

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

Sistem automat de labiolectură

Proiect de diplomă

**Prezentată ca cerință parțială pentru obținerea
titlului de *Inginer*
în domeniul *Inginerie Electronică și Telecomunicații*
programul de studii *Electronică Aplicată***

Conducător științific

Prof. Dr. Ing. Bogdan – Emanuel IONESCU

Absolvent

George – Cristian CIOFLAN

Anul 2018

TEMA PROIECTULUI DE DIPLOMĂ
a studentului **CIOFLAN N. George-Cristian, 444B**

1. Titlul temei: Sistem automat de labiolectură

2. Descrierea contribuției originale a studentului (în afara părții de documentare):

Contribuția originală va consta în dezvoltarea unor tehnici de analiză și prelucrare a informației video și de învățare asistată de calculator, folosind rețele neuronale adânci, cu obiectivul de a analiza automat particularitățile vorbirii în secvențe video. Sistemul vizat va presupune existența unui set de cuvinte cheie în limba română, pe care algoritmul le va putea recunoaște. Aplicația finală presupune traducerea mișcărilor buzelor în cuvinte. Cercetarea aferentă va include: sinteza bibliografică a realizărilor actuale din domeniu, contribuții teoretice la nivel de algoritmi, simularea acestora și validarea experimentală, conchiderea rezultatelor obținute și a contribuției originale, cât și enunțarea perspectivelor ulterioare de cercetare.

3. Resurse folosite la dezvoltarea proiectului:

Limbaje de programare: Matlab, Python; Biblioteci: OpenCV.

4. Proiectul se bazează pe cunoștințe dobândite în principal la următoarele 3-4 discipline:

Programare Obiect-Orientată, Decizie și Estimare în Prelucrarea Informațiilor, Imagistică Medicală.

5. Proprietatea intelectuală asupra proiectului aparține: studentului


6. Data înregistrării temei: 2017-11-29 17:41:02

Conducător(i) lucrare,

Prof. dr. ing. Bogdan Emanuel IONESCU

semnătura: 

Student,

semnătura: 

Director departament,

Prof. dr. ing Sever PAȘCA

semnătura: 

Decan,

/ Prof. dr. ing. Cristian NEGRESCU

semnătura: 

Cod Validare: **026a5e47b8**

Declarație de onestitate academică

Prin prezenta declar că lucrarea cu titlul **Sistem automat de labiolectură**, prezentată în cadrul Facultății de Electronică, Telecomunicații și Tehnologia Informației a Universității „Politehnica” din București ca cerință parțială pentru obținerea titlului de *Inginer* în domeniul Inginerie Electronică și Telecomunicații/Calculatoare și Tehnologia Informației, programul de studii *Electronică Aplicată* este scrisă de mine și nu a mai fost prezentată niciodată la o facultate sau instituție de învățământ superior din țară sau străinătate. Declar că toate sursele utilizate, inclusiv cele de pe Internet, sunt indicate în lucrare, ca referințe bibliografice. Fragmentele de text din alte surse, reproduse exact, chiar și în traducere proprie din altă limbă, sunt scrise între ghilimele și fac referință la sursă. Reformularea în cuvinte proprii a textelor scrise de către alți autori face referință la sursă. Înțeleg că plagiatul constituie infracțiune și se sancționează conform legilor în vigoare. Declar că toate rezultatele simulărilor, experimentelor și măsurărilor pe care le prezint ca fiind făcute de mine, precum și metodele prin care au fost obținute, sunt reale și provin din respectivele simulări, experimente și măsurători. Înțeleg că falsificarea datelor și rezultatelor constituie fraudă și se sancționează conform regulamentelor în vigoare.

București, Iunie 2018

Absolvent: George – Cristian CIOFLAN

.....


