.

Arhitectură implementării propuse în Milner et al., 2019 [42] implică folosirea unui set de caracteristici de intrare "hand-crafted" generate prin extragerea coeficienților MFCC, PLP ("perceptual linear prediction") și COVAREP [43] din înregistrările audio. Modulul clasificator

al arhitecturii este asemănător cu cel folosit de mine în acest proiect fiind constituit din două nivele de celule recurente LSTM bidirecționale urmate de un mecanism de atenție, detaliate în 3.5 respectiv 3.6. Setul de emoții clasificate este alcătuit din: fericire, tristețe, enervare, surprindere, dezgust, frică și neutru.

"Cross-corpus" se referă la antrenarea modelului folosind pe rând câte una din bazele de date dintr-un set și apoi testarea pe fiecare din cele rămase. "Multi-domain" înseamnă antrenare pe toate bazele de date și apoi testare pe anumie părți din fiecare. Motivele principale pentru care aceste tehnici sunt folosite în practică sunt mărirea generalității modelului și combatrea numărului scăzut de înregistrări per baza de date.

În acest articol s-au obținut rezultate foarte bune pentru ambele metode de utilizarea a bazelor de date, acurateţe ne-ponderată de 81.94% în cazul "cross-corpus" și 82% în cazul multi-domain.

Antrenarea "Multi-domain" nu a fost totuși cea mai de succes metodă folosită în acest articol de cercetarea. Milner et al., 2019 [42] propun și folisrea tehnicii numite "domain adversarial training", unde pe lângă sarcina clasificării emoției, o parte din model a fost antrenată să recunoască și baza de date din care înregistrarea face parte. Acest mecanism funcționează ca un regularizator în procesul de calcularea a erorii, fiind adunat la eroarea rezultată din sarcina principală, SER. Prin introducerea acestei îmbunătățiri acuratețea modelului crește atingând, 82.26%.

2.2.2 Improved End-to-End Speech Emotion Recognition Using Self Attention Mechanism and Multitask Learning

Li, Yuanchao et al., 2019 [33] au reușit să obțină o acurateţe neponderată cu 14.3% mai ridicată față de cea mai mare inregistrată de soluțiile tradiționale, prin metoda propusă. Această metodă se bazează pe conceptul de modele "end-to-end". Totuși arhitectura propusă se folosește și de alte tehnici ca mecanismul de atenție și antrenare "multi-task" pentru a depăși unele obstacole în recunoașterea emoțiilor în vorbire.

Tehnica de extragere a caracteristicilor de intrare printr-un algoritm Machine Learning face ca toate modulele de procesare din interiorul modelului să fie antrenabile. De aici apare și numele tehnicii, "end-to-end". Aceste modele sunt extrem de avantajoase în SER, obținând rezultate încurajatoare [34, 35]. Folosirea unei astfel de extrageri "automate" a caracteristicilor semnalelor reduce influența umană implicată în crearea modelului, deoarece nu mai necesită părerea unor specialiști în domeniul audio pentru a determina cele mai eficente caracteristici de intrare. Avantajele folosirii tehnicii "end-to-end" sunt descrise mai în detaliu atât în 1.3.2 cât și în 3.4.

Asemănător cu soluția propusă în Milner et al., 2019 [42], arhitectura folosește două nivele recurente bidirecționale la care s-a atașat un nivel de atenție și tehnica de învățare "multi-task". Cu toate acestea, cea de a două sarcină pe care o execută clasificatorul nu mai este recunoașterea bazei de date ci a sexului persoanei care vorbește în înregistrare. Prin folosirea acestui tip de antrenare clasificatorul are posibiltatea să învețe diferențele între caracteristicile vocii unui vorbitor masculin și feminin.

Modelul prezentat în acest articol științific folosește o singură bază de date de intrare. Astfel modelul devine specializat în a recunoaște emoții pe acel set de date, dar va da un randament mai slab în inferență pe înregistrări din afara acestui set. Mulțimea de emoții clasificate este mai redusă decât în cazul precedent, fiind constituită din emoțiile fericire, tristețe, enervare, și neutru, obținând o acurateţe de 82.8%, care depășește precizia maximă înregistrată, 68.5%, de metodele precedente pe aceași bază de date.

2.2.3 Automatic speech emotion recognition using recurrent neural networks with local attention

Misramadi et al. (2017) se focusează în articolul [37] pe evidențierea avantajului folosirii unei rețele neuronale recurente urmate de un nivel de atenție pentru studiul recunoașterii de emoții în vorbire. Succesul folosirii rețelelor recurente în domeniul SER a fost înregistrat în diferite soluții de-a lunglul anilor [33, 42, 44, 45]. Aceste rezultate prezintă cum folosirea unor rețele recurente profunde face ca modelul să învețe să recunoască atât informațiile emoționale de scurtă durată, per segment, cât și să extragă relațile temporale dintre acestea pe o perioadă mai lungă de timp.

Pe lângă folosirea acestor rețele neuronale ca modul principal de clasificare, in Misramadi et al. (2017) se propune folosirea unui mecanism de atenție bazat pe o sumă ponderată (prezentat în 3.6). Ponderile acestui mecanism sunt la rândul lor antrenate în procesul de învățare. În acest articol, tehnica de atenție este propusă că o îmbunătățire la arhiecturile tradiționale ale sitemelor SER bazate pe rețele recurente. Beneficile aduse de acest mecanism sunt comparate cu alte modalități, mai puțin de succes, de combinare a emoțiilor din segmentele semnalului audio pentru a obține informația emoțională totală pe intreagă înregistrare.

Celelalte modalități care realizează această sarcină sunt, simpla recunoaștere a emoției în fiecare segment, folosirea informației emoționale doar din ultimul segment și realizarea mediei informațiilor emoționale din toate segmentele. Aceste tehnici prezintă un număr de dezavantaje care sublinează importanța folosirii unui mecanism de atenție ponderat. În primul rând, nu este rezonabil să asumăm că fiecare segment din înregistrarea audio conține informație emoțională. Deoarece pauzele în vorbire și zgomotul de fundal au o frecvență mare în majoritatea înregistrărilor, modelul ar trebui să fie cababil să filtreze aceste segmente care pot să îi dăuneze acurateții acestuia. Pe lângă asta, asumarea că ultimul segment conține informația emoțională totală a semnalului audio este falsă deoarece dacă informația emoțională principală se află la începutul propoziției (de exemplu un râset în prima secundă și apoi linișite până la finalul înregistrării) modelul va devia de la emoția recunoscută atunci din cauza influențelor celorlalte segmente care se suprapun peste informația emoțională inițială. Utilizarea operației de medie asupra întregului set de segmente nu este nici ea eficientă, deoarece segmentele lipsite de emoții vor continua să aibe un efect asupra clasificării.

Din aceste motive, Misramadi et al., 2017 [37] propun folosirea unei sume ponderate, ponderile fiind invățate la antrenare, care va reuși să determine segmentele bogate în emoții și să își îndrepte atenția doar asupra acestora. Soluția prezentată în acest articol folosește o singură bază de date, extragerea datelor caracteristicilor este "hand-crafted", clasificatorul este alcătuit din două nivele recurente BLSTM (3.5), iar setul de emoții clasificate este același cu cel folosit și în secțiunea precedentă: fericire, tristețe, enervare și neutru. Rezultatele obținute în acest studiu înregistrează o acuratețe de 63.5% pe baza de date IEMOCAP [68].

2.3 Prezentarea soluției propuse

Considerând diferitele obstacole ale domeniului recunoașterii emoției în vorbire enumerate în sub-capitolul 1.3 și arhitecturile descrise în secțiunea anterioară, soluția pe care o propun în această lucrare de diplomă pentru recunoașterea emoției în vorbire este ilustrată în Fig 2.2.

Soluția SER propusă conține două module arhitecturale principale aferente extragerii caracteristicilor de intrare și construirii modelului clasificator. În continuare vor fi descrise pe scurt componentele care alcătuiesc aceste module, urmând să fie detaliate teoretic și practic în capitolele următoare.

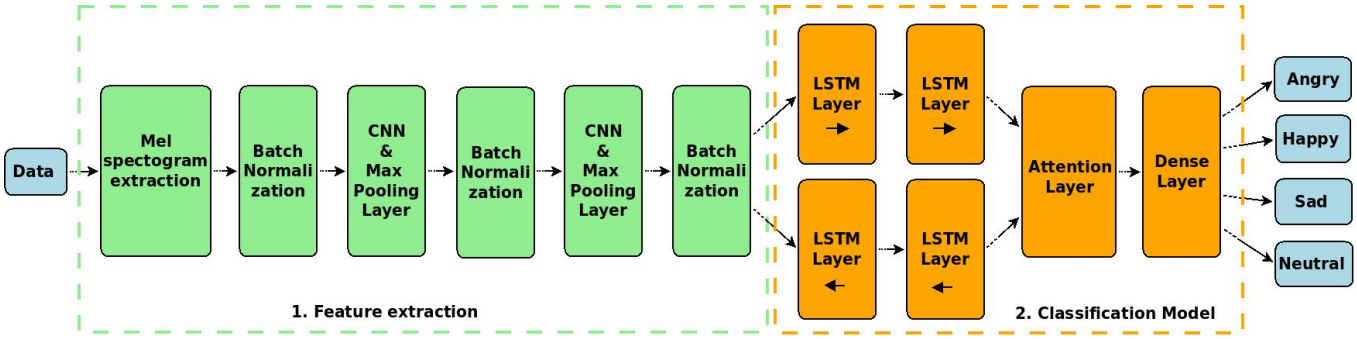


Figura 2.2: Arhitectura propusă pentru sistemul de recunoaștere a emoțiilor, alcătuită din cele două etape principale: extragerea caracteristicilor de intrare din semnalul audio (verde) și modelul clasificator. (portocaliu).

Datele de intrare folosite în această lucrare provin dintr-un set de mai multe baze de date alcătuit din: 'EMO-DB', 'RAVDESS', 'EMOVO', 'MAV', 'ENTERFACE' și 'JL'. Similar cu motivația prezentată la 2.2.1, decizia folosirii unui număr ridicat de baze de date este bazată pe depășirea dezavantajului numărului scăzut de exemple pentru antrenare specific domeniului SER, detaliată în 1.3.1. Deoarece semnalele audio sunt înregistrate în locații și limbi diferite, generalitatea modelului crește considerabil și face posibilă inferența pe înregistrări din afara acestor seturi de date.

Extragerea caracteristicilor de intrare din semnalul audio este realizată atât în manieră "end-to-end" folosind o rețea neuronală convolutionala, 3.4.3 cât și în manieră "hand-crafted" folosind operații matematice predefinite, 3.3. Arhitectura propusă, Fig. 2.2, folosește extragerea "end-to-end" iar metodă "hand-crafted" este folosită doar ca un termen de comparație și este prezentă doar în modul de utilizare pentru antrenare. Această decize este bazată pe rezultatele promițătoare obținute prin folosirea arhitecturilor "end-to-end" în mai multe sisteme SER și pentru că, după cum am menționat anterior, nu s-a descoperit încă un set de caracteristici "hand-crafted" care să cuprindă perfect informația emoțională. Între fiecare nivel al rețelei convoluționale am introdus tehnica numită "batch-normalization", 3.4.2, pentru a reduce fenomenul de expolize al gradienților și a grăbi procesul de antrenare.

Structura internă a modulului clasificator ales a fost bazată pe raționamentul detaliat în Misramadi et al., 2017 [37]. Astfel, modelul clasificator este unul complex altcatuit din 3 componente: rețeaua recurentă, mecanismul de atenție și rețeaua neuronală densă, după cum se poate observă și în Fig. 2.2.

Rețeaua neuronală recurentă este constituită din două celule recurente BLSTM ("Bidirectional Long Short-Term Memory") pe două nivele. Această tipologie de rețea neuronală a fost aleasă pentru a profita de relațiile temporale ale informațiillor emoționale dintre segmentele audio aflate la momente diferite. Rețeua recurentă este urmată de un mecanism de atenție care permite modelului să se focuseze doar pe segmentele semnalului audio bogate în emoție. Rețeaua neuronală generică densă este concatenată la modulul clasificator pentru a realiza translatarea rezultatelor rețelei recurente și a mecanismului de atenție în distribuția de probabilitate a emoțiilor clasificate.

Numărul de emoții clasificate este patru, la fel că în arhitecturile din Misramadi et al., 2017 [37] și Li, Yuanchao et al.2019 [33]: fericire, tristețe, enervare și neutru.

Lucrarea conține și o interfață grafică care permite utilizatorului să utilizeze sistemul de recunoaștere a emoțiilor în vorbire descris mai sus printr-o gamă largă de funcționalități. Interfața grafică permite două moduri de utilizare: antrenare și inferență. În partea de antrenare, utilizatorul poate să modifice configurația parametrilor modelului și să observe diferite statistici legate de starea acestuia în timpul procesului de învățare. În modul de utilizare inferențial, utilizatorul poate să clasifice emoțiile din fișiere audio pre-înregistrate sau din semnale înregistrate pe loc

prin intermediul interfeței grafice.

3 Bazele teoretice

Domeniul inteligenței artificiale diferă de oricare altă ramură a științei calculatoarelor deoarece își propune să rezolve problemele printr-o manieră stohastică. Avantajul privirii problemelor într-un mod probabilistic este că ne permite să aplicăm alogritmii obținuți direct în lumea reală. Algoritmii clasici pot fi extrem de performanți când vine vorba de a găsii soluții în timp polinomial, dar neputincioși dacă problema necesită soluții de un grad mai înalt de complexitate, timp exponențial. Lumea în care trăim este plină de astfel de probleme, iar abordările clasice funcționează doar pe difeirte abstractizări în care aspectele stohastice sunt eliminate aproape complet.

Metoda propusă de domeniul inteligenței artificiale este de a înlocui abordarea tradițională în care programatorul dictează pașii care duc la rezolvarea problemei cu un proces de antrenare prin care algorimtului îi sunt oferite doar un set de unelte pe care trebuie să învețe să le folosească singur din greșeli.

Recunoașterea emoțiilor în vorbire face parte din mulțimea acestor probleme denumite "grele", putând deveni o sarcină dificilă chiar și pentru oameni. Rezolvarea unei astfel de probleme poate fi realizată doar prin folosirea unei arhitecturi bazate pe inteligența artificială. Din acest motiv în urma studierii mai multor implementări prezentate în diferite articole științifice din acest domeniu am decis să construiesc un model de clasificare cu arhitectură din Fig. 2.2.

După cum se poate observă în figura de mai sus, sistemul SER propus este alcătuit din două părți principale: extragerea caracteristicilor de intrare și modelul clasificator. Fiecare din modulele care alcătuiesc aceste părți reprezintă decizii arhitecturale a căror motivație teoretică și practică uremaza să fie descrisă în secțiunile următoare. Deoarece extragerea caracteristicilor de intrare este realizată atât prin tehnica "hand-crafted" cât și "end-to-end", aspectele teoretice folosite de ambele metode vor fi descrise. Cu toate acestea, arhitectura propsă este cea "end-to-end", ilustrată în Fig. 2.2.

3.1 Machine learning și rețele neuronale artificiale

Machine Learning este domeniul de studiu care oferă calculatoarelor abilitatea de a învăța o sarcină fără să fie explicit programate pentru a face acest lucru. " [Arthur Samule, 1959]

Era în care ne aflăm este des numită "big data era", sugerând cantitatea imensă de informații pe care omenirea o deține pentru prima dată în istorie. În același timp, tehnologia a evoluat până în punctul în care putem să creăm alogritmi care să se folosească de aceste informații pentru a rezolva sarcini care păreau până acum imposibil de rezolvat pentru algorimtii generici. Machine Learning este domeniul reprezentativ pentru acest tip de algoritmi, alcătuind un set de metode care pot detecta relațiile din interiorul datelor de intrare, și să se folosească de aceste relații învățate pentru a face predicții pe seturi de date noi.

Machine Learning este structurat în patru mari tipuri de algoritmi: supervizați, nesupervizati, semi-supervizați și așa numiții "reinforcement learning algorithms". Proiectul meu este orientat pe găsirea unei soluții pentru recunoașterea emoției din vorbire. Această recunoaștere se numește în termeni tehnici *clasificare*, și face parte din ramură algoritmilor supervizați pe care urmează să mă axez în continuare.

Algoritmii supervizați trebuie să facă o mapare de la un număr de întrări X la un număr de ieșiri y după ce au procesat un set de perechi $D = (x_i, y_i)_{i=1}^{N}$, numit set de date de antrenare.

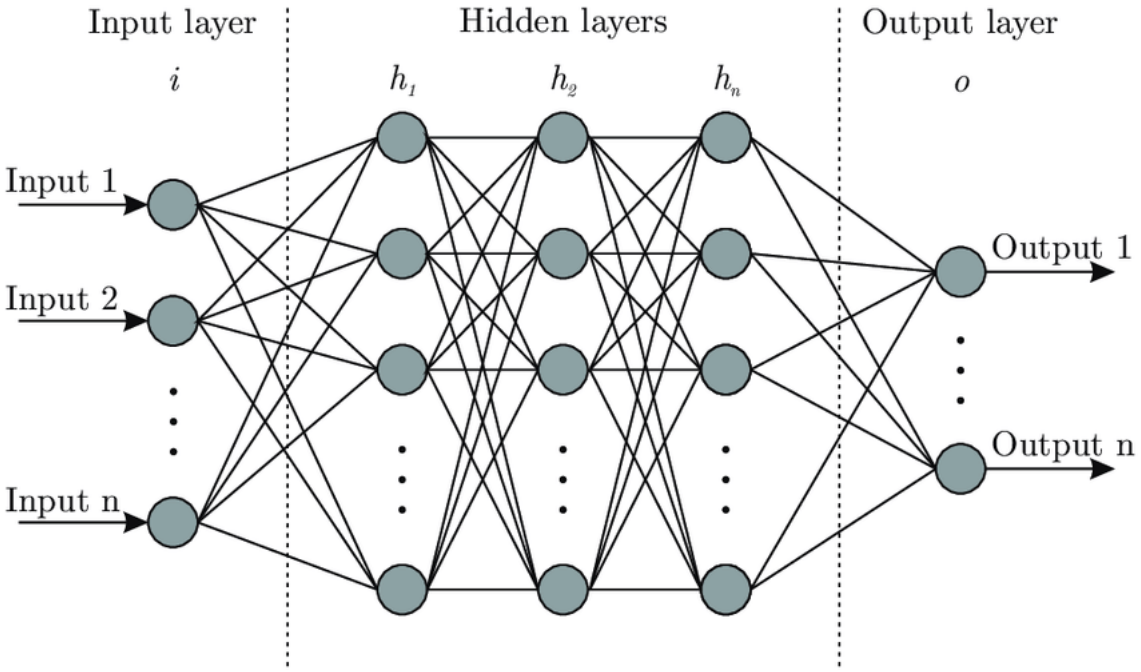


Figura 3.1: SStructura internă a unei rețele neuronale dense. Fiecare nod este conectat la toate nodurile din nivelele adiacente, iar între nivelul de intrare și ieșire există un număr arbitrar de nivele "ascunse". Figura aparține articolului Facundo et al. (2017) [49]

Algoritmii supervizați sunt ghidați astfel să găsească o soluție pentru o anumită problema printr-un set de exemple prin care li se prezintă un set de intrări și rezultatele dorite , conexiunea dintre intrare și ieșire rămânând să fie determinată de aceștia printr-un proces de învățare din greșeli.

Termenul de "deep learning" se referă la o sub-diviziune a domeniului Machine Learning, fiind specializată pe folosirea rețelelor neuronale profunde. Aceste rețele neuronale sunt denumite profunde deoarece reprezintă un set de funcții, nivele, aranjate într-un lanț aciclic, Fig. 3.1. Primul nivel se numește de obicei nivelul de intrare, nivelele din interiorul acestui lanț se numesc "ascunse", iar nivelul de final se numește nivel de ieșire. După cum se poate observa în Fig. 3.1 fiecare din nivele unei rețele neuronale este alcătuit la rândul lui dintr-un număr de neuroni. Arhitectura unei rețele neuronale este bazată pe conexiunile neuronale ale creierului, unde activarea unui neuron poate determina activarea unui alt neuron la care este conectat, obținându-se astfel o reacție în lanț care are ca scop final realizarea unor procese complexe.