Cuprins

Introducere	15
Capitolul 1 Fundamente teoretice	19
1.1 Principiile labiolecturii	19
1.1.1 Considerente generale.....	19
1.1.2 Influența lingvisticii în labiolectură.....	20
1.1.3 Concluzii.....	22
1.2 Rețele neuronale artificiale.....	23
1.2.1 Instruire asistată de calculator	23
1.2.2 Principiul de funcționare al rețelelor neuronale artificiale	24
1.2.3 Istoricul rețelelor neuronale artificiale	25
1.2.4 Rețele neuronale adânci.....	28
1.2.5 Rețele neuronale convoluționale	30
1.2.6 Proprietățile rețelelor neuronale convoluționale.....	32
1.3 Biblioteci specifice	34
1.3.1 TensorFlow.....	34
1.3.2 Keras.....	35
1.3.3 OpenCV	35
1.4 Rezultate anterioare.....	36
Capitolul 2 Arhitectura sistemului propus	39
2.1 Achiziția video	41
2.2 Identificarea unui cuvânt	41
2.2.1 Metoda conturului.....	42
2.2.2 Metoda histogramei	42

2.2.3 Metoda ariei	43
2.3 Identificarea regiunii de interes	44
2.3.1 Metoda directă	44
2.3.2 Metoda indirectă	44
2.3.3 Metoda indirectă cu memorie	45
2.3.4 Metoda cu cadru fix	45
2.4 Recunoașterea cuvântului	46
2.5 Gramatica	46
2.6 Video supratitrat	47
2.7 Rețeaua antrenată	47
2.7.1 Rețeaua Inception-V3 & Multilayer Perceptron	47
2.7.2 Inception-V3 & Long short-term memory	48
2.7.3 Long-term recurrent convolutional network	50
2.7.4 Convolutional 3D	51
Capitolul 3 Rezultate experimentale	53
3.1 Descrierea seturilor de date	53
3.2 Metrice de evaluare	56
3.2.1 Acuratețea	56
3.2.2 Word Error Rate	57
3.3 Analiza rețelelor neuronale adânci	57
3.4 Analiza metodelor de identificare a cuvintelor	61
3.5 Analiza metodelor de identificare a regiunii de interes	62
3.6 Rezultate finale	62
Concluzii și perspective	65
Bibliografie	67
Anexa 1	71
Diplomă obținută la Sesiunea de Comunicări Științifice Studențești	71
Anexa 2	73
Implementarea identificării regiunii de interes	73

Lista figurilor

Figura 1.1: Strategii de învățare automată	24
Figura 1.2: Schema de principiu a unei rețele neuronale artificiale [10].....	25
Figura 1.3: Topologia perceptronului [12].....	27
Figura 1.4: Funcția semi-liniară (ReLU).....	29
Figura 1.5: Schema principală a unei rețele neuronale convoluționale [17].....	33
Figura 2.1: Organigrama sistemului automat de labiolectură propus	40
Figura 2.2: Determinarea ariei gurii cu ajutorul celor 68 de repere faciale.. ..	43
Figura 2.3: Selecția regiunii de interes prin metoda cu cadru fix.. ..	45
Figura 2.4: Rețeaua Inception-V3 & MLP.....	48
Figura 2.5: Rețeaua Inception-V3 & LSTM.	49
Figura 2.6: Rețeaua LRCN.....	50
Figura 2.7: Rețeaua C3D.....	51
Figura 3.1: Subiecții setului de date CAMPUS – α LIRO	55
Figura 3.2: Variația acurateței în funcție de batch size.....	58
Figura 3.3: Variația acurateței în funcție de lungimea secvenței.....	59
Figura 3.4: Rezultatele utilizării metodei histogramei.....	61
Figura 3.5: Rezultatele utilizării metodei ariei.....	62
Figura 3.6: Etichetarea video-ului inițial folosind rezultatele sistemului	63

Lista tabelelor

Tabel 1.1: Seturi de date disponibile pentru antrenarea sistemelor automate de labiolectură	37
Tabel 3.1: Descrierea seturilor de date utilizate pentru antrenare.....	54
Tabel 3.2: Relația dintre dimensiunea secvențelor folosite la antrenare, acuratețea recunoașterii și setul de date.....	59
Tabel 3.3: Variația acurateței în funcție de dimensiunea imaginii	60
Tabel 3.4: Rezultatele metodelor de identificare a regiunii de interes.....	62
Tabel 3.5: Rezultatele sistemului comparativ cu cele ale unor subiecți umani.	64

Lista acronimelor

AI – Artificial Intelligence

ALR – Automated Lipreading System

AOI – Area Of Interest

ANN – Artificial Neural Network

API – Application Programming Interface

CAMPUS – **αLIRO** - Centrul de Cercetări Avansate pentru Materiale, Produse și Procese Inovative – αLipreading ROmanian

CNN – Convolutional Neural Network

CPU – Central Processing Unit

GPU – Graphics Processing Unit

CV – Computer Vision

DL – Deep Learning

DNN – Deep Neural Network

FPS – Frames Per Second

LSTM – Long Short-Term Memory

LRCN – Long-term Recurrent Convolutional Network

ML – Machine Learning

MLP – Multilayer Perceptron

RNN – Recurrent Neural Network

ROI – Region Of Interest

TF – Tensorflow

WER – Word Error Rate

Introducere

Încă din secolul XX, folosirea unor sisteme automate în viața de zi cu zi a oamenilor a devenit tot mai răspândită, conducând inerent la dezvoltarea societății în care trăim. În zilele noastre, realiarea unei interfețe om-mașină (IOM) este unul dintre domeniile de foarte mare interes, numeroși cercetători ocupându-se de această arie, ale cărei utilizări devin tot mai răspândite. Pe măsură ce gradul de utilizare al tehnologiei crește, este necesar ca această interacțiune să devină tot mai intuitivă, prin creșterea gradului de similitudine cu modul în care ființele umane interacționează. Pentru ca acest lucru să devină posibil, abordarea actuală a problemei propune ca în comunicarea dintre om și mașină să se utilizeze cât mai multe dintre modurile în care oamenii transferă informații. Pentru aceasta, este necesar ca mașinile, precum roboții sau calculatoarele de uz personal, să perceapă și să interpreteze cât mai mult din informațiile primite din mediul înconjurător, cu o acuratețe cât mai mare.

Pentru a putea realiza un nivel de conștientizare de către mașini a mediului înconjurător, au fost dezvoltate sisteme capabile să achiziționeze date din exterior într-un mod similar oamenilor [1], prin intermediul senzorilor și al camerelor video. Pe lângă achiziționarea datelor, mașinile trebuie să fie capabile să le proceseze și să le interpreteze. Pentru aceste etape, se consideră că utilizarea tehnicilor de tip Machine Learning (ML), cunoscută și ca instruire asistată de calculator, este cea mai adecvată metodă. În particular, folosirea algoritmilor de tip Deep Learning (DL), altfel spus rețele neuronale adânci, s-a dovedit a fi extrem de utilă în acest domeniu.

Dintre modurile în care se realizează interacțiunea om-mașină amintim comunicarea pe cale orală, care este facilitată prin intermediul unor sisteme de recunoaștere a vorbirii. Deoarece acest tip de comunicare este cel mai comun între ființele umane, este firesc ca direcția de evoluție a sistemelor tehnologice să fie una în care interacțiunea prin intermediul vorbirii să fie cea mai utilizată. În fapt, recunoașterea vorbirii este o temă de interes încă de la jumătatea secolului trecut, atunci când Fry prezenta aspectele teoretice ale unui sistem mecanic de recunoaștere a vorbirii[2]. În ultimii 50 de ani, numeroase implementări ale unor astfel de sisteme, hardware sau software, au fost propuse și realizate.

Dificultatea în cazul realizării unui sistem de recunoaștere automată a vorbirii o prezintă robustețea și acuratețea sistemului. O astfel de soluție software cu acuratețe de 100% nu a fost încă

dezvoltată și este foarte improbabil să se obțină în viitorul apropiat. Din acest considerent, s-a încercat adăugarea unor aplicații suplimentare, care să conducă la creșterea ratei de recunoaștere. O astfel de abordare au avut-o Silsbee et al. [9], care au dezvoltat un sistem audiovizual cunoscut ca „*Lipreading to Enhance Automatic Perception of Speech (LEAPS)*”, un sistem de labiolectură folosit la îmbunătățirea recunoașterii automate a vorbirii.