Neuronii unei rețele neuronale artificale generice sunt alcătuiți dintr-un set de ponderi care odată înmulțite cu fiecare intrare și adunate cu un coeficent, numit "bias", generează la rândul lor o intrare pentru neuronii următori. Ieșirea fiecărui neuron este determinată după formula următoare:

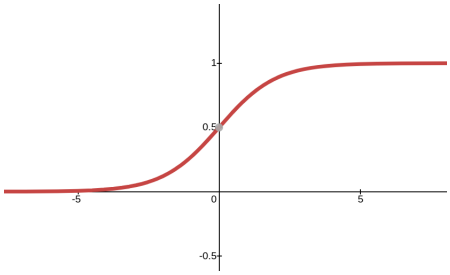
$$a_j^{(i)} = \sum_{k=1}^{M} w_{jk}^{(i)} + w_{j0}^{(i)} \tag{3.1}$$

unde $a_j^{(i)}$ reprezintă activarea neuronului j din nivelul i, iar $w_{jk}^{(i)}$ și $w_{j0}^{(i)}$ reprezintă ponderile respectiv coeficientul "bias" al acelui neuron.

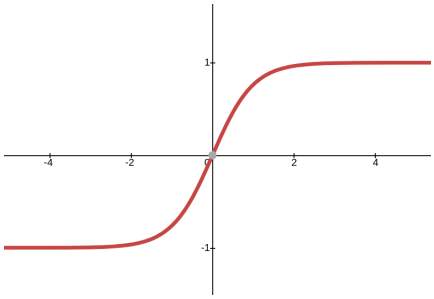
Deși prin conectarea unor astfel de neuroni se pot obține arhitecturi complicate, pentru ca o rețea neuronală să se poată să aproximeze funcții complexe fiecare din ieșirile acestor neuroni trebuie să fie urmate de o așa numită *funcție de activare*. Această funcție de activare are rolul de a elimina liniaritatea din interiorul rețelei neuronale și să mărească astfel numărul gradelor de libertate. Fiecare nivel poate fi reprezentat printr-o matrice iar fiecare trecere prin acel nivel poate fi exprimată printr-o înmulțire matricială. Fără existența acestor funcții de activare rețeaua

neruonala ar fi doar un set de înmulțiri matriciale, care la rândul lui poate fi reprezentat ca o matrice unică, eliminând în sine sensul rețelelor adânci. Din acest motiv activarea unui neuron devine $h_j^{(i)} = z(a_j^{(i)})$, unde z este funcția de activare aleasă.
Unele dintre cele mai populare funcții de activare sunt:

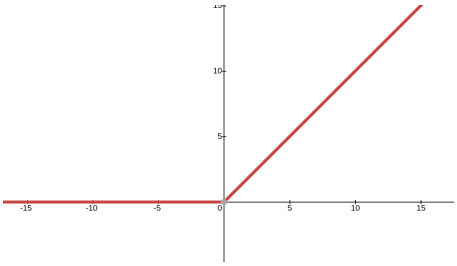
- *Sigmoid*, $f(x) = \frac{1}{1+e^{-x}}$, care aduce valoarea lui x in intervalul (0, 1)



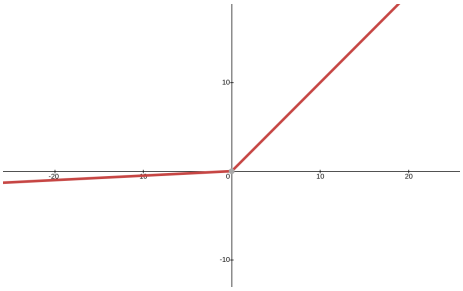
- *Tangenta hiperbolică*, $f(x) = \frac{(e^x - e^{-x})}{(e^x + e^{-x})}$, care aduce valoarea lui x in intervalul (-1, 1)



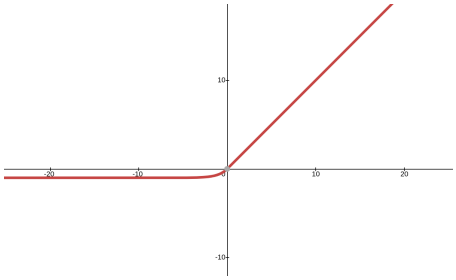
- *ReLu*, $f(x) = \max(0, x)$, care aduce valoarea lui x in intervalul [0, inf)



- *Leaky ReLu*, $f(x) = \max(0.01x, x)$, care aduce valoarea lui x in intervalul $(-\inf, \inf)$



- *ELu*, $f(x) = \max(\alpha(e^x - 1), x)$, care aduce valoarea lui x in intervalul $(-\inf, \inf)$



În funcție de problema pe care dorim să o rezolvăm, nivelul de ieșire poate să conțină la rândul lui o astfel de funcție de activare. În cazul clasificării binare este folosită funcția sigmoid iar în cazul clasifiarii unui număr mai mare de clase de obicei se folosește funcția "cross-entropy". Pentru regresie, sau alte probleme care nu clasifică datele de intrare, nivelul final nu este urmat de o astfel de funcție.

Antrenarea rețelelor neuronale se realizează prin folosirea unor algoritmi numiți optimizatori. Aceștia determină influența pe care fiecare pondere din rețea a avut-o asupra rezultatului final extrăgând derivata ponderii în raport cu eroarea totală a rețelei. Dacă acea influență a fost mare ponderea este modificată puternic, în caz contrar aceasta este modificat puțin. Prin folosirea unui număr mare de astfel de exemple ponderile ajung să conveargă la un set de valori care permit aproximarea unei multitudini de funcții complexe.

Funcția de calcul a erorii poate avea forme diferite, dar una din cele mai folosite este eroarea pătratică medie 3.2.

$$E(W) = \frac{1}{2}\sum_{n=1}^{N}(y(x_n, W) - t_n)^2 \tag{3.2}$$

unde $E(W)$ este eroarea calculată, N reprezintă numărul total de exemple de antrenare, $y(x_n, W)$ este predicția pentru exemplul cu numărul n iar t_n este valoarea adevărată.

Recunoașterea emoției în vorbire este o problemă de clasificare. În acest caz, $y(x_n, W)$ reprezintă o distribuție de probabilitate asupra emoțiilor estimate de model iar t_n este un vector de forma $[0, 1, 0, 0]$, în care valoarea 1 este pe poziția emoției aferente acelui exemplu.

În arhitectura acestui proiect am folosit mai multe tipuri de rețele neuronale pentru îndeplinirea mai multor funcționalități. Aceste tipologii și modul lor de utilizare urmează să fie descrise în detaliu în sub-capitolele următoare.

3.2 Transformata Fourier discretă

Orice undă, indiferent dacă e observată în univers sau mâzgălită de noi pe foaie, reprezintă de fapt doar o sumă de funcții sinusoidale de diferite frecvențe.

Un semnal audio este un semnal complex alcătuit din mai multe unde de diferite frecvențe care circulă sub formă unor perturbații prin mediu. Atunci când înregistrăm un semnal audio capturăm doar amplitudinile rezultate în urma combinării acestor unde. Transformata Fourier este un concept matematic care ne permite să descompunem un semnal în frecvențele care îl compun și magnitudinile acestora.

Motivul folosirii transformatei Fourier vine de la studiul seriilor Fourier. Prin studiul acestor serii, funcții periodice complicate sunt scrise ca simple sume de unde reprezentate matematic prin funcțiile sinus și cosinus.

Fie o funcție periodică $f(t)$, cu o perioada fundamentală T, aceasta are aferentă seria Fourier următoare:

$$g(t) = a_0 + \sum_{m=1}^{\infty} a_m \cos(\frac{2\pi mt}{T}) + \sum_{n=1}^{\infty} b_n \sin(\frac{2\pi nt}{T}) \tag{3.3}$$

unde a_0, a_m si b_n sunt coeficienții Fourier calculați sub forma

$$\begin{aligned} a_0 &= \frac{1}{T}\int_0^T f(t)dt \\ a_m &= \frac{2}{T}\int_0^T f(t)\cos(\frac{2\pi mt}{T})dt \\ b_n &= \frac{2}{T}\int_0^T f(t)\sin(\frac{2\pi nt}{T})dt \end{aligned} \tag{3.4}$$

Aceste relații pot fi scrise într-o formă mai elegantă din punct de vedere matematic prin folosirea numerelor complexe și a formulei lui Euler, $e^{2\pi i\theta} = \cos(2\pi\theta) + i\sin(2\pi\theta)$.

Astfel,

$$g(t) = \sum_{n=-\infty}^{\infty} c_n e^{i\frac{2\pi nt}{T}} \tag{3.5}$$

unde c_n reprezintă echivalentul coeficientiilor Fourier a_m și b_n din 3.4

$$c_n = \frac{1}{T}\int_0^T f(t)e^{-i\frac{2\pi nt}{T}}dt$$

Transformata Fourier este o operație care se aplică unei funcții complexe și produce o altă funcție complexă care conține aceeași informație ca funcția originală, dar reorganizată după frecvențele componente. Transforamta Fourier este o generalizare a seriilor Fourier complexe prezentate în 3.5, putând fi văzută că limita seriei Fourier când perioada tinde la infinit.

Fie $f : R \to C$ absolut integrabilă. Funcția complexă de variabilă reală $F : R \to C$,

$$F(\omega) = \int_{-\infty}^{\infty} e^{-2\pi i\omega t} f(t)dt$$

se numește transformată Fourier a funcției f, iar

$$f(t) = \int_{-\infty}^{\infty} e^{2\pi i\omega t} F(\omega)d\alpha$$

se numește inversa tranformatei Fourier.

Astfel transformata Fourier reprezintă o unealtă care ne permite să schimbăm unghiul din care privim diferitele semnale de tip undă, făcând posibilă trecerea din domeniul temporal în domeniul frecvență Fig.3.2. Acest domeniu ne permite să diferențiem între semnalele de diferite frecvențe care alcătuiesc unda. Prin folosirea inversei transformatei Fourier, putem chiar să eliminăm anumite semnale prin modificarea coeficientiilor aferenți frecvențelor acestora. Acest concept a deschis granitile unei arii uriașe de aplicații ale acestei teoreme, fiind revoluționară pentru domeniul procesării semnalelor.

"Short-time Fourier transform", sau STFT, este o varianta a transformatei Fourier prezentată mai sus, folosită pentru a determina frecventele sinusoidale și magnitudinile unor secțiuni locale dintr-un semnal. În practică, procedura prin care se calculează funcțiile STFT este de a împărții semnalul temporal în segmente mai mici de dimensiuni egale și apoi de a calcula transformată Fourier pe fiecare din aceste segmente separat.

$$STFT(r,\omega) = \int_{-\infty}^{\infty} f(t)w(t-r)e^{-i2\pi\omega t}dt$$

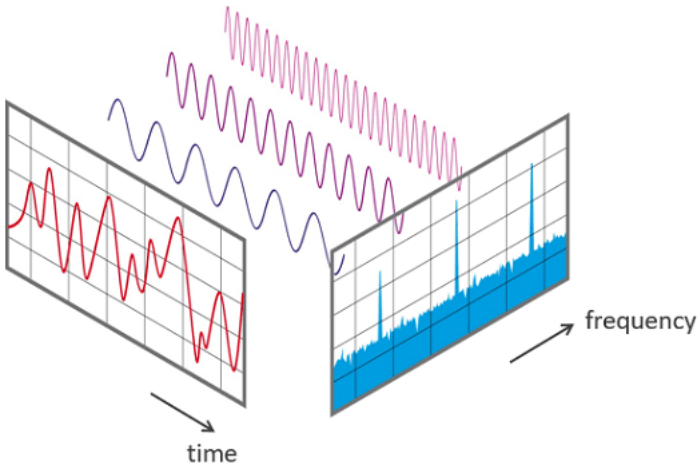


Figura 3.2: Ilustrare grafică a translatării semnalului audio în domeniul frecvență după aplicarea transformatei Fourier. Figura aparține paginii web din Trekhleb, 2018 [50].

unde w(r) este o funcție fereastră, definită la 4.1.2.

STFT oferă informații asupra frecvenței pe un anumit interval temporal local, fiind folosită în situații în care frecvențele componente ale unui semnal variază puternic în timp. Transformata Fourier în schimb oferă doar media informațiilor frecvențelor pe întregul interval de timp al semnalului [51].

Deoarece semnalul audio este discretizat si segmentat, in procesarea acestuia se foloseste varianta discreta a STFT.

$$STFT(m,\omega) = \sum_{n=-\infty}^{\infty} f[n]w[n-m]\mathrm{e}^{-i2\pi\omega n} \tag{3.6}$$

3.3 Caracteristicile Hand-crafted

Metoda "hand-crafted" de extragere a caracteristicilor de intrare din semnalul audio este bazată pe diferite formule matematice, unele dintre acestea urmând sa fie descrise mai jos. După cum am spus și în sub-capitolul 1.3, momentan nu există un set de caracteristici reprezentative pentru informația emoțională, iar majoritatea implementărilor aleg aceste caracteristici printr-o selecție empirică.

Aceste caracteristici sunt preluate din problema recunoașterii informației lingivstice din vorbire, "Speech Recognition" si chiar dacă alegerea setului de caracteristici este subiectivă fiecărei implementări, majoritatea împărtășesc un sub-set comun care se consideră că surprinde anumite informații importante din discursul uman. Coeficientii Mel cepstrali și funcțiile delta și delta-delta ale acestora fac de obicei parte din datele de intrare a sistemelor SER "hand-crafted" și din cauza popularitatii acestora urmează să le descriu în continuare.

3.3.1 Coeficientii Mel cepstrali

Coeficienții Mel cepstrali, sau MFCC, sunt cea mai folosită reprezentare a proprietăților spectrale ale semnalelor vocale. Aceștia funcționează cel mai bine în cazul domeniului "Speech Recognition" deoarece iau în calcul senzitivitatea percepției umane în interpretarea frecvențelor

prezente în semnalul audio. Pentru calcularea acestor coeficienți, trebuie să urmăm o serie de pași [52].

1. În primul rând se calculează o estimare a densității spectrale a semnalului, așa numite "Periodogram estimate of the power spectrum". Pentru calcularea acesteia se aplică transformata Fourier pe termen scurt, ecuația 3.6, pe fiecare segment din semnalul audio. Odată obținut echivalentul semnalului audio în domeniul frecvență , numit "complex spectrum", se ridică la pătrat valoarea absolută a acesteia pentru obținerea "Periodogram"-ei.

$$P_i(k) = \frac{1}{N} \mid STFT(i,k) \mid^2 \qquad ,1 \leq k \leq K$$

   unde i este numărul segmentului actual, N este numărul total de segmente audio și K este frecvența maximă, sau "lungimea transformatei Fourier discrete".

2. Următorul pas este aplicarea așa-ziselor "Mel filerbanks". "Mel filerbanks" sunt filtre triunghiulare care au valori nule pe majoritatea lungimii spectrum-ului. Aceste filtre se calculează prin scalarea unor frecvențe alese la distanțe egale dintr-un interval, de obicei $[300Hz, 8000Hz]$, în scară Mel. Această scară reprezintă limitele capacității umane de percepere a diferitelor frecvente sonore. Scalarea frecvențelor se realizează prin formula $M(f) = 1125\ln(1 + f/700)$. Oamenii sunt mult mai buni în a diferenția schimbări la frecvente mici față de frecvente înalte, iar prin încorporarea aceste scalări caracteristicile obținute vor fi mult mai apropiate de ce ar auzi defapt un om.
   După ce s-a aplicat complet scalare Mel se crează filtrele triunghiulare folosint următoarea formula

$$H_m(k) = \begin{cases} 0 & k < f(m-1) \\ \frac{k - f(m-1)}{f(m) - f(m-1)} & f(m-1) \leq k \leq f(m) \\ \frac{f(m+1) - k}{f(m+1) - f(m)} & f(m) \leq k \leq f(m+1) \\ 0 & k > f(m+1) \end{cases}$$

   unde m este numărul frecvenței scalate, iar k este frecvența curentă din spectrum.
   Filtrele triunghiulare sunt înmulțite apoi cu "power spectrum"-ul obținut la pasul anterior și se obțin astfel amplitudinile din fiecare filtru Mel.

3. Următorul pas reprezintă logaritmarea acestor amplitudini.

4. Ultimul pas fiind aplicarea transformatei cosinus discretă (DCT). Motivația acestui pas este că energiile obținute anterior au un grad ridicat de corelație iar aplicarea DCT realizează decorelara lor și oferă reprezentarea compresată a acestora. Formula pentru DCT fiind,

$$X_k = \frac{1}{2}(x_0 + (-1)^k x_{N-1}) + \sum_{n=1}^{N-2} x_n cos[\frac{\pi}{N-1} nk]$$

   Amplitudinile spectrum-ului rezultat reprezintă coeficienții Mel cepstrali.

3.3.2 Deltas si delta-deltas

Totuși, coeficienții Mel descriu doar coperta puterii spectrale a fiecărui segment din semnalul audio. S-a demonstrat empiric că o parte importantă din informația vocală se află și în spatele dinamicii acestor coeficienți, modul in care aceștia evoluează in timp. Acest lucru se calculează folosind caracteristicile delta și delta-deltas.

Coeficienții deltas sunt calculați din MFCC prin următoarea formulă:

$$d_t = \frac{\sum_{n=1}^{N} n(c_{t+n} - ct - n)}{2\sum_{n=1}^{N} n^2}$$

unde d_t reprezintă coeficientul delta, din segmentul t în funcție de coeficienții statici c_{t+n} și c_{t-n}. Coeficienții delta-delta sunt calculați prin aceeași formulă, înlocuindu-se doar coeficentii MFCC cu cei delta.

Pe lângă acești coeficienți, caracteristicile "hand-crafted" pot conține și următoarele: tonalitate, probabilitate vocală, radicalul mediei amplitudinii la pătrat, rata zero-crossing, chroma, coeficientul rollof, rația harmonics-to-noise, bruiajul, media, variația, minimul și maximul semnalului audio etc. Toate acestea se calculează cu funcții matematice predefinite asemănătoare cu procesul de obținere a coeficentiilor MFCC.

În implementarea extragerii caracteristicilor "hand-crafted" a acestei lucrări am folosit: MFCC, delta, delta-deltas, radicalul mediei la pătrat a amplitudinii fiecărui segment, rata zero-crossing, chroma și coeficientul rollof.

3.4 Extragerea caracteristicilor End-to-end

Avansările recente ale algoritmilor și a hardware-ului calculatoarelor au făcut posibilă antrenarea rețelelor neuronale într-o manieră end-to-end pentru sarcini care necesitau în trecut expertiza umană. Pe lângă că aceste arhitecturi de rețele neuronale solicită mai puțin efort uman decât abordările tradiționale, în general obțin și performanțe superioare. Acest aspect este în special adevărat atunci când cantitatea de date pentru antrenare disponibilă este mare, deoarece beneficiile optimizării holistice, minimizarea erorii detecției emoțiilor, tinde să le depășească pe acelea ale cunoștințelor anterioare reflectate in funcțiile matematice al metodei "hand-crafted" [30].

Arhitectura "end-to-end" este prezentată în Fig. 2.2. După cum poate fi observat, extragerea caracteristicilor în această arhitectură este realizată prin atașarea unei rețele convoluționale combinată cu o tehnică de normalizare între fiecare nivel. Aceste rețele convoluționale sunt celebre pentru procesarea imaginilor pentru că, spre deosebire de rețelele neuroanle artificiale normale, se folosesc de relațiile spațiale bidimensionale dintre pixeli. Astfel pentru a ne folosi de acest avantaj se extrage spectograma Mel a semnalului audio, care va fi folosită ca date de intrare pentru aceste rețele. Un exemplu al unei astfel de spectograme se poate observa în Fig. 3.3.