În mod firesc, următoarea etapă în dezvoltarea aplicațiilor de acest tip a reprezentat-o realizarea unei soluții software de sine stătătoare, un sistem capabil să recunoască cuvinte exclusiv pe baza cadrelor dintr-o filmare. Rațiunea din spatele acestei dezvoltări o reprezintă evoluția domeniului achiziției și procesării imaginilor, cât și numărul tot mai mare de imagini care sunt captate la fiecare moment de timp. În zilele noastre, se estimează că peste 1000 de fotografii sunt realizate în fiecare secundă, însemnând cel puțin un cadru pe milisecundă. Mai mult, dacă luăm în considerare echipamente precum camerele de supraveghere și ținem cont de numărul de cadre pe secundă pe care acestea sunt capabile să le înregistreze, suntem obligați să multiplicăm numărul mai sus amintit cu cel puțin un milion. Asta înseamnă, deci, că suntem înconjurați de informație vizuală, informație care trebuie nu doar achiziționată, ci și procesată și vizualizată.

Au apărut astfel aplicațiile [3][4] de tip Automated Lipreading Recognition (ALR), sau sisteme de citit automat pe buze, care au cunoscut un avans puternic în ultimii 30 de ani. Cele mai recente sisteme ating niveluri de acuratețe tot mai ridicate, un exemplu concludent în acest sens fiind proiectul lui Assael et al. [5], capabil să recunoască cuvinte în limba engleză cu o precizie de 95.2%, depășind astfel atât experții umani, cât și cei mai performanți algoritmi dezvoltați până la acest moment.

La momentul actual, cercetările făcute arată că nu există un sistem automat de labiolectură (SAL) creat pentru a detecta și recunoaște cuvinte în limba română. Se impune așadar dezvoltarea și implementarea unei astfel de aplicații, dat fiind faptul că există numeroase contexte în care aceasta ar putea fi folosită. Astfel, recunoașterea unor cuvinte cheie dintr-o frază poate fi utilizată fie pentru a realiza o interfață om-mașină într-un mediu cu nivel ridicat de zgomot, fie pentru implementarea unui sistem național de supraveghere, în care imaginile de la camerele stradale de supraveghere pot fi interpretate de sistem, care alertează autoritățile la apariția unui astfel de cuvânt. În plus, un sistem mai avansat, capabil să recunoască orice cuvânt definit în Dicționarul Explicativ Român, ar putea fi integrat în aplicații de transpunere a imaginilor în scris, folosite pentru a subtitra programele Televiziunii Naționale Române sau, împreună cu un sistem de recunoaștere a vorbirii, pentru a facilita comunicarea între oameni. Rezultatele obținute la nivel internațional în acest domeniu demonstrează că un o astfel de aplicație este realizabilă, chiar cu o acuratețe înaltă, ceea ce face ca prezentul proiect să fie unul fezabil.

Ne propunem, așadar, să realizăm un sistem automat de labiolectură pentru limba română, capabil să realizeze atât detecția, cât și recunoașterea unor cuvinte cheie, folosind exclusiv informație vizuală. Acest proiect are ca punct de plecare lucrarea prezentată în cadrul „Sesiunii de Comunicări Științifice Studențești UPB, Mai, 2018”, după cum se poate observa în Anexa 1. Pentru prima parte a proiectului, vom realiza un algoritm de procesare a imaginilor, astfel încât să putem selecta din filmul înregistrat doar acele cadre care reprezintă informație dorită, altfel spus cadrele care compun

cuvintele pronunțate de vorbitor. A doua etapă a prezentei lucrări o reprezintă recunoașterea cuvintelor amintite anterior, pe baza unei rețele neuronale adânci deja antrenate. Pentru aceasta, este necesar să realizăm respectiva rețea, urmând mai apoi să o antrenăm folosind un set de date ce conține cuvinte în limba română.

Proiectul cuprinde, astfel, următoarele etape:

- Studiul noțiunii de labiolectură și a tehnicilor folosite pentru a realiza acest proces, studiul metodelor de ML folosind DL ce urmează a fi implementate în cadrul proiectului și analiza tehnologiilor folosite în lucrare, a căror funcționare va fi descrisă în Capitolul 1.
- Implementarea sistemului capabil să citească pe buze, a cărei arhitectură va fi descrisă în Capitolul 2. În acest capitol se vor analiza punctele critice din lanțul de preprocesare a datelor, urmând a se oferi soluții viabile și robuste. De asemenea, se vor propune anumite structuri de rețele neuronale ce ar putea fi folosite în etapa de recunoaștere, împreună cu modul în care acestea pot fi antrenate.
- Evaluarea performanțelor sistemului realizat, atât la nivel de bloc, cât și în integralitatea sa, va fi descrisă în Capitolul 3.

În finalul lucrării vom prezenta atât concluziile proiectului, cât și anumite îmbunătățiri care îi vor fi aduse în viitor.

Capitolul 1

Fundamente teoretice

1.1 Principiile labiolecturii

1.1.1 Considerente generale

Conform Dicționarului Explicativ Român, labiolectura reprezintă „, aptitudinea de a înțelege vorbirea după mișcarea buzelor”. Contextele în care această abilitate a fost utilizată cu succes sunt numeroase și diverse. Folosind un sistem automat, Frank Hubner [5] a reușit să reconstruiască dialogurile din filmele mute înregistrate de Eva Braun la reședința lui Adolf Hitler, realizând astfel un documentar care oferă informații noi despre personalitatea și modul de viață a liderului nazist. Un alt exemplu de utilizare a unui astfel de sistem în cinematografie îl reprezintă identificarea dialogurilor din filmele mute, pentru a se analiza dacă actorii aveau sau nu un scenariu. Nu în ultimul rând, există procese penale [5] în care experți umani în labiolectură au reușit, prin analiza unor înregistrări anterioare crimei, să determine vinovăția acuzatului.

Este important de menționat că, deși sunt încă folosiți în curțile de judecată, acuratețea experților umani nu este extrem de ridicată. Deoarece comunicarea pe cale orală este prin definiție multimodală, labiolectura reprezintă doar o etapă a comunicării, informația necesară pentru înțelegerea vorbirii fiind conținută și în gesturi, mimică, limbaj corporal sau tonalitate. Este așadar dificilă reconstrucția informației transmise dacă aceasta a fost recepționată exclusiv prin intermediul mișcării buzelor. Cu toate acestea, importanța înțelegerii pe această cale a cuvintelor transmise de interlocutor nu este una marginală, așa cum o demonstrează efectul McGurk [7]. Astfel, în 1976, Harry McGurk și John MacDonald au demonstrat influența pe care o are vizualizarea vorbitorului în

înțelegerea mesajului transmis de acesta. Prin suprapunerea unei înregistrări audio cu o persoană ce pronunța silaba [ba] peste o înregistrare video cu aceeași persoană care rostea silaba [ga], subiecții experimentului au răspuns că persoana înregistrată spunea [da]. Pe de altă parte, în momentul în care subiecților li s-a arătat exclusiv înregistrarea video, aceștia au catalogat-o cu acuratețe ca reprezentând o rostire a cuvântului [ga]. Este așadar imperios necesar să înțelegem principiile și mecanismele labiolecturii, dată fiind importanța pe care aceasta o are în transmiterea mesajului dorit.

1.1.2 Influența lingvisticii în labiolectură

Pentru a putea înțelege pe deplin noțiunile ce stau la baza labiolecturii, trebuie să realizăm o analiză lingvistică a limbii române. În particular, în cadrul acestei lucrări vom analiza structura fonetică a limbii române, fonetica fiind cea disciplină a lingvisticii ce are ca obiect de studiu elementele fonice, i.e. sunetele, produse și/sau receptate în cadrul procesului de comunicare interumană. Deoarece prezentul proiect nu își propune să prelucrez informație auditivă, este deci suficient să facem referire la fonetica articulatorie, ramura foneticii ce are în vedere proprietățile elementelor fonice din limbă, din perspectiva modului în care ele sunt produse de către vorbitor. Din același considerent, cel al utilizării unei informații strict vizuale, vom analiza numai modul de producere al elementelor fonice segmentale, acele unități fonice ce nu caracterizează, ci constituie segmentul fonic: consoane și vocale.