3.4.1 Spectograma Mel

Prin spectograma se înțelege reprezentarea vizuală al unui spectrum de frecvențe al unui semnal în raport cu timpul. Astfel într-o spectogramă axa orizontală reprezintă timpul, axa verticală reprezintă frecvența iar intensitatea culorii reprezintă magnitudinrea(amplitudinea) frecvenței observate la acel moment din timp. Calcularea spectrogrameleor Mel este extrem de asemănătoare cu calcularea coeficientilor de corelați Mel prezentați în sub-sub-capitolul 3.1.2. Astfel pașii 1, 2, și 3, prezentați anterior, se reproduc. Singura diferență este că în loc de calcularea transformatei cosinus discrete și extragerea coeficientilor ca la pasul 4, evoluția valorilor amplitudinii frecventelor în funcție de timp sunt translatate într-o matrice care va alcătuii spectograma.

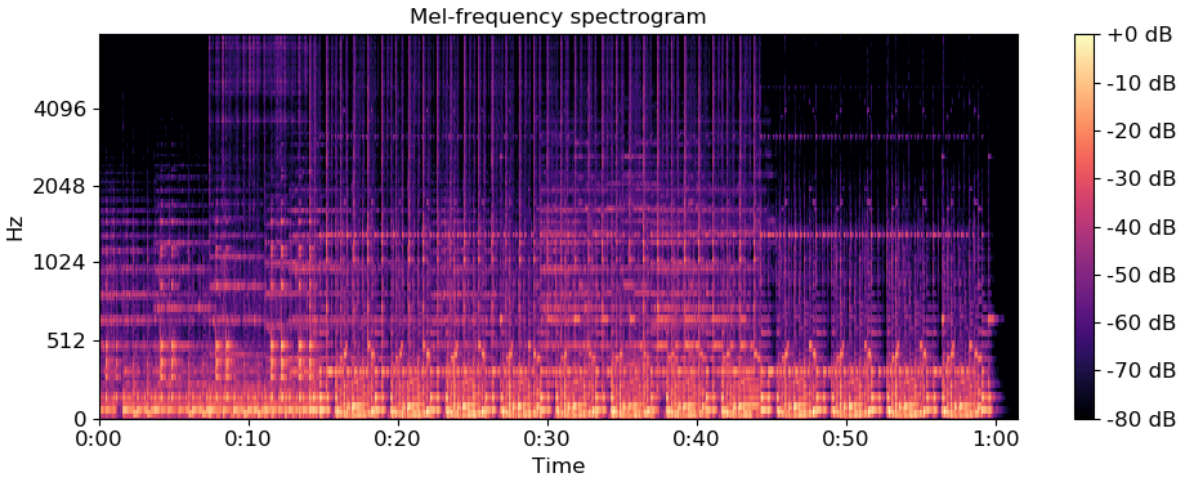


Figura 3.3: Spectograma Mel a unui semnal audio extras cu ajutor librărie librosa [53]. Spectograma ilustrează valoarea amplitudinilor frecventelor din fiecare moment al semnalului audio.

3.4.2 Batch normalization

Unul din motivele pentru care antrenarea rețelelor neuronale profunde, cu mai multe nivele, devine dificilă este faptul că distribuția intrărilor fiecărui nivel se schimbă în timpul antrenării prin modificarea parametrilor nivelului anterior care le generează. Acest lucru încetinește drastic procesul de învățare al modelului deoarece alegerea modului de inițializare al parametrilor dobândește un impact mult mai ridicat și face necesară alegerea unor rate de antrenare mai mici. Acest fenomen se numește "internal covariate shift", iar o soluție propusă de Ioffe & Szegedy et al., 2015 [54], este normalizarea intrărilor fiecărui nivel.

Formula pentru normalizare este:

$$x_i' = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \varepsilon}}$$

unde x_i reprezintă intrarea cu numărul i, μ este media setului de intrări al nivelului curent, σ este deviația standard a acestor intrări iar ϵ reprezintă doar o constanta pentru evitarea împărțirii la 0.

Pe lângă reducerea fenomenului prezentat anterior, "internal covaraite shift", aceast mecanism de normalizare aduce alte două mari beneficii. În timpul antrenării, gradienții ponderilor niveleor circulă de la nivelul final al rețelei spre început, modificând parametrii modelului pentru a reduce eroarea înregistrată. Prin folosirea normalizării înainte de fiecare nivel, se reduce dependența acestor gradienți de mărimea ponderilor nivelelor rețelei sau de valoarea lor inițială. Acest lucru ne permite să folosim rate de antrenare mai mari fără a pune în pericol convergența modelului.

Cel de al doilea avantaj este că s-a demonstrat empiric, [54], faptul că tehnica "batch normalization" îmbunătățește regularizarea parametrilor, limitarea creșterilor acestora, și mărirea generalității modelului.

Modele SER sunt succeptibile la un fenomen denumit "explozia gradienților", deoarece folsoesc arhitecturi complexe cu un număr mare de parametrii. Pe lângă acest considerent, acuratețea acestor modele depinde puternic de variabile greu de controlat precum modul de înregistrare al bazelor de date, locația unde s-a desfășurat înregistrarea, multitudinea vorbitorilor, diferențele de exprimare și limbile folosite. "Batch normalizaton", oferă beneficii în ambele privințe deoarce funcționează că un regularizator, încetinind "exploziile" din interiorul nivelelor, mărirea excesivă

a parametrilor, și normalizează datele de intrare reducând astfel influența diferențelor de la o înregistrare la alta.

3.4.3 Rețele neurononale convoluționale

Rețelele neuronale convoluționale, sau CNN, reprezintă o adaptare a rețelelor neuronale artificiale generice pentru rezolvarea sarcinilor vizuale, sau în care datele de intrare sunt organizate în două dimensiuni. Deși aceste tipuri de rețele neuronale au fost introduse prima dată în 1989 ((Le Cun et al., 1989 [55]), au crescut în popularitate abia în ultimii ani, dominând astăzi majoritatea soluțiilor propuse pentru diferitele sarcini din domeniul vizual.

Conceptul principal din spatele acestor arhitecturi reprezintă operația matematică numită *convoluție*, după care a și fost numită această tipologie de rețea neuronală. Convoluția reprezintă o operație care primește ca intrări două funcții și prezintă modul în care una dintre funcții își modifică forma în funcție de cea de a două funcție. Formula unei convolutii discrete este următoarea:

$$s(t) = (f * g)(t) = \sum_{n=-\infty}^{\infty} f(n) \cdot g(t-n)$$

unde s reprezinta convoluția funcțiilor f si g. În doua dimensiuni această forumlă devine

$$S(i,j) = (I * K)(i,j) = \sum_{m} \sum_{n} K(m,n) \cdot I(i-m, j-n) \tag{3.7}$$

unde S este convoluția rezultată prin combinarea funcțiilor bidimensionale I si K.

Bazată pe această formulă matematică, rețele neuronale convoluționale au reușit să depășească unele dintre principalele piedici pe care rețelele neuronale simple le întâlneau în procesarea imaginilor. Rețele neuronale tradiționale crează conexiuni între fiecare unitate de intrare și fiecare neuron din nivelul aferent. Având un număr satisfăcător de date de intrare acestea pot să învețe să rezolve diferite sarcini care par imposibile pentru alogiritmii generici din știința calculatoarelor. Totuși, dimensiuniea acestor arhitecturi poate crește extrem de rapid, iar pentru probleme ca procesarea imaginilor, unde diferite imagini pot avea sute de mii de pixeli, depășesc rapid capacitățile de procesare disponibile.

Pe lângă acest dezavantaj, arhitecturile generice nu reușesc să suprindă esența procesării de imagini, sau a mecanismului de vedere uman, care este bazat pe conceptul că pixelii care se află în vecinătatea unui anumit pixel sunt mult mai puternic corelați cu acesta decât oricare alți pixeli. Astfel deși aceste nivele din rețelele neuronale generice se folosesc de întreaga informație de intrare rămân inflexibile când vine vorba de a determina anumite linii, curbe sau forme ale obiectelor observate, ele luând în considerare doar intensitatea culorii unui pixel dintr-o locație dată.

Rețelele neuronale convoluționale reușesc să treacă peste aceste obstacole prin folosirea a trei idei care au adus multe beneficii și în alte tipologii de rețele neuronale, *filtre de recepție locale*, *refolosirea parametriilor* și *subesantionare*.

Metodele tradiționale de antrenare folosesc înmulțirea matricială între întregul set de date de intrare și întregul set de parametrii dintr-un nivel. În schimb, CNN se folsoesc de așa numite *filtre*, sau *"kernels"*, care reprezintă un set de paramterii de dimensiuni de câteva ori mai mici decât datele de intrare. Aceste filtre sunt apoi înmulțite doar cu anumite parți echivalente dimensional din datele de intrare. Astfel, de exmplu având o imagine în dimensiuni 28x28 și un filtru de dimensiuni 3x3, acest filtru este aplicat pe bucăți de dimensiuni 3x3 din matricea intrărilor plimbându-l pe întreagă suprafață a imaginii. Rezultatul obținut devine una din intrările unui filtru următor. Acest mecanism poate fi observat în Fig. 3.4.

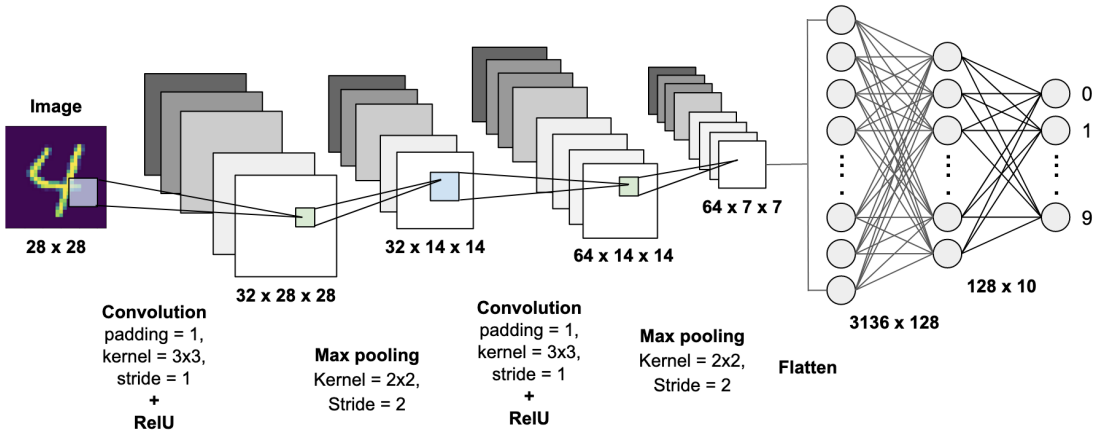


Figura 3.4: Arhitectura unei rețele neuronale convoluționale specialziata pe recunoașterea cifrelor scrise de mâna. Figura aparține paginii din Patel, 2019 [56]

Simpla folosire a acestor filtre de dimensiuni reduse rezolva majoritatea problemlor impuse de arhitecturile tradiționale. În primul rând prin acest mod rețeaua neuronală primește capacitatea de a determina caracteristici abstracte. Acest lucru este posibil prin faptul că filtrele se focusează pe determinarea relațiilor din pixelii învecinați. În timpul antrenarii aceste filtre vor învață să detemine anumite caracteristici din interiorul imaginilor. Deoarece aceștia sunt aplicați succesiv pe intreagă imagine de intrare, ei o să determine existența și poziția acelor caracteristici oriunde în matricea de intrare. Nivelele de la baza rețelei învață să determine caracteristici simple că diferite puncte, linii sau curbe din diferite locații ale imaginii, în timp ce nivelele de la capătul rețelei prin combinarea informației de la filtrele anterioare ajung să determine carcateristici complexe, ca diferite forme sau obiecte. Operația de convoluie, 3.7, prezintă perfect modul de aplicare al acestor filtre unde I reprezintă imaginea iar K filtrul aplicat pe fiecare porțiune din acea imagine.

Folisrea acestor filtre aduce și un mare avantaj din punct de vedere computațional. Deoarece între nivelele rețelei și intrările acestora nu mai există conexiuni între fiecare unitate și deoarece o pondere, "weight", nu este înmulțită doar cu un singur element de intrare ci este refolosit de un filtru pentru întreagă imagine, numărul parametriilor folosiți scade considerabil. Acest lucru reduce cerințele de memorie, crește eficiență statistică și ne permite să folosim arhitecturi mai complexe în numărul de filtre aplicate.

Odată ce aplicarea filtrelor s-a finalizat, rezultatul este de obicei transformat prin folosirea unei funcții de subeșantionare. Subeșantionarea se realizează cu ajutorul unor operații matematice ca maximul, media, mediana etc., se realizează extragerea unui singur element dintr-un grup de pixeli învecinați, Fig 3.5 . Dacă este folosită funcția de maxim acest mecanismt se numește "Max-pooling".

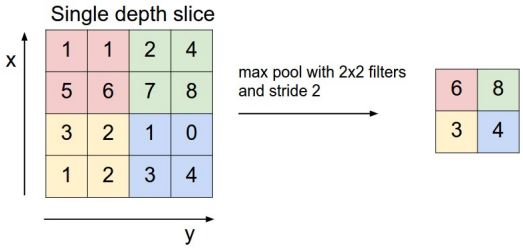


Figura 3.5: Exemplificare a mecanismului de subeșantionare bazat pe funcția maxim. Figura aparține notitelor din cursul CS231n Stanford [57].

Unul dintre beneficiile acestei tehnici, pe lângă reducerea resurselor de memorie și de procesare necesare, este reducerea variației ieșirilor la mici inflexiuni ale datelelor de intrare. Această tehnică încearcă să asigure faptul că modificări care nu ar trebui să influențeze rezultatul nu o să interfereze cu procesul de învățare.

În crearea unei arhitecturi de rețea neuronală convolutionala, principalii parametrii fixați, pe lângă numărul de nivele, sunt dimensiunea filtrelor pentru fiecare nivel, dimensiunea pasului cu care aceste filtre străbat imaginea și dimensiunea și modul "padding"-ului de la marginea imaginii. "Padding"-ul reprezintă adăugarea unui număr de rânduri sau coloane de diferite valori (de ex 0, sau copii ale anumitor rânduri sau coloane) la marginea imaginii pentru a ne asigura că rezoluția imaginii este divizibilă cu lungimea și lățimea filtrului aplicat.

3.5 Rețele neuronale recurente

Rețelele neuronale recurente, ca și cele convoluționale, sunt specializate pe procearea informației bidimensionale. Cu toate acestea diferența apare în faptul că pentru rețele recurente a două dimensiune nu mai este spațială ci temporală. Aceste rețele neuronale primesc datele de intrare într-o manieră secvențială și reușesc să depisteze relațiile temporale dintre două sau mai multe segmente de date aflate la momente de timp diferite.

Arhitectura acestei tipologii de rețele neuronale se bazează pe determinarea stării curente a sistemului și transmiterii acesteia celulei aflate la momentul de timp următor. Secretul din spatele rețelelor recurente stă în folosirea aceluiași set de parametrii pentru celulele de la fiecare moment. Astfel având date de intrare organizate secvențial în funcție de timp, $x^{(t)}$, celula unei rețele neuronale recurente folosește o așa zisă "celulă de memorie" pentru a salva starea din momentul t, $h^{(t)}$, și a o transimte ca o a două intrare momentului $t+1$. Folosirea unor parametrii comuni atât pentru procesarea intrărilor $x^{(t)}$ și $h^{(t-1)}$ la momentul t cât și a intrarlor $x^{(t+1)}$ și $h^{(t)}$ la $t+1$ permite rețelei neruonale să determine relațiile temporale între aceste două momente diferite și astfel să ia decizii în prezent bazate și pe informațiile din trecut, Fig. 3.6.

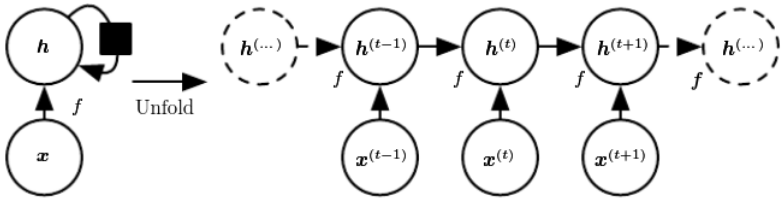


Figura 3.6: Derularea celulei recurente în timp. Figura aparține Goodfellow et al., 2016 [58].

Celula de memorie este o denumire fictivă, ea referindu-se doar la mecanismul de prezervare a stării curente și transmiterii ei la celula recurentă dintr-un moment viitor. Acest lucru se realizează prin folosirea mai multor seturi de parametrii. Într-o rețea neuronală normală fiecare celulă, neuron, este alcătuită dintr-un singur set de parametrii, ponderi, care se înmulțesc matricial cu datele de intrare rezultând astfel gradul de activare al acelui neuron. Formula care stă la baza calculelor din interiorul fiecărui neuron pentru o rețea neuronală simplă este $Y = W^T * X + b$, unde Y este "activarea" neuronului iar W și b reprezintă parametrii antrenați. Dacă un nivel dintr-o rețea neuronală artificială tradițională conține mai mulți neuroni, un nivel al unei rețele recurente este alcătuită dintr-o singură celulă distribuită pe mai multe puncte de timp. Spre deosibire de neuronul unei rețele generice, celula RNN trebuie să realizeze mai multe sarcini. Pe lângă

extragerea informațiilor din datele de intrade curente, de la momentul t, celula recurentă trebuie să proceseze și starea de la momentul $t-1$ inanite de a determina ieșirea $y^{(t)}$ și starea curentă $h^{(t)}$. Din acest motiv pentru fiecare dintre aceste sarcini diferite se alocă un set de ponderii separat, de exemplu U, W, V din Fig. 3.7.

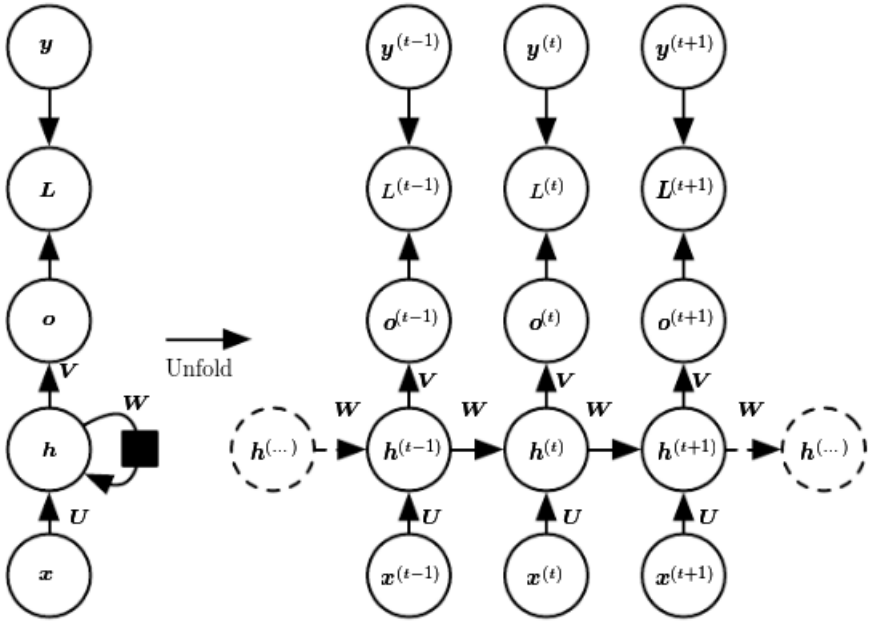


Figura 3.7: Desfășurarea procesului de antrenare pe intervalul de timp al datelor de intrare. La fiecare moement t se calculează o eroare $L^{(t)}$ că va genera modificarea tuturor ponderilor din t și momentele predecesoare care au contribuit la această. Figura aparține Goodfellow et al., 2016 [58].

Deoarece o celulă recurentă folosește un număr mai mare de parametrii, computațiile din interiorul acesteia devin mult mai complicate. Pentru o rețea neruonala recurentă generică o celulă poate fi descrisă prin următoarele ecuații:

$$a^{(t)} = b + Wh^{(t-1)} + Ux^{(t)} \tag{3.8}$$

$$h^{(t)} = \tanh\left(a^{(t)}\right) \tag{3.9}$$

$$o^{(t)} = c + Vh^{(t)} \tag{3.10}$$

unde $a^{(t)}$ și $h^{(t)}$ reprezintă starea internă a celulei înainte și după aplicarea unei funcții de activare, $o^{(t)}$ reprezintă valoarea la ieșirea celulei, U reprezintă ponderile datelor de intrare, W reprezintă ponderile aferente stării precedente a rețelei și V ponderile de la ieșirea celulei.

Deoarece arhitectura unei celule recurente folosete seturi diferite de parametrii pentru cele două intrări, stare precedentă și datele de intrare curente, aceasta poate controla atât impactul informațiilor acumulate din trecut cât și importanța datelor noi observate în prezent. Acești parametrii, împreună cu cei aferenți controlului ieșirilor V, sunt modificați în timpul antrenării din două perspective: reducerea erorii ieșirii celulei de la timpul t, $L^{(t)}$, și reducerea erorii adăugate de folosirea stării curente , $h^{(t)}$, în calculele ieșirilor celuleor de la momentele de timp viitoare, $L^{(t+1)}$, $L^{(t+2)}$, etc.

Una din problemele depistate în cazul rețelelor recurente este echilibrarea influenței introduse de stările de la momente de timp îndepărtate comparate cu cele de la momente de timp apropiate. Fiecare celulă recurentă stabilește impactul stariilor precedente printr-o înmulțire matricială cu un

anumit set de parametrii, 3.9. Astfel, ponderea informațiilor unei stări dintr-un trecut îndepărtat poate scădea sau crește exponențial. Dacă neglijăm funcția de activare și "bias"-ul, pentru ușurință notației, se poate demonstra că sistemul de tranziție a informațiilor recurente, stărilor, în timp devine:

$$h^{(t)} = W^T h^{(t-1)} + U^T x^{(t)}$$
$$h^{(t)} = W^T (W^T h^{(t-2)} + U^T x^{(t-1)}) + U^T x^{(t)}$$
$$h^{(t)} = (W^2)^T h^{(t-2)} + W^T U^T x^{(t-1)} + U^T x^{(t)}$$
$$h^{(t)} = (W^t)^T h^{(0)} + (W^{t-1})^T U^T x^{(1)} + \ldots + W^T U^T x^{(t-1)} + U^T x^{(t)}$$

unde h reprezintă starea internă a celulei recurente la un moment t, W reprezintă ponderile stării precedente, x datele de intrare la momentul t iar U ponderile acestor intrări.

Se poate observa cum informațiile obținute la stările anterioare sunt puternic influențate de distanță acestora față de momentul prezent. Deoarece de obicei valorile ponderilor din W sunt initializate cu valori mai mici decât 1, rețelele neuronale recurente tind să uite informațiile din stările incipiente dând o importanță mult mai mare celor din momentele de timp mai apropiate. În multe cazuri acest lucru nu este rezultatul dorit. Un exemplu ar fi crearea unui traducător dintr-o limba A într-o limba B. Sensul unui cuvânt aflat la capătul unei propoziții poate depinde de unul din cuvintele de la începutul propoziției. Dacă traducătorul uită de existența acelui cuvânt traducerea va fi în final una eronată.

Pentru rezolvarea acestei probleme s-au propus noi arhitecturi ale celulelor care alcătuiesc rețelele neuronale recurente. Una din aceste arhitecturi, și cea folosită de mine în acest proiect, este "Long Short-Term memory cell", sau LSTM. Diferența dintr-o celulă LSTM și una recurentă normală este folosirea unor așa numite porți care reușesc să îmbunătățească capacitatea celulei de a controla informațiile păstrate sau uitate în procesul de antrenare recurentă.

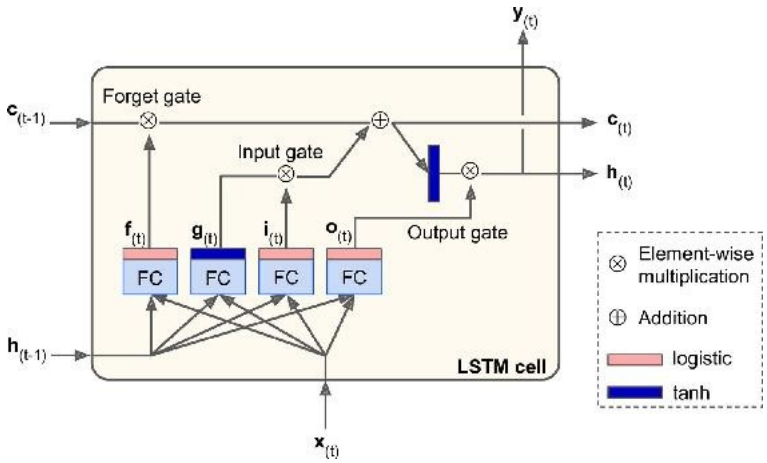


Figura 3.8: Structura internă a unei celul recurente LSTM. Se pot observa conexiunile porților care controlează procesele din interiorul celulei: $f_{(t)}$, poarta de uitare, $g_{(t)}$ nivelul de procesare, $i(t)$ poarta de intrare și $o_{(t)}$ poarta de ieșire. Figura aparține Gron, 2017 [59].

Celulele LSTM folosesc doi vectori pentru memorarea stării celulei, unul pentru memorarea de lungă durată iar cel de al doilea pentru memorarea pe o durata scurtă. În Fig.3.8 putem obesrva arhitectura internă a unei astfel de celule și a porților care determina transferul de informație.

Fiecare din aceste porți sunt specializae în timpul antrenării pentru rezolvarea anumitor sarcini specifice:

1. $g(t)$, reprezintă mecanismul principal de procesare de informații din celulă. Valorile obținute la ieșirea acestei părți din celulă reprezintă valorile candidate care ar putea fi

introduse în starea celuiei. Într-o celulă normală, aceste valori ar trece direct în calcularea stării $h(t)$ și ieșirii $y(t)$, fiind aferentul funcției $a(t)$ din 3.9. Într-o celulă LSTM însă, aceste valori sunt filtrate și abia apoi salvate în starea celulei prin folosirea porților.

2. $f(t)$, este denumită "forget gate", sau poarta de uitare, care controlează care părți din memoria de lungă durată a celulei va fi uitată.

3. $i(t)$, este denumită "input gate" sau "update gate", și decide ce valori din $g(t)$ ar trebui să fie adăugate la memoria de lungă durată.

4. $o(t)$, este denumită "output gate", și controlează care părți din memoria de lungă durata ar trebui să fie citite și folosite că ieșiri, atât pentru $h(t)$ cât și pentru $y(t)$.

Se poate observa cum în interiorul celulei LSTM se folosesc două funcții de activare diferite, funcția sigmoid și tanh, prezentate în 3.1. Funcția sigmoid filtrează cantitatea de informație pe care fiecare poartă o cedează mai departe, fiind urmată de o operație de înmulțire, dacă o valoare din vectorul rezultat este 0 informația aferentă acelei poziții nu este transmisă în continuare. Cu cât acea valoare este mai apropiată de 1 cu atât informația este transmisă mai complet.

Transferul de informație printr-o astfel de celulă se face de la stânga la dreapta, Fig.3.8. În primul rând, poarta de uitare, $f(t)$, va filtra memoriile de lungă durata semnificative provenite din $c(t-1)$ luând în considerare memoria de scurtă durata de la momentul anterior, $h(t-1)$, și datele de intrare curente, $x(t)$. Funcția $g(t)$ produce noul set de informații obținute prin combinarea stării trecute cu datele de intrare. Aceste valori sunt apoi înmulțite cu rezultatul porții de "update" care decide care din aceste informații merită stocate în memoria "long-term", fiind adunate la aceasta. După ce varianta reînoită a memoriei de lungă durată este obținută, $c(t)$, ea este conectată la o ieșire a celulei și la o altă funcție de activare tanh. Poarta de ieșire, $o(t)$, transformă aceste valori în ieșirea celulei, $y(t)$, cât și noua memorie de scurtă durată, $h(t)$.

Astfel, celula LSTM poate să recunoască care din datele de intrare sunt importante și să le stocheze în starea de lungă durata, prin poarta de input, să învețe să le păstreze pentru cât timp sunt relevante , prin poartă de uitare, și să le extragă oricând le consideră necesare, folosind poartă de ieșire. Succesul obținut de celulele LSTM în captarea relațiilor temporale de lungă durata poate fi explicat prin utlizarea acestor succesiuni de porți de control. Totuși, acest succes a fost demonstrat și într-o manieră empirică prin folosirea unor baze de date special concepute pentru această sarcină (Bengio et al., 1994 [46]; Hochreiter and Schmidhuber, 1997 [47]; Hochreiter et al., 2001 [48]) dar și prin aplicarea lor în diferite probleme din lumea inteligenței artificiale.

Rețelele neuronale recurente au avut mult succes în recunoașterea emoției în vorbire. Motivul pentru care am ales să folsoesc acest tip de rețele a fost capacitatea acestora de a determina relații temporale între segmentele înregistrărilor audio. Deoarece emoția nu apare doar într-un singur segment, fracțiune de secundă, ci aceasta este prezenta pe întreagă durată a semnalului audio, obținerea conexiunilor temporale dintre informațiile emoționale prezente în aceste segmente este crucială.

3.6 Mecanismul de atenție

Mecanismului de atenție este bazat pe principiul de funcționare al percepției umane. Omenii își focuseaza atenția în mod selectiv, doar pe anumite părți din spațiul vizual sau auditoriu, pentru a extrage informația necesară atunci când este nevoie. Odată extarsa, această informație este combinată cu cea obținută din alte puncte de atenție pentru a crea o reprezentare internă a scenei [32, 60].

În SER, acest mecanism are ca scop identificarea segmentelor înregistrării audio cu o mare încărcătură emoțională și îndreptarea atenției modelului pe aceste segmente în timpul antrenării. Din punct de vedere matematic, dorim să obținem un set de parametrii α_i, unde i ia valori între 1 și numărul de segmente extrase, L, care să funcționeze ca ponderi reprezentative pentru importanța emoțională a fiecărui segment.

$$r = \sum_{i=1}^{L} \alpha_i h_i \tag{3.11}$$

unde r este suma ponderată a segmentelor semnalului audio, α_i reprezintă ponderea segmentului i iar h_i reprezintă datele aferente segmentului i.

Mecanismul de atenție poate fi interpretat că o medie ponderată a datelor din fiecare segment. Calcularea ponderilor folosite în această medie se realizează prin adăugarea unui nou nivel de parametrii care vor fi antrenați să depisteze emoțiile din fiecare segment în forma următoare.

$$\alpha_i = \frac{\exp(w^T h_i)}{\sum_{t=1}^{L} \exp(w^T h_t)} \tag{3.12}$$

unde w reprezintă parametrii antrenați.

Această funcție se numește "softmax" și are ca rezultat un vector egal cu numărul segmentelor extrase. Pentru fiecare segment se obținute un număr între $(0, 1)$, care poate fi interpretat ca probabilitatea ca acel segment să fie bogat în informație emoțională. Parametrii w sunt antrenați în timpul procesului de învățare împreună cu restul modelului de clasificare.

3.7 Cicluri OODA

Conceptul de ciclu OODA a fost introdus prima dată de către strategul militar al Statelor Unite ale Americii, colonelul John Boyd. Boyd susținea că procesul de luare al decizilor decurge într-un ciclu cu 4 stagii principale observare-orientare-decizie-acțiune. Orice entitate care execută acest ciclu mai repede, observând și reacționând la evenimentele care iau loc înaintea oponentului, poate avea un avantaj considerabil și chiar să între în ciclul de decizie al oponentului. Astăzi conceptul de ciclu OODA este folosit în diferite domenii cum ar fii afaceri, aplicarea legii, strategii militare și securitate cibernetică.

În Machine Learning aceste cicluri decizionale pot fi adaptapte pentru a eficientiza procesul de antrenare. La fiecare N epocii, traversări complete ale setului de date de antrenare în timpul procesului de învățare, se va extrage câte un set de exemple pentru a reduce timpul în care acest proces are loc. Oponentul în această situație este chiar modelul în sine în versiunea în care folosește întregul set de date de alungul antrenării. Folosirea acestui mecanism permite grăbirea antrenarii fără a avea efecte negative puternice asupra preciziei modelului. Timpul de învățarea al unui sistem SER este mare din cauza complexității ridicate a problemei, iar prin introducerea acestui mecanism acest interval de timp poate fi aproximativ înjumătățit.

În soluția propusă numărul de exemple extrase la fiecare N epoci este stabilit prin formulă următoare:

$$D_i = D_{i-1} - \frac{p * D}{\frac{E}{N}} \tag{3.13}$$

unde D este numărul total de exemple folosite la antrenare, D_i reprezintă numărul de exemple folosite în epoca i, p este procentajul de exemple extrase, E este numărul total de epocii iar N este numărul de epocii dintr-un ciclu.

În această lucrare de diplomă s-au utilizat două moduri de determinare a setului de exemple care urmează să fie extrase, în mod aleatoriu și în funcție de eroarea fiecărui exemplu. Rezultatele pentru ambele cazuri sunt prezentate în capitolul 5. Această funcționalitate este opțională fiind folosită doar pentru a reduce timpul de antrenare.

4 Descrierea implementării

După cum am menționat și în capitolele anterioare, proiectul meu este alcătuit din două părți principale: modelul Machine Learning pentru recunoașterea emoțiilor în semnale audio și interfața grafică pentru utilizator. Prima parte reprezintă soluția propusă pentru consturirea unui model SER iar cea de a două face că această soluție să poată fi antrenată, testată și utilizată într-un mod accesibil, oferind o gama de funcționalități.

Diagramă din Fig. 4.1 ilustrează distribuția claselor folosite și relațiile dintre acestea. Se poate observa cum există un număr mare de module de procesare de date, aferente diferitelor moduri de extragere a caractersiticitlor semnalului audio, dar și a diferitelor funcționalități, precum antrenarea și testarea, inferența și inferența în timp real (online). Clasa care susține interfață grafică (UI_MainWindow) apelează aceste moduri de utilizare în funcție de acțiunile utilizatorului.

Lucrarea de diplomă a fost implementată în limbajul de programare Python, folosind cadrul de programare specializat pentru dezvoltarea aplicațiilor Machine Learning, Tensorflow [61]. Mai exact pentru a eficentiza înmulțirile matriciale care stau la baza oricărei operații din spatele modelelor ML, am folosit o versiune specială numită *tensorflow-gpu*, care realizează oricare astfel de operație matematică folosind placa video. Aceste tipuri de procesare sunt specializate in a lucra cu date din domeniul video, devenind mult mai eficiente când vine vorba de înmulțiri matriciale de dimensiuni mari. Avantajul obținut prin folsoirea acestei versiuni de Tensorflow este mărirea vitezei de antrenare a modelelor. Pentru extragerea semnalului din fiserele audio, realizarea interfeței grafice și a funcționalității de înregistrarea online am folosit diferite librării specifice prezentate în următoarele subcapitole.

4.1 Modului de implementare a sistemului de recunoaștere a emoțiilor

După cum am menționat și în capitolul 2, un model Machine Learning de clasificare tipic trebuie să realizeze trei sarcini: preprocesarea datelor, extragerea caracteristicilor de intrare și clasificarea propriu-zisă. Arhitectura propusă de mine pentru realizarea unei soluții pentru problema recunoașterii emoției este ilustrată în Fig. 2.2. În capitolul precedent am prezentat o descriere teoretică a algoritmilor folosiți pentru construirea unui astfel de model, urmând că în acest capitol să prezint modul în care aceste concepte teoretice au fost implementate.

4.1.1 Bazele de date folosite

În conextul domeniului SER una din principalele probleme este numărul redus de exemple din majoritatea bazelor de date existente. Cu toate că în ultimii ani au apărut baze de date noi, care folosesc un număr mai mare de exemple, acestea nu sunt publice și necesită sume destul de mari de bani pentru a le accesa. Din aceste motive am decis să folosesc mai multe baze de date publice în limbi și configurații de înregistrare diferite. Pe lângă aceste baze de date profesioniste, am decis să adaug și un set de înregistrări proprii pentru a familiariza modelul cu configurațiile microfonului meu și pentru a contribui la majorarea numărului de exemple pentru antrenare.

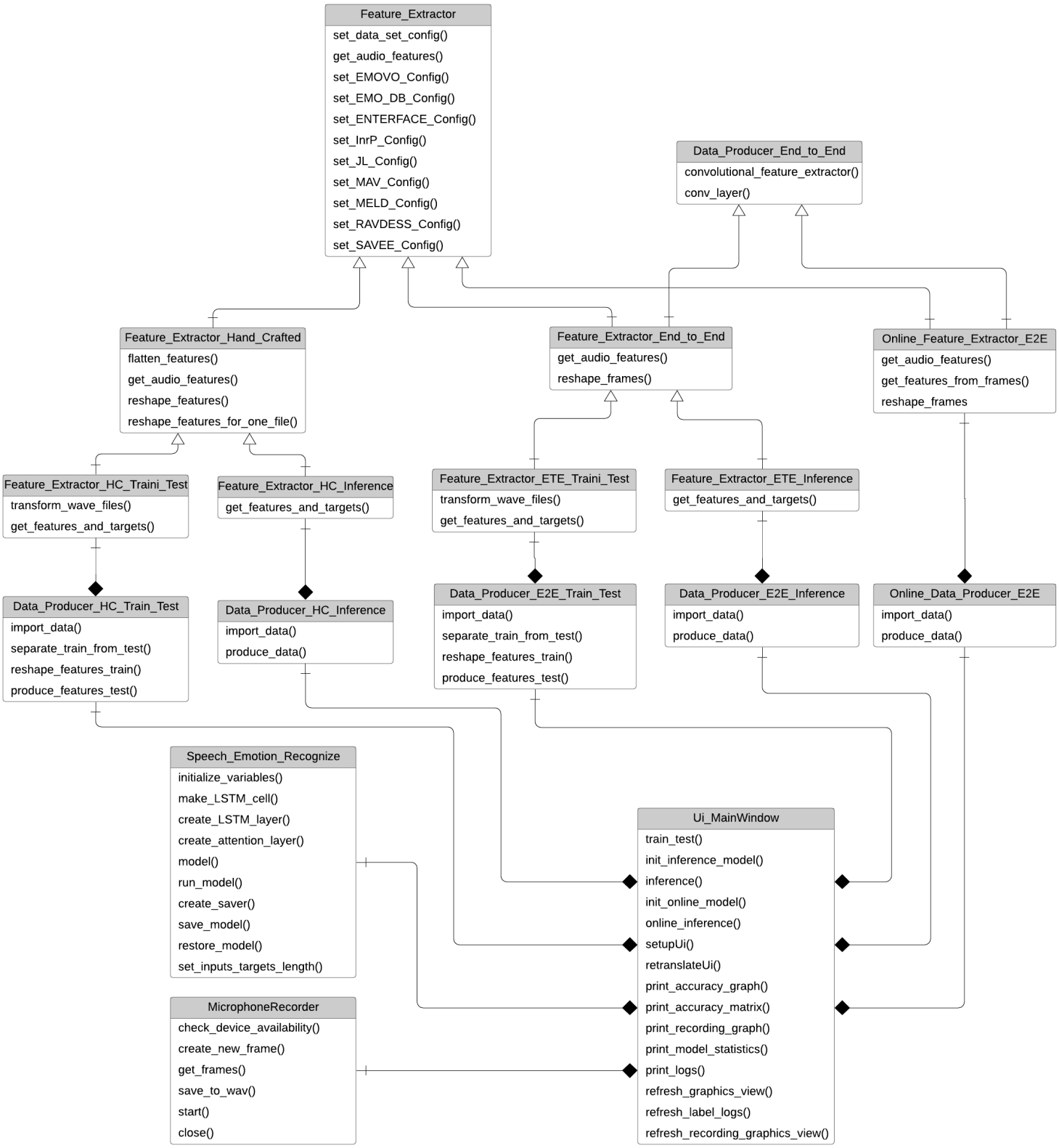


Figura 4.1: Diagrama UML ilustrativă a distribuției claselor proiectului. Utilizatorul are acces la întregul set de funcționalități al aplicației prin interfață grafică implementată în clasa *UI_MainWindow*.