Pentru a putea descrie corect modul în care sunt produse consoanele și vocalele, este necesară cunoașterea modului în care se realizează actul fonator. Principala contribuție în acest proces o are laringele, cu toate că aparatul fonator este format din mai multe organe articulatorii: plămâni, trahee, laringe, cavitate bucală și cavitate nazală. Apare astfel prima diferență între vocale și consoane: în timp ce la articularea vocalelor curentul fonator iese nemijlocit, vibrația aerului fiind dată doar de vibrația coardelor vocale, în cazul consoanelor articularea este dată de existența unui obstacol pe traseul curentului fonator. Ținând cont de mecanismele ce intervin în procesul de producere a sunetelor, cât și de această diferență dintre consoane și vocale, Corniță [8] realizează următoarea clasificare a sunetelor:

A) Clasificarea consoanelor:

- a. După modul de articulare (având în vedere dacă coardele vocale vibrează sau nu, dacă dacă intervine sau nu cavitatea nazală în producerea sunetului și analizând modul în care curentul fonator întâlnește un obstacol):
 - i. Explozive: sunete pentru care se realizează o ocluziune totală - /b, d, g, k, p, t/;
 - ii. Fricative: sunete pentru care se realizează o fricțiune dată de îngustarea tubului fonator - /f, h, j, s, v, z/;
 - iii. Africate: sunete pentru care se realizează o ocluziune urmată de o fricțiune - /č, ģ/;
 - iv. Sonante vibrante și laterale: suntete pentru care se realizează o ocluziune intermitentă - /l, r/;
 - v. Sonante nazale: sunete pentru care se realizează o ocluziune la nivelul cavității bucale, curentul fonator fiind eliminat prin fosele nazale- /m, n/.

b. După locul de articulare:

- i. Bilabiale: sunete pentru care se realizează o apropiere sau o unire a buzelor - /b, m, p/;
- ii. Labiodentale: sunete pentru care se realizează o apropiere a buzei inferioare de incisivii superiori - /f, v/;
- iii. Dentale: sunete pentru care se realizează o atingere cu varful limbii a interiorului incisivilor superiori - /d, s, z, t/;
- iv. Dentale-Alveolare: sunete pentru care se realizează o atingere cu vârful limbii a primei zonei a boltei palatine - /l, n, r/;
- v. Prepalatale: sunete pentru care se realizează împingerea limbii în zona anterioară a boltei palatine - /č, ģ, j/;
- vi. Velare: sunete pentru care se realizează retragerea limbii în zona posterioară a boltei palatine - /g, k/;
- vii. Laringale: sunet pentru care se realizează strâmtarea laringelui - /h/.

B) Clasificarea vocalelor

a. După unghiul de deschidere a maxilarelor, denumit apertură:

- i. Deschise: sunete pentru care deschiderea maxilarelor este amplă - /a/;
- ii. Medii: sunete pentru care unghiul scade - /e, i/;
- iii. Închise: sunete pentru care unghiul este cel mai redus - /o, u/.

b. După locul de articulare:

- i. Palatale: sunete pentru care curentul fonator este direcționat spre zona anterioară a boltei palatine - /e, i/;
- ii. Centrale: sunete pentru care curentul fonator este direcționat spre zona centrală a boltei palatine - /a/;
- iii. Palatale: sunete pentru care curentul fonator este direcționat spre zona posterioară a boltei palatine - /o, u/.

c. După gradul de participare a buzelor:

- i. Labializate: sunete pentru care buzele realizează modelarea sunetului - /o, u/;
- ii. Nelabializate: sunete în formarea cărora buzele nu participă - /a, e, i/.

Dacă analizăm această clasificare, observăm că există anumite proprietăți ale elementelor fonice ce constituie limba română, proprietăți care creează, la rândul lor, dificultăți în înțelegerea unui mesaj transmis de interlocutor folosind exclusiv labiolectura:

A) Există perechi de sunete ale căror caracteristici tipice foneticii descriptive sunt identice (e.g. /s/ și /z/, /f/ și /v/ sau /l/ și /r/). Distingerea între sunetele aparținând uneia dintre aceste perechi este imposibilă prin simpla observare a modului în care buzele unui vorbitor se mișcă pentru a realiza pronunția.