Bazele de date pe care le-am folosit sunt următoarele: EMO-DB, RAVDESS, SAVEE, EMOVO, MAV, ENTERFACE, JL și InrP.

- EMO-DB [62] este o bază de date în limba germană, conținând aproximativ 500 de înregistrări jucate de 10 actori în contextul a 7 emoții: fericire, enervare, anxietate, frică, plictisire, dezgust și neutru.
- RAVDESS [63] conține 1440 de fișiere audio care reprezintă înregistrări jucate de 24 de actori profesioniști aferente a 7 emoții: neutru, fericire, tristețe, enervare, frică, surpindere și dezgust. Limba în care au fost înregistrate aceste emoții jucate a fost engleză, cu un accent nord-american.
- EMOVO [64] este o baza de date care conține 588 de înregistrări audio a 7 emoții dezgust, frică, enervare, fericire, surpiza, tristețe și neutru. Înregistrările au fost obținute prin angajarea a 6 actori.
- MAV [65] sau "Montreal Affective Voices" este alcătuită din 90 de înregistrări nonverbale corespunzând unui set de 8 emoții: enervare, dezgust, frică, durere, tristețe, surprindere, fericire, plăcere și neutru, înregistrate de 10 actori.
- ENTERFACE [66] conține 1320 de înregistrări audio, care surpind 6 emoții: enervare, dezgust, frică, fericire, tristețe și surpindere în 14 limbi diferite.
- JL [67] este alcătuită din 2400 înregistrări din 240 de propoziții jucate de 4 actori în contextul a 5 emoții: enervare, tristețe, neutru, fericire, entuziasm.
- InrP reprezitna setul de exemple personale alcătuit din 245 de înregistrări care cuprind cele 4 emoții clasificate: enervare, tristețe, fericire și neutru.

4.1.2 Preprocesarea datelor

Tipul datelor de intrare folosite de modelul propus de mine sunt fișierele audio, mai exact fișierele cu extensia .wav care este cea mai folosită extensie pentru stocarea înregistrărilor audio în bazele de date SER existente. Încărcarea valorilor amplitudinilor semnalului audio într-o matrice de stocare se realizează folosind o librărie specifică pentru porcesarea semnalului audio în Python, numită *librosa* [53]. Această librărie oferă o gama largă de funcții pentru procesarea semnalelor audio, urmând să fie prezentate în subcapitolul explicativ pentru extragerea caracteristicilor de intrare 4.1.3.

Pentru alcătuirea setului de date am folosit înregistrări din diferitele baze de date prezentate mai sus. Înregistrările fiecărei baze de date au fost realizate folosind diferite frecvențe, variind între 16KHz și 48KHz. Pentru a reduce memoria folosită cât și a timpului necesar pentru antrenare (de la 12h la aproximativ 4h), am decis să re-esantionez fișierele audio cu o frecvență de 16KHz folosind librăria librosa. Teorema lui Nyquist ne spune că folosind o rată de eșantionare de 16KHz putem surpinde fără pierderi orice semnal cu frecvente mai mici sau egale cu 8KHz. Aceste frecvențe sunt mai mult decât necesare pentru captarea vocii umane, care în contexte normale nu depășește mai mult de 4KHz. Din punct de vedere empiric nu am gaist nici un dezavantaj în a folosit mecanismul de re-eșantionare, sistemul SER obținând aceași acuratețe în același număr de epoci cu sau fără această tehnică.

Înainte ca semnalul audio să poată fi folosit pentru extragerea informațiilor emoționale, acesta trece printr-un proces de partitionare în segmente de lungime fixă. Acest lucru ne permite să aplicăm transformatele Fourier, sau alte funcții asemănătoare, pe porțiuni mai mici din semnalul

inițial, obținând o descriere mai detaliată a acestia. Pentru a reduce datele pierdute între segmente, de obicei se alege un pas de lungime mai mică decât dimensiunea segmentelor. Astfel o parte din informația segmentului actual va fi prezentă și în cel viitor.

Următoarea etapă a procesului de preprocesare este aplicarea unor funcții fereastara pe fiecare "frame", segment audio. Acest lucru este realizat pentru a reduce pierderea de informații la aplicarea transformărilor Fourier din cauza discontinuității de la marginea segmentelor. Funcțiile fereastră sunt funcții care converg la zero în afară unui anumit interval. Înmulțind un segment cu o astfel de funcție are ca rezultat dispariția discontinuitatiilor de la marginea segmentului, similar cu privitul printr-o fereastră normală. Funcția fereastră folosită se numește "Hann-window", numită după Julius von Hann, și are formula următoare.

$$w[n] = \sin^2(\frac{\pi n}{N}) \tag{4.1}$$

4.1.3 Extragerea caracteristicilor de intrare

Semnalul audio nemodificat nu este cea mai eficientă modalitate de reprezentare a informației emoționale dintr-un discurs. Din acest motiv există modalități, funcții matematice, care să pună în lumină caracteristicile semnalului audio pentru a putea fi interpretat de sistemele SER în detectarea emoțiilor. Aceste formule aplicate direct pe semnalul audio au fost prezentate în subcapitolul 3.3 pentru caracteristicile de intrare "hand-crafted" și 3.4 pentru extragerea antrenabilă a caracteristicilor.

În primul caz, extragerea caracteristilor de intrare, precum coeficienții *cepstrali Mel (MFCC)* sau *delas* și *delta-deltas* s-a relizeazat prin folosirea funcțiilor aferente prezente în librăria *librosa*. Pentru ca extragerea acestor date să fie corectă și pentru ca rezultatele obținute să aibe dimensionalitatea dorită, a fost nevoie să caluclez lungimea potrivită a segmentelor, mărimea pasului de la un segment la altul, mărimea funcției fereastră etc.

```
def _get_audio_features(self, wav_file):
        signal, rate = librosa.load(wav_file,  self.hz)
        mfcc = librosa.feature.mfcc(y=signal, sr=rate, hop_length=260,
                                    n_mfcc=20)
        delta = librosa.feature.delta(mfcc)
        delta_deltas = librosa.feature.delta(delta)
        rms = librosa.feature.rms(y=signal, frame_length=640,
                                  hop_length=260)
        zcr = librosa.feature.zero_crossing_rate(y=signal,
                                  frame_length=640, hop_length=260)
        chroma = librosa.feature.chroma_stft(y=signal, sr=rate, n_fft=820,
                                  win_length=640, hop_length=260)
        rolloff = librosa.feature.spectral_rolloff(y=signal, sr=rate,
                                  n_fft=820, win_length=640, hop_length=260)
        features = [mfcc, delta, delta_deltas, rms, zcr, chroma, rolloff]
        return features
```

Algoritm 4.1: Extragerea caracteristicilor hand-crafted, 3.3, folosind libraria librosa.

În cel de al doilea caz am folosit librăria librosa doar pentru a aduce semnalul audio la o formă care eficientizează procesul de antrenare al rețelelor neuronale convoluționale. Astfel am extras spectograma Mel, 3.4.1, folosind funcția corespunzătoare din librăria *librosa*. Rezultatul obținut este o imagine de dimensiuni 128x128 pentru fiecare din segmentele semnalului audio. Această imagine este apoi normalizată și transmisă rețelei convoluționale care se ocupă cu extragerea automată a informației emoționale. Deoarece rețeaua convoluțională este la rândul ei antrenată în timpul procesului de învățare, caracteristicile extrase de aceasta vor fi mai specifice sarcinii de

recunoaștere a emoției decât cele folosite în metodă "hand-crafted".

```
1   def _get_audio_features(self, wav_file):
2         signal, _ = librosa.load(wav_file, self.hz)
3         librosa.core.time_to_frames
4         stft = librosa.feature.melspectrogram(signal, n_fft=512,
5                                 win_length=128, hop_length=32, center=False)
6         return stft
```

Algoritm 4.2: Extragerea spectogramei Mel, 3.4.1, folosind libraria librosa.

Arhitectura rețelei convoluționale folosite este prezentată în 3.4, iar secvența de cod care realizează această funcționalitate este ilustrată în Alg. 4.3.

Funcția _convolutional_feature_extractor este responsabilă pentru crearea modelului rețelei convoluționale. Parametrul de intrare *stft* reprezintă vectorul de spectograme Mel aferente segmentelor unei anumite înregistrări și este transmis primului nivel al rețelei convoluționale.

Fiecare nivel al rețelei convoluționale este construit folosind metoda *conv_layer*. În această metodă se crează parametrii antrenabili din fiecare nivel, Alg.4.3 linia 5, în funcție de dimensiunea filtrelor și a datelor de intrare. Parametrii obținuți sunt transmiși funcției *tf.nn.conv2d* a librăriei Tensorflow care va crea nivelul convolutional specializat pe date în două dimensiuni.

Pentru primul, nivel dimensiunea filtrelor a fost aleasă de *8x8x1* pixeli iar numărul acestor filtrii este de 32, *channels_out*. Între nivele rețelei am folosit funcția de activare *tanh*, urmată de funcția de subeșantionare maxim din librăria Tensorflow, *tf.nn.max_pool*. Motivația folosirii acestor funcții este descrisă în secțiunile 3.1 respectiv 3.4.3.

Tehnică numită "batch normalization", 3.4.2, este implementată prin normalizarea valorilor rezultate în urmă funcției "max_pooling" după fiecare nivel al acestei rețele.

Cel de-al doilea nivel este construit în aceeași modalitate că și primul, dar conține 16 filtre de dimensiunea *4x4x32*. La final, rezultatul care ajunge să aibă dimensiunea *(4 * nr_segmente)x16x16*, este aplatizat și adus la forma *(4*nr_segmente)x256* pentru a deservi ca intrare pentru rețeaua neuronală recurentă.

```
1   class Data_Producer_End_to_End(object):
2       def conv_layer(self, input_data, filter_size, channels_in, channels_out,
3                           strides, conv_layer_dropout, name="Conv"):
4           W = tf.get_variable("Weights_"+name+"_Layer", dtype=tf.float32,
5                   shape=[filter_size, filter_size, channels_in, channels_out])
6           return tf.nn.tanh(tf.nn.dropout(tf.nn.conv2d(input=input_data,
7                                                        filter=W,
8                                                        strides=strides,
9                                                        padding='SAME',
10                                                       use_cudnn_on_gpu=True),
11                                                       conv_layer_dropout))
12
13      def _convolutional_feature_extractor(self, stft, conv_layer_dropout):
14              self.init = tf.glorot_normal_initializer()
15              with tf.variable_scope("Convbb", reuse=tf.AUTO_REUSE,
16                                      initializer=self.init):
17                  stft = batch_normalization(stft)
18                  conv1 = self.conv_layer(input_data=tf.expand_dims(stft,axis=3),
19                                          filter_size=8,
20                                          channels_in=1,
21                                          channels_out=32,
22                                          strides=[1, 2, 2, 1],
23                                          conv_layer_dropout=conv_layer_dropout,
24                                          name="conv1")
25                  conv2 = tf.nn.max_pool(conv1, [1,2,2,1], [1,2,2,1],
26                                          padding="SAME")
```

```
            conv2 = batch_normalization(conv2)
            conv3 = self.conv_layer(input_data=conv2,
                                    filter_size=4,
                                    channels_in=32,
                                    channels_out=16,
                                    strides=[1, 2, 2, 1],
                          conv_layer_dropout=conv_layer_dropout,
                                    name="conv2")
            conv3 = tf.nn.max_pool(conv3, [1,2,2,1], [1,2,2,1],
                                    padding="SAME")
            conv3 = batch_normalization(conv3)
            conv_out = tf.reshape(conv3, (-1, 256))
            return conv_out
```

Algoritm 4.3: Implementarea nivelelor convoluționale care realizează extragerea caracteristicilor în manieră end-to-end folosind procedurile librăriei Tensorflow.

4.1.4 Clasificatorul sistemului SER

În Fig.2.2 se poate observa cum modulul clasificator este alcătuit din două celule bidirecționale recurente pe două nivele, urmat de mecanismul de atenție și de un nivel neuronal "dens". Fiecare dintre aceste componente aduce un anumit avantaj arhitectural care este prezenat în continuare.

Rețelele recurente, 3.5, spre deosebire de alte tipuri de rețele neuronale reușesc să determine atât informațiile emoționale de la un anumit moment de timp, cât și relațiile temporale între emoții din diferite momente. Astfel emoțile prezente în începutul înregistrării o să aibe o anumită influență și asupra emoțiilor recunoscute la finalul acesteia. În construirea modulului neuronal recurent am decis să folosesc două celule LSTM bidirecționale, 3.8, pentru a marii acuratețea sistemului SER. Termenul bidirecțional înseamnă că o celulă va procesa înregistrarea de la început spre final în timp ce cea de a două va procesa aceași informație în sens invers, de la final spre început. În mod normal rețelele recurente se folsoesc de informația actuală și cea din trecut pentru a lua decizia la momentul curent, totuși există aplicații, că și SER, unde decizia la momentul t poate fi influențată și de date prezente la $t + 1$. Un exemplu ar fi "speech recognition", unde intonația unei litere poate depinde de litera care o urmază, "co-articulație", sau chiar de cuvintele următoare (e.g. cuvântul "the" în limba engleză se pronunță diferit dacă e urmat de un cuvânt care începe cu o vocală sau consoană). În recunoașterea de emoții în vorbire, emoția fluctuează într-o conversație. Oamenii pot intuii anumite expresii de astfel emoții, de exemplu când cineva spune o glumă și bufnește în ras la final. Algorimtii de detecție a emoțiilor ar trebui să reușească și ei să se folosească de informații viitoare, "intuiții", pentru a determina emoția din prezent.

Pentru a marii numărul gradelor de libertate ale rețelei recurente, și în final măririi acurateții modelului, am folosit pentru ambele tipuri de traversări ale semnalului audio două nivele neuronale. Secțiunea de cod corespunzătoare acestui modul este prezentată mai jos.

```
def make_lstm_cell(self, hidden_size):
    cell = tf.contrib.rnn.LSTMCell(num_units=hidden_size,
                                   use_peepholes=True,
                                   initializer=tf.glorot_uniform_initializer())
    if self._is_training and self._keep_prob < 1:
        cell = tf.contrib.rnn.DropoutWrapper(cell,
                                             input_keep_prob=self._keep_prob,
                                             output_keep_prob=self._keep_prob,
                                             variational_recurrent=True,
                                             dtype=tf.float32,
                                             input_size=hidden_size)
    return cell
```

```
13
14 def create_LSTM_layer(self, inputs, hid_size, name=None):
15     with tf.variable_scope(name):
16         lstm_cells_fw = [self.make_lstm_cell(hid_size) for _ in range(2)]
17         lstm_cells_bw = [self.make_lstm_cell(hid_size) for _ in range(2)]
18         multi_cell_fw = tf.contrib.rnn.MultiRNNCell(lstm_cells_fw,
19                                                     state_is_tuple=True)
20         multi_cell_bw = tf.contrib.rnn.MultiRNNCell(lstm_cells_bw,
21                                                     state_is_tuple=True)
22         initial_zero_state_fw = multi_cell_fw.zero_state(1, tf.float32)
23         initial_zero_state_bw = multi_cell_bw.zero_state(1, tf.float32)
24         inputs = tf.expand_dims(inputs, axis=0)
25         outputs, _ = tf.nn.bidirectional_dynamic_rnn(multi_cell_fw,
26                                     multi_cell_bw, inputs,
27                                     initial_state_fw=initial_zero_state_fw,
28                                     initial_state_bw=initial_zero_state_bw)
29         return tf.concat(outputs, 2)[0]
```

Algoritm 4.4: Implementarea celulelor LSTM bidirectionale pe doua nivele folosind procedurile specifice ale librariei Tensorflow.

Prima funcție, *make_lstm_cell* va crea celula recurentă LSTM, primind ca parametrii numărul de "neuroni" interni, care este egal pentru fiecare poartă, *hidden_size*. Dacă modelul se află în timpul procesului de antrenare se poate folosii tehnica de regularizare numită *dropout*. Această tehnică va dezactiva o anumită parte (1 - *keep_prob*) din rețeaua recurentă în mod aleator pentru fiecare tranziție prin rețea. Prn acest mecanism se elimină dependențele puternice dintre neuroni, care poate duce la procesul de "overfitting". Fenomenul de "overfitting" apare atunci când modelul devine mult prea bine specializat pe datele de antrenare și ajunge să nu generalizeze bine pe datele de test, având rezultate slabe în acest caz.

Funcția *create_LSTM_layer* este cea care crează întreg modulul recurent prin concatenarea celulelor recurente generate de *make_lstm_cell*. În listele *lstm_cells_fw* și *lstm_cells_bw* sunt stocate cele două nivele recurente, celule LSTM, care sunt date apoi ca parametru funcției din librăria Tensorflow [61], *tf.nn.bidirectional_dynamic_rnn*. Această funcție va crea rețeaua neuronală, oferind celulelor din *lstm_cells_fw* datele de intrare în de la început spre sfârșit și celor din *lstm_cells_bw* în sens invers. Rezultatele celor două seturi de nivele recurente este salvat în *outputs*. Varianta concatenată este returnată de această funcție, reprezentând ieșirile acestui modul.

Rezultatele care reies în urmă procesării recurente a datelor sunt transmise mecanismului de atenție prezentat în 3.6. După cum am spus și mai sus această parte a sistemului SER are rolul de accentua informațiile din segmentele bogate în emoții și de a le neglija pe cele care pot să aducă o notă de ambiguitate în procesul de clasificare.

```
1 def create_attention_layer(self, frame_predictions, weigths_dim):
2     W = tf.get_variable("Attention_Weights", dtype=tf.float32,
3                         shape=[weigths_dim, 1])
4     b = tf.get_variable("Attention_Bias", dtype=tf.float32, shape=[1])
5     alpha = tf.matmul(frame_predictions, W) + b
6     alpha = tf.nn.softmax(alpha, axis=0)
7     return tf.expand_dims(tf.reduce_sum(tf.multiply(frame_predictions,
8                           alpha[: tf.newaxis]), axis=0), axis=0)
```

Algoritm 4.5: Metoda care implementeaza mecanismul de atentie, 3.6.

Funcția care implementează tehinca de atenție este destul de scurtă. În prmul rând, se crează ponderile *W* și "bias"-ul *b* cu dimenisunea rezultatelor modulului recurent aferente fiecărui segment. Aceste ponderi sunt înmulțite matricial cu datele fiecărui segment urmate de aplicarea

funcției *softmax* 3.12 pe rezultate. În acest moment fiecărui segment i s-a atribuit un coeficient de "importanță" *alpha*, rămânând doar ca acești coeficenti să fie înmulțiți cu valorile rezultate în urmă modulului recurent și să se adune valorile obținute pentru a realiza suma ponderată descrisă și în 3.6.

Odată ce mecanismul de atenție a fost folosit, informațiile din diferitele segmente ale semnalului audio au fost comasate într-un singur set obținându-se astfel rezumatul informației emoționale a întregii înregistrării audio. Acest set este transmis unui nivel neuronal simplu care are rolul de a reduce dimensiunea acestui rezumat emoțional și de a oferi la ieșire probabilitatea ca înregistrarea curentă să aparțină uneia dintre celor 4 emoții clasificate: enervare, fericire, tristețe și neutru.

Funcția care leagă toate aceste componente și crează modelul propus în arhitectura din 2.2 este următoarea.

```
1  def model(self):
2      with tf.variable_scope("Speech_Emotion_Recognizer", reuse=tf.AUTO_REUSE
3                             ,initializer=self.init):
4          rnn_layer = self.create_LSTM_layer(self._inputs,
5                                             self._hidden_size,
6                                             "Reccurent_Module")
7          attention_layer_output = self.create_attention_layer(rnn_layer,
8                                                               self._hidden_size*2)
9          predictions_1 = tf.layers.dense(attention_layer_output,
10                                             self._emotion_number,
11                                             name="Output_Layer")
12         predictions = tf.reduce_sum(predictions_1, axis=0)
13         targets_raw_ = tf.nn.softmax(predictions, axis=0)
14         targets_ = tf.cast(tf.equal(targets_raw_,
15                                     tf.reduce_max(targets_raw_)),
16                                     tf.float32)
17         if self._is_inference:
18             self.predictions_raw = targets_
19             self.predictions = targets_raw_
20                 return
21         self.label_pred = tf.argmax(targets_raw_)
22         self.label_true = tf.argmax(self._targets)
23         self.accuracy = tf.cast(tf.equal(self.label_pred, self.label_true),
24                                 tf.float32)
25         if not self._is_training:
26             return
27         cross_entropy = tf.nn.softmax_cross_entropy_with_logits_v2(
28                                                         labels=self._targets,
29                                                         logits=predictions)
30         adam_opt = tf.train.AdamOptimizer(self._learning_rate)
31         self.optimizer = adam_opt.minimize(cross_entropy)
```

Algoritm 4.6: Functia care creaza graficul de executie Tensorflow, conectand toate componentele de procesare ale sistemului SER.

Funcția *model* folosește caracteristicile extrase de rețeaua convolutionala în cazul end-to-end sau formulele matematice predefinite în cazul hand-crafted prin parametru *self._inputs*. Aceste caracteristici sunt transmise modulului recurent, care este apoi conectat la mecanismul de atenție și la nivelul rețelei neuronale dense numit "Output_Layer". Rezultatele acestui nivel au dimensiunea *self._emotion_number*, în cazul nostru 4, pe care se aplică funcția *softmax*, 3.12, pentru a fi reprezntative din punct de vedere probabilistic. Pentru a fi aduse într-un format de tipul [0,0,1,0], rezultatele sunt comparate cu probabilitatea maximă, memorând 1 dacă sunt egale și 0 în caz contrar.

Dacă modelul este folosit în cazul de inferență după linia 17 din algoritmul 4.6 se returnează

predicția obținută, care este afișată în interfață grafică.

În schimb, dacă se execută procesul de testare, se salvează emoția clasificată și cea adevărată pentru crearea statisticilor ilustrate în interfața cu utilizatorul și se determină dacă predicția a avut succes (dacă emoția clasificată este egală cu cea adevărată) la linia 23.

În cazul în care modelul se află în timpul antrenării se calculează eroarea, sau mai exact funcția de "loss", care ne spune exact cât de departe a fost modelul de a prezice corect emoția. Funcția folosită de obicei în cazul clasificării de mai multe clase mutual exclusive se numește "cross entropy", $L = -\sum_{i=1}^{K} y_i \log(P_i)$ unde K este numărul de clase, y_i este valoare adevărată a clasei i (0 sau 1) iar P_i este probabilitatea clasei i. Dacă y_i are valoarea 1, iar probabilitatea este apropiată de 1 atunci eroarea, L, are o valoare foarte mică. În schimb, dacă probabilitatea este mică prin aplicarea logaritmului negativ, se va obține o valoare a erorii mare.

Această valoare a erorii, alături de rata de antrenare, sunt oferite ca parametru de intrare unui optimizator care are rolul de a realiza mecanismul numit "backtracking", prin care se vor actualiza valorile ponderilor din interiorul rețelei neuronale convoluționale, recurente, mecanismului de atenție și retelei dense, pentru a reduce eroarea determinată pentru exemplul curent. Optimizatorul folosit este numit Adam [69] ("Adaptive momement estimation"), deoarece este mult mai rapid decât tipurile de optimizatori tradiționali precum "gradiend descent". Acest optimizator fiind o extensie a algoritmului "gradient descent" , si folosește diferite rate de învățare pentru parametrii interni și așa numite "momente" care sunt funcții matematice ce determină când învățarea poate fi accelerată fără pierderea acurateții. Eficiența acestui algoritm a fost demonstrată in diverse implementări, fiind unul dintre cei mai folosiți optimizatori din domeniul Machine Learning.

4.2 Implementarea și utilizarea interfeței grafice

Interfața cu utilizatorul oferă două moduri de utilizare, pentru antrenare și pentru inferență. Primul mod permite utilizatorului să aleagă diferite configurații de parametrii pentru antrenarea modelului și să observe statistici, în timp real, asupra evoluției acestuia. Cel de al doilea mod este orientat pe partea aplicativă a sistemului SER, permițând utilizatorului să determine emoția predominantă din diferite fișiere audio pre-înregistrate sau chiar să își inregistreze propriul discurs pentru inferență. Cele două moduri de utilizare vor fi descrise în subcapitolele următoare, împreună cu câteva imagini care prezintă interfața grafică.

Utilizatorul poate alege pe care dintre cele două moduri dorește să le folosească prin două butoane radio *Train* și *Inference*. Butonul *Start* începe să execute acțiunea aleasă anterior.

Pentru realizarea acestei interfețe am folosit librăria PyQt5, care este o varianta a celebrei librării Qt pentru realizarea elementelor grafice din limbajul de programare C++. Tema folosită pentru aplicație a fost realizată în librăria qdarkgraystyle [13], oferind culoarea și textura entităților grafice din interfață, precum butoane, comboBox-uri, checkBox-uri, slider-e, tabele etc.

4.2.1 Interfața grafică în modul de antrenare

Interfața grafică în modul de antrenare, ilustrat în Fig. 4.2, oferă în partea stângă o serie de funcționalități de editare a parametriilor modelului în timp ce în partea dreaptă se poate observa un grafic care ilustrează evoluția acurateții modelului în timpul antrenarii și paginile care conțin jurnalul cu logurile și matricea de confuzie.

În secțiunea "Model setttings" din interfața grafică utilizatorul poate să controlzeze contextul în care se execută procesul de antrenare prin intermediul următorului set de parametrii:

- *Select dataset*, este un comboBox care listează setul de baze de date care pot fi folosite

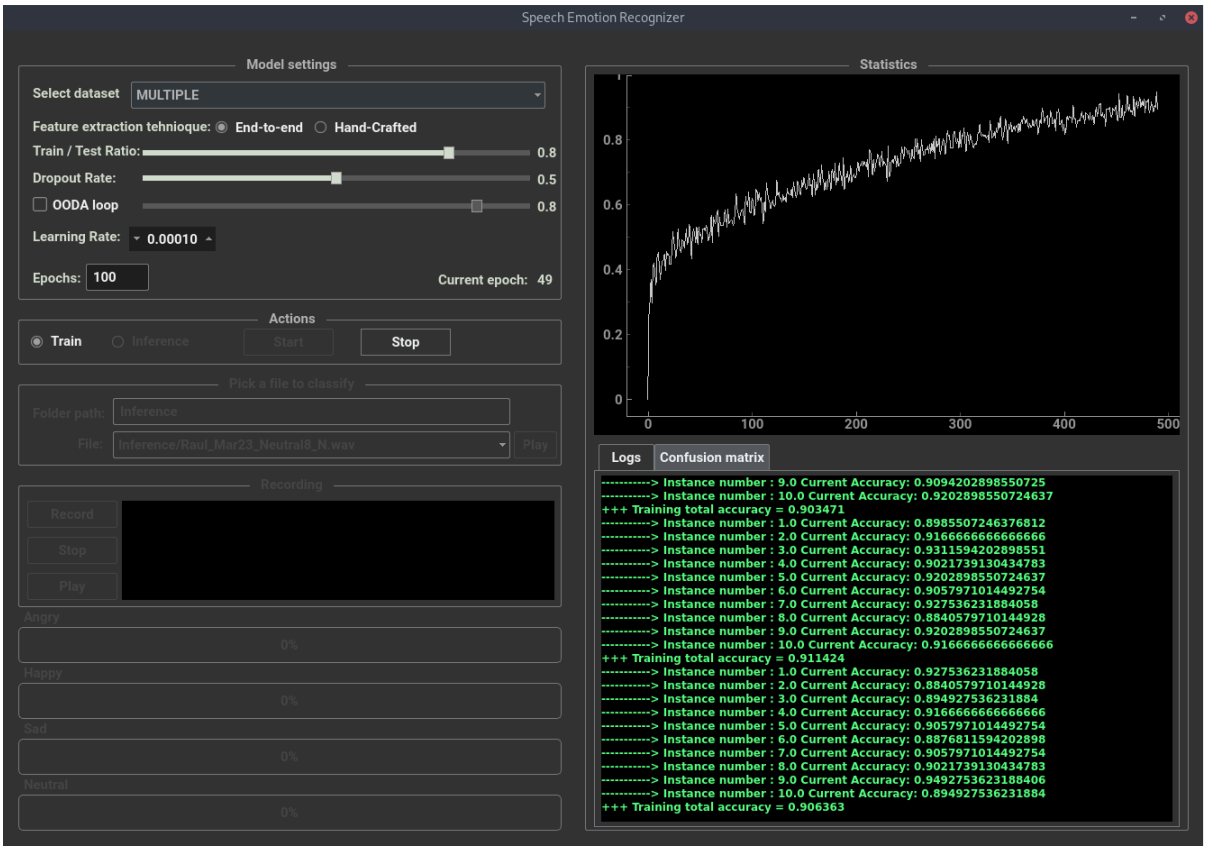


Figura 4.2: Interfață grafică în modul de antrenare

pentru antrenare. Utilizatorul poate să aleagă una dintre ele sau opțiune Multiple, unde un număr din acestea au fost alese pentru a facilita opțiunea folosirii unui număr mai mare de baze de date într-o antrenare de tip "multi-domain".

- *Feature extraction tehnique*, prezintă două butoane tip radio prin care utilizatorul poate decide modul în care sunt extrase caracteristicile de intrare din semnalul audio.
- *Train/Test Ratio* este un slider orizontal care permite utilizatorului să decidă rația din setul de date utilizată pentru antrenare din baza de date aleasă la *Select dataset*. Restul exemplelor vor fi folosite pentru testare.
- *Dropout Rate*, este la fel un slider orizontal prin care utilizatorul decide probabilitatea ca un neuron să fie activ în timpul antrenării. Această tehnică a fost prezentată și în 4.1.4.
- *Learning Rate*, este un text editabil prin care utilizatorul poate să decidă rată cu care se vor modifica ponderile în timpul antrenării.
- *OODA loop*, este un checkBox prin care se introduce modul de antrenare "rapid". Sliderul alăturat determină procentul din setul de exemple de intrare care va fi extras folosind algoritmul prezentat la 3.7. Inițial această funcționalitate este dezactivată dar poate fi activată de utilizator pentru a reduce timpul de antrenare.
- *Epochs*, este un text editabil în care utilizatorul menționează numărul de epoci rulate în timpul antrenării.
- *Current epoch*, este un câmp text needitabil care afișează epoca în care se află procesul de învățare.

Aplicația setează o configurație de parametrii recomandată la pornire, dar utilizatorul poate să facă diferite încercări modificând câmpurile sugerate mai sus. După ce parametrii au fost setați, prin apăsarea butonului *Start* se începe antrenarea modelului. Deoarece am dorit să prezint starea modelului în timp real pe parcursul învățării, a fost nevoie să creez un nou fir de execuție pentru antrenare după apăsarea butonului *Start*. La anumite momente modelul va genera un set de informații car vor fi transmise firului de execuție principal pentru a le ilustra în zonele grafice aferente. Dacă butonul care setează modul de execuție este în starea *Inference*, *app.radioButton_2*, apăsarea butonului *Start* va realiza clasificarea unuia din fiserele audio prezentă în directorul *Inference*.

```
def on_start_button_clicked(app):
    global thread_train
    if app.radioButton.isChecked():
        app.refresh_label_7()
        app.refresh_graphics_view()
        thread_train = Train_App(app)
        thread_train.print_accuracy_signal.connect(app.print_accuracy_graph)
        thread_train.print_stats.connect(app.print_stats_model)
        thread_train.print_matrix.connect(app.print_accuracy_matrix)
        thread_train.print_epoch.connect(app.print_label_19)
        thread_train.start()
    elif app.radioButton_2.isChecked():
        global ses, ser_inference_model, files
        vals = inference(ses, ser_inference_model, files ,
                         app.comboBox_2.currentText()) * 100
    pass
```

Algoritm 4.7: Metoda interfetei grafice apelata automat in urma apasarii butonului Start.

Train_App reprezintă noul fir de execuție cu ajutorul căruia se va antrena modelul. În secvența de cod de mai jos se poate observa cum clasa *Train_App* moștenește *QtCore.QThread*. Odată instanțiată această clasă apelează funcția *run*, care conține apelul către funcția *train*, care pornește procesul de învățare al modelului SER.

```
class Train_App(QtCore.QThread):
    print_accuracy_signal = QtCore.pyqtSignal(float)
    print_stats = QtCore.pyqtSignal(str)
    print_matrix = QtCore.pyqtSignal(object)
    print_epoch = QtCore.pyqtSignal(str)
    stopFlag = False
    def __init__(self, app_rnning, parent=None):
        QtCore.QThread.__init__(self, parent)
        self.app_rnning = app_rnning

    def run(self):
        train(self, int(self.app_rnning.lineEdit.text()),
                float(self.app_rnning.horizontalSlider_2.value()) / 10,
                float(self.app_rnning.horizontalSlider.value()) / 10,
                float(self.app_rnning.doubleSpinBox.value()) ,
                map_config[self.app_rnning.comboBox.currentText()],
                self.app_rnning.radioButton_3.isChecked())
```

Algoritm 4.8: Clasa aferenta firului de executie pentru procesul de antrenare.

Funcția train primește parametrii care crează contextul de execuție al modelului, prezentați anterior, și începe procesul de antrenare. În interiorul acestei funcții se resetează graficul de execuție *Tensorflow* pentru procesul actual, se crează nouă sesiune *Tensorflow* pe care vom rula modelul, se crează și apelează clasa care extrage caracteristicile de intrare din înregistrările audio și începe

procesul de învățare. La finalul antrenării este apelat modelul testor iar modelul antrenat este salvat pentru a putea fi folosit în modul de utilizare pentru inferență.

Statisticile sunt realizate în timp real prin intermediul informațiilor transmise din cadrul procesului de învățare. Librăria PyQt5 folosește conceptul de semnale pentru actualizarea informațiilor din interfețele grafice. Astfel un semnal este emis din firul de execuție pentru antrenare printr-o funcție care este "conectată" la cea care modifică interfața grafică. Această funcție "conectoare" face parte din clasa firului de execuție și modul de utilizare este ilustrat în secvența de cod 4.7.

Prima statistică ilustrată în interiorul aplicației este graficul care prezintă evoluția acurateții modelului în timpul antrenării. Acest grafic poate fi folosit pentru a determina anumite probleme care pot să apară în timpul antrenării sau pentru a determina cel mai favorabil număr de epoci.

Cea de a două statistică este una multiplă, fiind alcătuită din două pagini. Prima pagină are rolul de a înștiința utilizatorul legat de procesele care i-au parte în interiorul modelului, spre exemplu extragerea datelor dintr-un anumit set sau acuratețea atinsă pe datele de antrenare într-o anumită epoca, și este ilustrată în Fig 4.3. Cea de a două pagină, Fig 4.4, prezintă o matrice de confuzie care are pe coloane emoțiile clasificate, iar pe rânduri emoțiile adevărate. Astfel, pe diagonală se va afla numărul claselor identificate corect, iar pe restul pozițiilor din matrice se va afisa numărul de confuzii, exemple clasificate greșit, din fiecare caz.

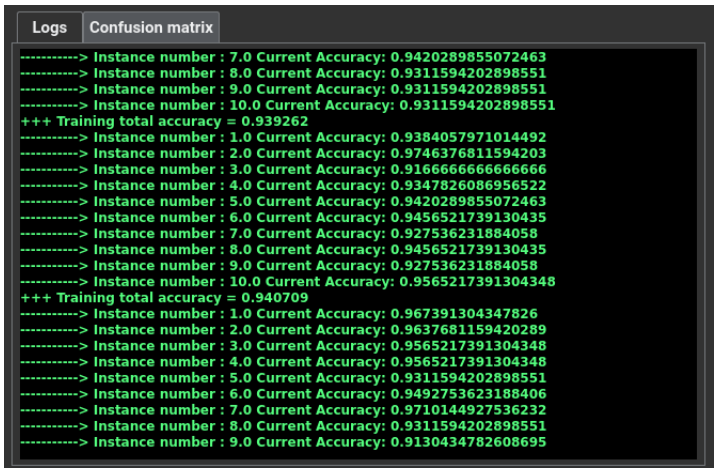


Figura 4.3: Logarea progresului antrenării.

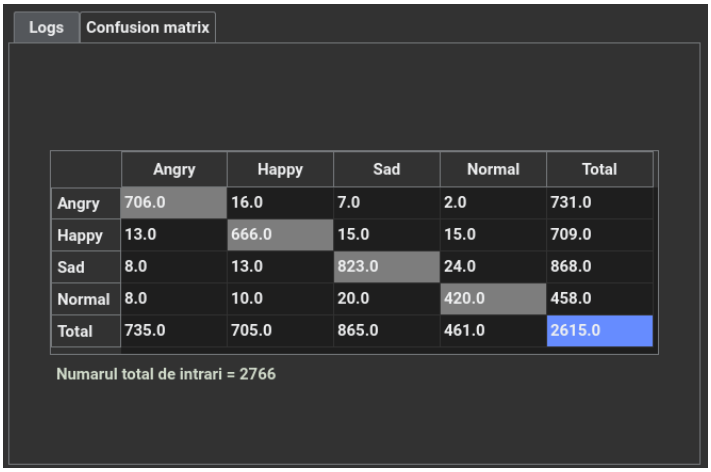


Figura 4.4: Matricea de confuzie a clasificatorului.

4.2.2 Interfața grafică în modul de infererența

Modul de utilizare inferențial este disponibil doar pentru extragerea datelor în varianta end-to-end. Din cauza limitărilor impuse de folosirea extragerii caracteristicilor în modul "hand-crafted", prezentate de-a lungul lucrării, nu am putut garanta ca setul de caracteristici propus să reușească să cuprindă în totalitate informația emoțională. Astfel, extragerea datelor în varianta "hand-crafted" este folosită doar pentru antrenare, iar acuratețea obținută este folosită că un grad de comparație pentru varianta propusă, end-to-end.

Interfața grafică este prezentată în Fig. 4.5. După cum se poate observa, după apăsarea butonului *Inference*, aplicația dezactivează toate funcționalitățile corespunzătoare modului de antrenare și le activează pe cele din modul inferență. În acest mod utlizatorul are posibilitatea să determine emoția din fișiere pre-înregistrate sau să își înregistreze pe loc propriile discursuri care vor fi clasificate automat după oprirea înregistrării.

În spațiul textului editabil *Folder path*, utilizatorul poate să introducă calea spre directorul în care se află fișierele audio pre-înregistrate pe care dorește să le clasifice. În mod implicit,

directorul folosit se numește "Inference" și este inclus în directorul în care se află proiectul. Combo box-ul *File* enumeră toate fișierele audio cu extensia .wav din acel director. Prin apăsarea butonului *Play* din dreapta combo box-ului utilizatorul poate auzi înregistrarea pe care dorește să o clasifice. Pentru clasificare însă, este nevoie că utilizatorul să aleagă fișierul dorit și apoi să apese pe butonul *Start*. În urma acestui șir de comenzi, probabilitățile aferente emoțiilor identificate vor fi ilustrate in barele de progres *Angry*, *Happy*, *Sad* și *Neutral*.

Recunoașterea emoției în timp real este posibilă prin apăsarea butonului *Record*. Odată apăsat acest buton, aplicația va începe să înregistreze prin microfonul integrat al calculatorului. Când utilizatorul a terminat de înregistrat discursul pe care dorește să îl clasifice, pentru a determina emoția, utilizatorul trebuie să apese butonul *Stop*. Butonul *Stop* va opri firul de execuție pentru inregistrare, va salva discursul într-un fișier audio și va oferi înregistrarea obținută modelului pentru clasificare. Ca și în cazul anterior, distribuția de probabilitate a emoțiilor va fi prezentată grafic în barele de progres aferente.

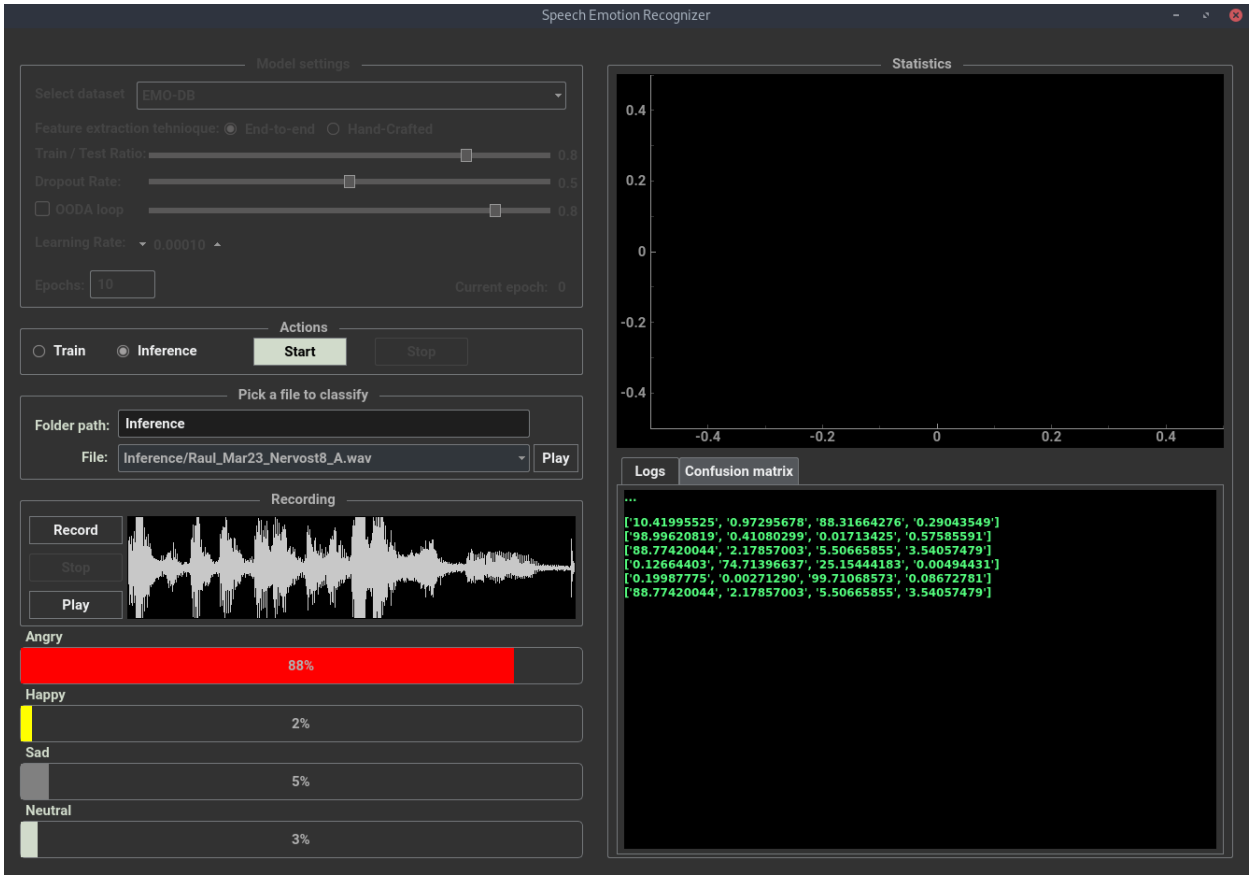


Figura 4.5: Interfața grafică în modul de inferență

În timpul înregistrării, semnalul audio va fi proiectat în timp real în interfața grafică, prin afișarea amplitudinilor acestuia. Pentru a continua înregistrarea și a actualiza interfața grafică în același timp, după apăsarea butonului *Record* se crează un nou fir de execuție special pentru procesul de înregistrare, asemănător cu crearea unui nou fir de execuție pentru procesul de antrenare prezentat în sub-capitolul 4.2.1. Inregistrearea semnalului audio este realizată folosind librăria PyAudio [12], iar proiectarea semnalului inregsitrat este realizată prin memorarea unui set de amplitudini ale acestuia și proiectarea lor folosind librăria PyQt5 la o frecvență calculată în funcție de rata de eșantionare a semnalului.

În timpul înregistrării pot există momente de liniște prelungite care dacă persistă pot influența precizia clasificatorului. Pentru a reduce aceste momente la un minim s-a folosit librăria webrtcvad

[11], care conține metode capabile să determine existența unui discurs într-un segment audio folosindu-se de anumite formule matematice asemănătoare cu cele prezentate în 3.4.1. Secvența de cod care realizează înregistrarea și această filtrare a vocii umane este ilustrată mai jos.

```
self.stream = pyaudio.PyAudio().open(format=self.sample_format,
                                     channels=self.channels,
                                     rate=self.rate,
                                     input=True,
                                     frames_per_buffer=self.chunk_size,
                                     stream_callback=self.process_new_frame)

self.vad = webrtcvad.Vad()
self.vad.set_mode(1)
```

Algoritm 4.9: Initializarea fluxului de transmitere a datelor pentru inregistrarea

Funcția *open* a librăriei PyAudio [12] pornește un flux de date provenite de la microfon, care după ce atinge un număr de segmente egal cu valoarea *self.chunk_size* îl transmite metodei *self.process_new_frame*. Deoarece aceste linii de cod realizează funcția de înregistrare, valoarea parametrului *input* este setată pe adevărat. Pentru a face funcția opusă, ascultarea unui fișier audio, fluxul de date este creat aproximativ în aceași metodă, modificandu-se valoarea parametrului input pe fals și setarea valoii parametrului output pe adevărat.

În câmpul *self.vad* se stochează o instanța a mecanismului de detectare a vocii și se setează severitatea filtrului pe valoarea 1, unde intervalul de agresivitate al acestuia este [0, 3].

```
def process_new_frame(self, data, frame_count, time_info, status):
    data = np.frombuffer(data, dtype=np.int16)
    with self.lock:
        if self.vad.is_speech(data, self.rate):
            self.frames.append(data)
        if self._print_frames_count == self._print_chunk_size:
            self.thread.print_recording_signal.emit(self._print_frames)
            self._print_frames = np.array([])
            self._print_frames_count = 0
        else:
            self._print_frames = np.concatenate((self._print_frames,data),
   axis=0)
            self._print_frames_count+=1
        if self.stop:
            return None, pyaudio.paComplete
return None, pyaudio.paContinue
```

Algoritm 4.10: Procesarea unui nou segmente al semnalului audio inregistrat. Daca segmentul nu contine discurs uman este exclus din inregistrarea finala.

Metoda *process_new_frame* primește în parametrul *data* un set de segmente audio de lungime *frame_count*. Aceste segmente sunt verificate de mecanismul de detecție prin metoda *self.vad.is_speech*, pentru existența unei vocii umane, și în caz pozitiv sunt adăugate la setul de segmente care urmează să fie clasificate de sistemul SER.

Semnalul audio înregistrat este proiectat în interfața grafică prin folosirea metodei *emit* a semnalului firului de execuție de înregistrare *self.thread.print_recording_signal*. În vectorul *self._print_frames* sunt salvate toate segmentele primite pentru a afișa întregul semnal în interfața audio.

La apăsarea butonului Stop din interfața grafică, câmpul *self.stop* va opri procesul de înregistrare generând salvarea semnalului într-un fișier audio de tip .wav și pornirea clasificatorului.

5 Rezultate si experimente

În acest capitol vor fi prezentate rezultatele obținute în urma executării unui set de experminente prin care soluția propusă este testată în diferite configurații. Rezultatele obținute sunt apoi comparate cu cele ale altor arhitecturi din domeniu. Deși sistemul SER a fost implementat special pentru lucrul cu mai multe baze de date într-o manieră "end-to-end", rezultatele obținute atât în cazurile în care s-a folosit o singură baza de date, cât și în cele în care extragerea datelor a fost de tip "hand-crafted", se mențin la nivelul celorlalte soluții SER.

Soluția propusă în această lucrare de diplomă este antrenată într-o manieră "multi-domain", pe înregistrări care provin din baze de date diferite. Antrenarea "multi-domain" prezintă o serie de avantaje, enumerate la 4.1.1, însă face ca și compararea arhitecturii cu alte sisteme SER să fie dificilă. Din acest motiv, în continuare vor fi enumerate rezultatele obținute de model în particular pentru unele dintre bazele de date care alcătuiesc setul descris la 4.1.1. Rezultatele obținute, pe fiecare din bazele de date cât și pe întregul set, pot fi studiate și în Tabela 5.1.

Antrenând modelul pe întreaga bază de date EMO-DB [62], cu înregistrări aferente a 7 emoții, acuratețea maximă înregistrată a fost de 77%, în timp ce acuratețea medie a fost de aproximativ 75%. Aceste rezultate sunt comparative cu cele obținute de Kerkeni et al., 2018 [75], unde folosind un modul clasificator similar, două celule recurente LSTM urmate de două nivele dense, s-a obținut o acuratețe maximă de 73% și una medie de 69.55%. Una din cele mai de succes soluții SER antrenate pe această baza de date este prezentată în Issa et al, 2020 [76], unde numărul înregistrărilor a fost mărit prin folosirea unor tehnici de augmentare precum adăugarea unui nou set de exemple obținut prin accelerarea la 1.23% a înregistrărilor, incetinrea la 0.81%, mutarea punctului de start sau adăugarea unui zgomot la anumite părți ale înregistrării. În urma acestei serii de augmentari, acuratețea modelului propus în Issa et al, 2020 [76] a atins precizia de 82.86%.

Baza de date RAVDESS [63] conține 1440 de înregistrări incorporând 8 emoții. Folosind înregistrări doar din această bază date pentru antrenare, modelul propus de către mine a obținut o acuratețe maximă de 71.08% și una medie de 68%. Acest rezultat este comparative cu precizia înregistrată din Issa et al, 2020 [76] de 71.61%. Zeng et al., 2017 [77] au obținut o acuratețe de 65.97% folosind rețele neuronale profunde și o antrenare tip "multi-task", antrenând modelul să clasifice emoții atât în vorbire cât și în cântece. Folosind un model clasficator bazat pe rețele neuronale convoluționale Popova et al., 2018 [78] au înregistrat o acuratețe de 71% pe această baze de date.

Realizând antrenarea calsificatorului pe 7 emoții folosind înregistrări din baza de date EMOVO [64] s-a atins o acuratețe maximă de 70%. Rezultate similare au fost obținute și în Latif et al., 2018 [79], acuratețe de 76.22%, unde clasificatorul sistemului SER a fost bazat pe o tipologie specială de rețele neuronale numite "Deep Belief Neural Networks". Latif et al., 2018 [79] propun și folosirea tehnicii "transfer learning", prezentată la 2.1.4, unde înaintea antrenarii pe baza de date EMOVO modelul este antrenat mai întâi pe baze de date din alte limbi. Această tehnică ii permite modelului să depășească valoarea inițială a acurateții atingând precizia de 80%.

Antrenând soluția propusă în această lucrare pe baza de date ENTERFACE'05 [66], care conține înregistrări reprezentative pentru un set de 6 emoții, acuratețea maximă obținută a fost 83.26%. Rezultate similare au fost obținute în Schuller, 2011 [80], unde prin folosirea uni clasifiator tradițional pentru probmela recunoașterii emoției, "Support Vector Machine", s-a atins o precizie de 62.8% pe baza de date ENTERFACE'05. Rezultate mai promițătoare au fost obținute în Ooi et al., 2014 [81], în care modelul propus înregistrează precizia 75.89%.

După cum se poate observa, chiar dacă soluția propusă este de tip "multi-domain", rezultatele

Tabela 5.1: Compararea acurateții înregistrate antrenând modelul propus pe bazele de date: EMO-DB, RAVDESS, EMOVO, ENTERFACE'05 și pe întregul set de baze de date menționat la 4.1.1. Antrenarea a fost realizata în două moduri, pe întregul set de emoții al fiecărei baze de date cât și pe cele patru emoții principale: enervare, tristețe, fericire și neutru.

Baza de date	Acuratețea pe întregul set de emoții	Acuratețea pe cele patru emoții
EMO-DB	77%	91.04%
RAVDESS	71.08%	76.11%
EMOVO	70%	80.59%
ENTERFACE'05	83.26%	90.05%
Întregul set de baze de date	-	84.21%

obținute de sistemul SER implementat în această lucrare se apropie considerabil de cele obținute de implementări care sunt specializate pe câte una dintre acestea. Prin reducerea numărului de exemple pentru a cuprinde doar setul de emoții enumerat la 2.3 (fericire, tristețe, enervare și neutru), acuratețea pe fiecare dintre bazele de date depășește rezultatele prezentate mai sus. În cazul bazei de date EMO-DB acuratețea maximă obținută pe cele 4 emoții este de 91.04%, în cazul RAVDESS 76.11%, EMOVO 80.59% și ENTERFACE'05 90.05%.

Un alt experiment a fost realizat prin modificare modului de extragere a caracteristicilor de intrare. Metoda "end-to-end", 3.4, este cea propusă pentru arhitectura finală a sistemului SER implementat în această lucrare, totuși pentru a evidenția importanța acestei tehnici în continuare vor fi prezentate rezultatele înregistrate pentru clasificare bazată pe caracteristici "hand-crafted". Acuratețea maximă înregistrată în cadrul acestui experiment a fost de 68.56%.

Soluția propusă cuprinde atât o antrenare tip "multi-domain" cât și o extragere a caracteristicilor de intrare "end-to-end". Rezultatele obținute folosind această configurație finală sunt încurajatoare, fiind asemănătoare cu cele obținute de o altă arhitectură SER antrenată într-o modalitate similară. Acuratețea neponderată maximă obținută de modelul propus în această lucrare a fost de 84.1%, având o valoare medie de aproximativ 82%. Aceste rezultate le depășesc cu aproximativ 15% pe cele obținute folosind caracteristicile "hand-crafted", susținând beneficiile extragerii "end-to-end". În același timp, deși era de așteptat că acuratețea modelului să scadă folosind mai multe baze de date din cauza diferențelor puternice dintre limbile și modalitățile de înregistrare ale acestora, se poate observa cum acuratețea se menține, și chiar depășește preciziile înregistrare pe bazele de date RAVDESS si EMOVO. Sistemul SER prezentat în Milner et al., 2019 [42] profită de generalitatea obținută prin antrenarea "multi-domain" într-o arhitectură SER similară cu cea implementată în această lucrare, 2.2.1. Acuratețea neponderată obținută în acest studiu a fost de 82.26% pe 6 emoții diferite, fiind apropiată de cea obținută de mine chiar dacă setul de baze de date și numărul de emoții clasificate diferă.

Din punct de vedere arhitectural soluția propusă poate fi comparată cu alte câteva din domeniul SER. Folosind o singură baza de date, dar același mod de extragere a caracteristicilor de intrare, soluția prezentată în Li, Yuanchao et al., 2019 [33], a atins o precizie de 82.8%, fiind una dintre cele mai înalte măsurate pe baza de date IEMOCAP [68]. Misramadi et al., 2017 [37], au introdus un modul clasificator bazat pe rețele neuronale recurente bidirecționale combinate cu un mecanism de atenție, similar cu cel din acest proiect, și au obținut o acuratețe de 63.5% pe baza de date IEMOCAP [68]. Alte rezultate cu arhitecturi apropiate de cea propusă de mine au fost: Kerkeni et al. (2018) [70] unde s-a înregistrat o acuratețe de 82.14% pe baza de date EMO-DB [62], Fonnegra et al., 2018 [71] au obținut rezultate promițătoare folosind baza de date ENTERFACE'05 [66] cu o acuratețe 92%, Lim et al., 2016 [72] au executat mai multe experminete pe baza de date

Tabela 5.2: Compararea acurateții obținute prin antrenarea modelului folosind modul "rapid", bazat pe ciclurile OODA, reducând numărul de exemple de antrenare în ordinea descrescătoare a erorii, OODA-ERROR, și aleator, OODA-RANDOM.

Tipul de reducere folosit	Acuratețea înregistrată
OODA-ERROR	71%
OODA-RANDOM	80%

EMO-DB [62] obținând precizii de 87.74% într-o arhitectură cu rețele convoluționale, 79.87% într-o arhitectură cu rețele neuronale recurente și 88.01% combinând cele două tipologii de rețele neuronale într-o manieră asemănătoare cu arhitectura folosită în această lucrare de diplomă.

Alte experimente au fost realizate de către mine pentru a se încerca reducerea timpului de antrenare al modelului păstrând însă o eroare tolerabilă. Prin folosirea ciclurilor OODA, 3.7, la un anumit număr de epocii se extrage un set de exemple de antrenare până se atinge proporția rămasă dorită. Setul de exemple extrase a fost deteminat în două moduri: în ordinea descrescătoare a erorii fiecărei înregistrări și aleatoriu. În primul caz, diferența dintre acuratețea înregistrată și cea din modul de antrenare normal este mult prea mare, obținând o precizie de maxim 71%, prezentată în Tabela 5.2, și una medie de 68%. Cel de-al doilea caz prezintă rezultate mai promițătoare, acuratețea maximă înregistrată fiind de 80%, prezentată în Tabela 5.2, iar cea medie de 78%. Timpul de antrenare cu un număr de 80 de epocii este redus la 3 ore și 30 de minute de la 5 ore in cazul folosirii întregului set din baza de date. Aceste rezultate au permis introducerea unui nou mod de antrenare "rapid", prin care modelul își păstrează arhitectura inițială iar precizia se menține la un nivel satisfăcător. Acest mod de învățare este unul opțional fiind adresat utilizatorilor care nu pot aștepte întreagă durata necesară antrenării dar pot să accepte o mică scădere a preciziei.

6 Concluzii

Lucrarea de diplomă conține atât o soluție pentru recunoașterea emoției în vorbire cât și o interfață grafică care permite antrenarea, testarea și inferența eficientă a sistemului SER. Domeniul recunoașterii emoției în vorbire nu este până în acest moment unul "cucerit", prezentând o gama largă de obstacole care nu au permis obținerea unei acurateți destul de satisfăcătoare pentru a permite acestor algoritmi să fie introduși pe piață. Totuși, soluția implementată de mine reușește să atingă o acuratețe comparabilă cu a unor arhitecturi de success din domeniu și chiar să le depășească în anumite configurații. Această acuratețe este pusă apoi în practică prin interfață grafică care permite utilizatorului să clasifice emoțiile din interiorul discursurilor provenite din mai multe surse, fișiere preînregistrare sau înregistrate pe loc.

Sistemul SER propus încearcă să determine o arhitectură eficientă pentru rezolvarea recunoașterii de emoții în vorbire intergrand diferite tehnici și concepte cu scopul de a cuprinde cât mai complet complexitatea problemei. Alegerea folosirii unui set de mai multe baze de date este justificată de îmbunătățirea generalității modelului Machine Learning. Extragerea datelor în manieră "end-to-end" este motivată de lipsa unui set de caracteristici audio de intrare specializate pe recunoașterea emoțiilor în vorbire. Implementarea unui modul clasificator bazat pe rețele neuronale recurente este aleasă pentru a profita de relațiile temporale dintre emoțiile din segmente audio aflate la momente diferite. Concatenarea unui mecanism de atenție la rețeaua recurentă este bazată pe filtrarea segmentelor lipsite de emoție pentru a reduce inconsistențele introduse de acestea.

În construirea acestui proiect am folosit multe dintre tehnicile învățate în decursul facultății de Automatică și Calculatoare, precum programarea orientată pe obiecte, dezvoltarea interfetelor grafice, programarea concurentă și diferite concepte de inteligență artificială. Totuși, pentru realizarea unui sistem SER am necesitat unele noțiuni specifice, ca care au fost obținute prin studiul unor articole științifice din domeniu, enumerate în decursul lucrării. Tehnologiile folosite au crescut în număr cu mulțimea de functinalitati adăugate incluzând limbajul de programare Python, celebra librărie Tensorflow pentru dezvoltarea aplicațiilor Machine Learning, Librosa [53] pentru extragerea informațiilor auditive, webrtcvad [11] pentru identificarea semgentelor care conțin voce umană și PyAudio [12] pentru înregistrarea și redarea fișierelor audio.

Scopul final al unui sistem SER este acela de a fi introdus în interfețele de comunicare om-mașină din viitor, pentru a oferii mașinilor capacitatea de a înțelege conversațiile la care iau parte și într-un context emoțional. Integrarea acestor algoritmi va crește calitatea conversațiilor, permițând ca interfețele de comunicare om-mașină să atingă o calitate asemănătoare cu cele de la om la om. Până a ajunge în acel punct, algoritmii de recunoaștere a emoțiilor trebuie să mai treacă printr-o serie de imbunatariri pentru a crește gradual acuratețea înregistrată. Soluția propusă de mine reprezintă o arhitectură într-o stare incipientă, având potențialul de a fi extinsă prin introducerea mai multor tehnici.

Prima modalitate de îmbunătățire a sistemului SER propus, și cea mai simplă, ar fi mărirea numărului de baze de date folosite pentru a amplifica generalitatea modelului, pentru a combata problema numărului redus de exemple la antrenare și pentru a mării numărul de emoții clasificate.

O altă modificare ar putea fi introducerea unui modul care să diminueze diferențele dintre înregistrările provenite din baze de date diferite. Algoritmi care realizează această sarcina au fost deja introduși în alte soluții din domeniu. De exemplu Deng et al. , 2014 [73] au folosit cu succes o tehnică numită "adaptive denoising-autoencoders", care învață să determine și să elimine diferențele dintre mai multe baze de date prin aducerea înregistrărilor acestora la o formă asemănătoare cu cele dintr-o anumită baza de date țintă. Alte tehinici de reducere a discrepanțelor bazelor de date sunt normalizarea semnalului audio per bază de date sau normalizare per vorbitor,

Bjorn et al, 2010 [27].

Tehinca antrenării pe mai multe sarcini, "multi-task", poate la rândul ei să aducă imbunatațiri puternice modelului clasificator. Prin antrenarea modelului pe o serie de sarcini simultan, acesta poate să devină inflexibil la variații ale semnalului audio care nu ar trebui să influențeze emoția clasificată. Li, Yuanchao et al., 2019 [33] au antrenat arhitectură SER propusă, atât pe recunoașterea emoțiilor în vorbire, cât și pe determinarea sexului vorbitorului, în timp ce Milner et al., 2019 [42] au folosit ca sarcină secundară determinarea bazei de date din care face parte înregistrarea curentă. Aceste două tehnici s-au dovedit a fi avantajoase în combatarea influenței sexului vorbitorului cât și a bazei de date de proveniență în decursul procesului de clasificare.

O altă extindere a proiectului curent ar putea fi combinarea soluției propuse cu un algoritm de detecție a emoției vizuale. Astfel produsul final va putea determina emoția umană folosindu-se de două tipuri diferite de stimului. Această metodă s-a demonstratat a fi avantajoasă în difeite articole ca Tzirakis et al., 2014 [31] sau Sana et al., 2010 [74], unde s-a depășit acuratețea sistemului SER inițială prin adăugarea informației vizuale în clasificarea emoțiilor.

Proiectul meu de diplomă prezintă astfel o soluție incipientă pentru una din problemele care au rămas încă nerezolvate în domeniul inteligenței artificiale, recunoașterea emoției în vorbire. Acesta include atât un model Machine Learning clasificator, cât și o interfață grafică care permite utilizatorului să încerce diferite configurații de parametrii, să observe statistici detaliate ale proceselor care iau parte în timpul antrenării și să încerce modelul antrenat pe înregistrări din diferite surse. Chiar dacă sistemul SER prezentat nu este momentan viabil pentru a fi introdus pe piață, rezultatele obținute sunt comparabile cu cele ale unora dintre cele mai de succes implementari din domeniul recunoașterii de emoții în vorbire din prezent. Rezultatele obținute sunt încurajatoare si susțin faptul că soluția Machine Learning propusă are potențialul de a depași precizia curentă prin aplciarea mai multor tehnici deja exisitente în acest domeniu.

Bibliografie

[1] P. J. Bavel. *Fourier Transform*. URL: http://www.thefouriertransform.com/.

[2] *... Transformata Fourier*. 2018. URL: https://ro.wikipedia.org/wiki/Transformata_Fourier.

[3] Julius Smith. *Spectral Audio Signal Processing*. Jan. 2008.

[4] *... Short-time Fourier transform*. 2018. URL: https://en.wikipedia.org/wiki/Short-time_Fourier_transform.

[5] James Lyons. *Mel Frequency Cepstral Coefficient (MFCC) tutorial*. 2013. URL: http://practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs//#computing-the-mel-filterbank.

[6] Haytham Fayek. *Speech Processing for Machine Learning: Filter banks, Mel-Frequency Cepstral Coefficients (MFCCs) and What's In-Between*. 2016. URL: https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html.

[7] *... Mel-frequency cepstrum*. 2019. URL: https://en.wikipedia.org/wiki/Mel-frequency_cepstrum.

[8] *... Mel scale*. 2020. URL: https://en.wikipedia.org/wiki/Mel_scale.

[9] *... Periodogram*. 2019. URL: https://en.wikipedia.org/wiki/Periodogram.

[10] *... OODA loop*. 2020. URL: https://en.wikipedia.org/wiki/OODA_loop.

[14] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2006. ISBN: 0387310738.

[15] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012. ISBN: 0262018020.

Referinte

[11] John Wiseman. *Py-webrtcvad*. 2019. URL: https://github.com/wiseman/py-webrtcvad.

[12] Hubert Pham. *PyAudio*. 2006. URL: https://pypi.org/project/PyAudio/.

[13] Michell Stuttgart. *qdarkgraystyle*. 2019. URL: https://github.com/mstuttgart/qdarkgraystyle.

[16] Smiley Blanton. "The voice and the emotions". In: *Quarterly Journal of Speech - QUART J SPEECH* 1 (Jan. 1915), pp. 154–172. DOI: 10.1080/00335631509360475.

[17] Stephen Levinson et al. "The origin of human multi-modal communication". In: *Philosophical transactions of the Royal Society of London. Series B, Biological sciences* 369 (Sept. 2014). DOI: 10.1098/rstb.2013.0302.

[18] Morten Christiansen et al. "Language Evolution: Consensus and Controversies". In: *Trends in cognitive sciences* 7 (Aug. 2003), pp. 300–307. DOI: 10.1016/S1364-6613(03)00136-0.

[19] Frank Dellaert et al. “Recognizing Emotion In Speech”. In: *International Conference on Spoken Language Processing, ICSLP, Proceedings* 3 (Dec. 1996).

[20] Xu Huahu et al. “Application of Speech Emotion Recognition in Intelligent Household Robot”. In: Nov. 2010, pp. 537 –541. DOI: 10.1109/AICI.2010.118.

[21] Purnima Gupta et al. “Two-stream emotion recognition for call center monitoring.” In: Jan. 2007, pp. 2241–2244.

[22] Mariusz Szwoch et al. “Emotion Recognition for Affect Aware Video Games”. In: Jan. 2015, pp. 227–236. ISBN: 978-3-319-10661-8. DOI: 10.1007/978-3-319-10662-5_28.

[23] Diana Van Lancker Sidtis et al. “Recognition of emotionalprosodic meanings in speech by autistic, schizophrenic, and normal children”. In: *Developmental Neuropsychology - DEVELOP NEUROPSYCHOL* 5 (Jan. 1989), pp. 207–226. DOI: 10.1080/87565648909540433.

[24] Björn Schuller. “Speech emotion recognition: Two decades in a nutshell, benchmarks, and ongoing trends”. In: *Communications of the ACM* 61 (Apr. 2018), pp. 90–99. DOI: 10.1145/3129340.

[25] Soujanya Poria et al. *MELD: A Multimodal Multi-Party Dataset for Emotion Recognition in Conversations*. Oct. 2018.

[26] Shashidhar Koolagudi. “Emotion recognition from speech: A review”. In: *International Journal of Speech Technology* 15 (June 2012). DOI: 10.1007/s10772-011-9125-1.

[27] Björn Schuller et al. “Cross-Corpus Acoustic Emotion Recognition: Variances and Strategies”. In: *IEEE Transactions on Affective Computing* 1 (July 2010), pp. 119–131. DOI: 10.1109/T-AFFC.2010.8.

[28] Tin Nwe et al. “Speech Emotion Recognition Using Hidden Markov Models”. In: *Speech Communication* 41 (Nov. 2003), pp. 603–623. DOI: 10.1016/S0167-6393(03)00099-2.

[29] Marc Schröder et al. “Issues in emotion-oriented computing towards a shared understanding”. In: 2006.

[30] A. Graves et al. “Towards end-to-end speech recognition with recurrent neural networks”. In: *31st International Conference on Machine Learning, ICML 2014* 5 (Jan. 2014), pp. 1764–1772.

[31] Panagiotis Tzirakis et al. “End-to-End Multimodal Emotion Recognition Using Deep Neural Networks”. In: *IEEE Journal of Selected Topics in Signal Processing* PP (Apr. 2017). DOI: 10.1109/JSTSP.2017.2764438.

[32] Zixing Zhang et al. “Attention-augmented End-to-end Multi-task Learning for Emotion Prediction from Speech”. In: May 2019, pp. 6705–6709. DOI: 10.1109/ICASSP.2019.8682896.

[33] Yuanchao Li et al. “Improved End-to-End Speech Emotion Recognition Using Self Attention Mechanism and Multitask Learning”. In: *INTERSPEECH*. 2019.

[34] George Trigeorgis et al. “Adieu features? End-to-end speech emotion recognition using a deep convolutional recurrent network”. In: Mar. 2016, pp. 5200–5204. DOI: 10.1109/ICASSP.2016.7472669.

[35] P. Tzirakis et al. “End-to-End Speech Emotion Recognition Using Deep Neural Networks”. In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2018, pp. 5089–5093.

[36] Berkehan Akçay et al. “Speech emotion recognition: Emotional models, databases, features, preprocessing methods, supporting modalities, and classifiers”. In: *Speech Communication* 116 (Jan. 2020). DOI: 10.1016/j.specom.2019.12.001.

[37] S. Mirsamadi et al. “Automatic speech emotion recognition using recurrent neural networks with local attention”. In: *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2017, pp. 2227–2231.

[38] Adrian Rosebrock. *Autoencoders with Keras, TensorFlow, and Deep Learning*. 2020. URL: https://www.pyimagesearch.com/2020/02/17/autoencoders-with-keras-tensorflow-and-deep-learning/.

[39] Linlin Chao et al. “Improving generation performance of speech emotion recognition by denoising autoencoders”. In: *Proceedings of the 9th International Symposium on Chinese Spoken Language Processing, ISCSLP 2014* (Oct. 2014), pp. 341–344. DOI: 10.1109/ISCSLP.2014.6936627.

[40] Jun Deng et al. “Autoencoder-based Unsupervised Domain Adaptation for Speech Emotion Recognition”. In: *Signal Processing Letters, IEEE* 21 (Sept. 2014), pp. 1068–1072. DOI: 10.1109/LSP.2014.2324759.

[41] J. Deng et al. “Sparse Autoencoder-Based Feature Transfer Learning for Speech Emotion Recognition”. In: *2013 Humaine Association Conference on Affective Computing and Intelligent Interaction*. 2013, pp. 511–516.

[42] Rosanna Milner et al. “A Cross-Corpus Study on Speech Emotion Recognition”. In: Dec. 2019. DOI: 10.1109/ASRU46091.2019.9003838.

[43] Gilles Degottex et al. “COVAREP: A Collaborative Voice Analysis Repository for Speech Technologies”. In: May 2014. DOI: 10.1109/ICASSP.2014.6853739.

[44] Rubén Fonnegra et al. “Speech Emotion Recognition Based on a Recurrent Neural Network Classification Model”. In: Jan. 2018, pp. 882–892. ISBN: 978-3-319-76269-2. DOI: 10.1007/978-3-319-76270-8_59.

[45] Jinkyu Lee et al. “High-level Feature Representation using Recurrent Neural Network for Speech Emotion Recognition”. In: Sept. 2015.

[46] Y. Bengio et al. “Learning long-term dependencies with gradient descent is difficult”. In: *IEEE Transactions on Neural Networks* 5.2 (1994), pp. 157–166.

[47] Sepp Hochreiter et al. “Long Short-term Memory”. In: *Neural computation* 9 (Dec. 1997), pp. 1735–80. DOI: 10.1162/neco.1997.9.8.1735.

[48] Sepp Hochreiter et al. “Learning To Learn Using Gradient Descent”. In: Sept. 2001, pp. 87–94. DOI: 10.1007/3-540-44668-0_13.

[49] Facundo Bre et al. “Prediction of wind pressure coefficients on building surfaces using Artificial Neural Networks”. In: *Energy and Buildings* 158 (Nov. 2017). DOI: 10.1016/j.enbuild.2017.11.045.

[50] Oleksii Trekhleb. *Playing with Discrete Fourier Transform Algorithm in JavaScript*. 2018. URL: https://dev.to/trekhleb/playing-with-discrete-fourier-transform-algorithm-in-javascript-53n5.

[51] Nasser Kehtarnavaz. “CHAPTER 7 - Frequency Domain Processing”. In: *Digital Signal Processing System Design (Second Edition)*. Ed. by Nasser Kehtarnavaz. Second Edition. Burlington: Academic Press, 2008, pp. 175 –196. ISBN: 978-0-12-374490-6. DOI: https://doi.org/10.1016/B978-0-12-374490-6.00007-6. URL: http://www.sciencedirect.com/science/article/pii/B9780123744906000076.

[52] Leila Kerkeni et al. “Speech Emotion Recognition: Methods and Cases Study”. In: Jan. 2018, pp. 175–182. DOI: 10.5220/0006611601750182.

[53] Brian McFee et al. “librosa: Audio and Music Signal Analysis in Python”. In: Jan. 2015, pp. 18–24. DOI: 10.25080/Majora-7b98e3ed-003. URL: https://librosa.github.io/librosa/.

[54] Sergey Ioffe et al. “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”. In: (Feb. 2015).

[55] Y. LeCun et al. “Backpropagation Applied to Handwritten Zip Code Recognition”. In: *Neural Computation* 1.4 (1989), pp. 541–551.

[56] Krut Patel. *MNIST Handwritten Digits Classification using a Convolutional Neural Network (CNN)*. 2019. URL: https://towardsdatascience.com/mnist-handwritten-digits-classification-using-a-convolutional-neural-network-cnn-af5fafbc35e9.

[57] Fei-Fei Li et al. *CS231n: Convolutional Neural Networks for Visual Recognition*. 2020. URL: https://cs231n.github.io/convolutional-networks/.

[58] Ian Goodfellow et al. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[59] Aurlien Gron. *Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. 1st. OReilly Media, Inc., 2017. ISBN: 1491962291.

[60] Volodymyr Mnih et al. “Recurrent Models of Visual Attention”. In: *Advances in Neural Information Processing Systems* 3 (June 2014).

[61] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: https://www.tensorflow.org/.

[62] Astrid Paeschke et al. “F0-CONTOURS IN EMOTIONAL SPEECH”. In: 1999.

[63] Steven R. Livingstone et al. “The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS): A dynamic, multimodal set of facial and vocal expressions in North American English”. In: *PLOS ONE* 13.5 (May 2018), pp. 1–35. DOI: 10.1371/journal.pone.0196391. URL: https://doi.org/10.1371/journal.pone.0196391.

[64] Giovanni Costantini et al. “EMOVO Corpus: an Italian Emotional Speech Database”. In: May 2014. ISBN: 9782951740884.

[65] Pascal Belin et al. “The Montreal Affective Voices: A validated set of nonverbal affect bursts for research on auditory affective processing”. In: *Behavior research methods* 40 (May 2008), pp. 531–9. DOI: 10.3758/BRM.40.2.531.

[66] O. Martin et al. “The eNTERFACEŠ05 Audio-Visual Emotion Database”. In: Feb. 2006, pp. 8 –8. ISBN: 0-7695-2571-7. DOI: 10.1109/ICDEW.2006.145.

[67] Li Tian. *JL corpus*. 2018. URL: https://www.kaggle.com/tli725/jl-corpus.

[68] Carlos Busso et al. “IEMOCAP: Interactive emotional dyadic motion capture database”. In: *Language Resources and Evaluation* 42 (Dec. 2008), pp. 335–359. DOI: 10.1007/s10579-008-9076-6.

[69] Diederik Kingma et al. “Adam: A Method for Stochastic Optimization”. In: *International Conference on Learning Representations* (Dec. 2014).

[70] Leila Kerkeni et al. “Speech Emotion Recognition: Methods and Cases Study”. In: Jan. 2018, pp. 175–182. DOI: 10.5220/0006611601750182.

[71] Rubén Fonnegra et al. “Speech Emotion Recognition Based on a Recurrent Neural Network Classification Model”. In: Jan. 2018, pp. 882–892. ISBN: 978-3-319-76269-2. DOI: 10.1007/978-3-319-76270-8_59.

[72] Wootaek Lim et al. “Speech emotion recognition using convolutional and Recurrent Neural Networks”. In: Dec. 2016, pp. 1–4. DOI: 10.1109/APSIPA.2016.7820699.

[73] Jun Deng et al. “Autoencoder-based Unsupervised Domain Adaptation for Speech Emotion Recognition”. In: *Signal Processing Letters, IEEE* 21 (Sept. 2014), pp. 1068–1072. DOI: 10.1109/LSP.2014.2324759.

[74] Sana ul haq et al. “Multimodal Emotion Recognition”. In: *Machine Audition: Principles, Algorithms and Systems* (Jan. 2010). DOI: 10.4018/978-1-61520-919-4.ch017.

[75] Leila Kerkeni et al. “Speech Emotion Recognition: Methods and Cases Study”. In: Jan. 2018, pp. 175–182. DOI: 10.5220/0006611601750182.

[76] Dias Issa et al. “Speech emotion recognition with deep convolutional neural networks”. In: *Biomedical Signal Processing and Control* 59 (2020), p. 101894. ISSN: 1746-8094. DOI: https://doi.org/10.1016/j.bspc.2020.101894. URL: http://www.sciencedirect.com/science/article/pii/S1746809420300501.

[77] Yuni Zeng et al. “Spectrogram based multi-task audio classification”. In: *Multimedia Tools and Applications* 78 (Dec. 2017). DOI: 10.1007/s11042-017-5539-3.

[78] Anastasiya Popova et al. “Emotion Recognition in Sound”. In: vol. 736. Jan. 2018, pp. 117–124. ISBN: 978-3-319-66603-7. DOI: 10.1007/978-3-319-66604-4_18.

[79] Siddique Latif et al. “Transfer Learning for Improving Speech Emotion Classification Accuracy”. In: Sept. 2018, pp. 257–261. DOI: 10.21437/Interspeech.2018-1625.

[80] Björn Schuller. “Affective Speaker State Analysis in the Presence of Reverberation”. In: *International Journal of Speech Technology* 14 (June 2011), pp. 77–87. DOI: 10.1007/s10772-011-9090-8.

[81] Chien Shing Ooi et al. “A new approach of audio emotion recognition”. In: *Expert Systems with Applications* 41.13 (2014), pp. 5858 –5869. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2014.03.026. URL: http://www.sciencedirect.com/science/article/pii/S0957417414001638.