B) Se observă prezența unor grupuri de foneme care, deși au proprietăți diferite, sunt caracterizate de aceleași mișcări ale aparatului fonator superior (e.g. /č/, /ģ/ și /j/, pentru care diferă modul de articulare, fapt însă neobservabil de un subiect uman din cauza faptului că ocluziunea care se produce în pronunțarea fonemelor /č/ și /ģ/ este mascată de apropierea incisivilor). Acest

aspect conduce, încă o dată, la dificultăți în recunoașterea fonemului folosind exclusiv informația vizuală.

- C) Există grupuri de sunete pentru care, deși grafemul (i.e. semnul grafic prin care un fonem este reprezentat) este diferit, fonemul este, în fapt, același (e.g. grafemul [c] din cuvântul „coala” și grafemul [k] din cuvântul „koala” sunt ambele reprezentate prin intermediul unui unic fonem, și anume /k/). În această situație, deducem că este necesară ca informație suplimentară nu doar reprezentarea acustică a fonemului, ci întreg morfemul, acea unitate elementară care poartă cu sine un înțeles.

Observăm că, spre deosebire de situațiile A) și B), unde interveneau doar reprezentările audio-vizuale ale fonemelor, reprezentări bazate pe fonetică, în cazul prezentat la punctul C) trebuie realizată interpretarea datelor utilizând noțiuni de semantică, apelând deci la sensul cuvintelor. Prin lărgirea ariei de analiză se desprind astfel noi situații ce îngreunează labiolectura:

- D) Omofonele (i.e. cuvinte care se pronunță la fel, fără a se scrie identic) conduc la erori în determinarea cuvântului rostit în absența unui context mai larg în care cuvântul să fie încadrat (e.g. „ea” și „ia” sau „deal”, „de-al” și „de-a-l”).

1.1.3 Concluzii

Se poate observa, așadar, că există două mari categorii de obstacole în realizarea unei lecturi labiale cu precizie mare. Prima categorie face referire la labiolectura ideo-vizuală, bazată exclusiv pe perceperea mișcărilor faciale și labiale, și încadrează situațiile menționate la punctele A), B) și C). O analiză independentă a labiolemelor (i.e. a pozițiilor limbii și buzelor în timpul pronunțării unor sunete) este deci insuficientă, fiind necesar să interpretăm și labiolemele învecinate pentru a putea determina cu acuratețe elementul fonic pronunțat. Astfel, în absența stimulilor acustici și fără a face apel la semantică, este imposibilă discriminarea între cuvinte precum „mere”, „pere” și „bere” sau „fată” și „vată”. Este relevant de menționat faptul că nu există un termen în limba română care să definească cuvintele identice din punctul de vedere al lecturii labiale ideo-vizuale. Această absență, spre deosebire de limba engleză, acolo unde întâlnim termenul „*viseme*”, ne arată incidența scăzută a unor astfel de situații în fonetica limbii române.

Cea de-a doua categorie de dificultăți ce pot apărea în procesul de labiolectură se referă la labiolectura vizual-fonetică și ilustrează situația menționată la punctul D). Astfel, în acest caz, chiar și prezența unor stimuli acustici devine insuficientă, cuvinte precum „s-au” și „sau” având o reprezentare fonemică identică. Devine deci necesar să recurgem din nou la sensul cuvintelor, sens dedus din contextul în care ele au fost folosite. Trebuie însă amintit faptul că limba română are, în mare măsură, o ortografie fonemică, cu puține abateri (e.g. perechea de consoane /ks/ se pronunță la fel în „îmbâcsit” și în „axă”). Din acest aspect deducem că există un număr relativ redus de omofone în limba română, comparativ cu limba engleză. Putem așadar conchide, ținând cont și de argumentul adus anterior, că, din perspectiva foneticii celor două limbi, procesul de labiolectură se poate realiza cu o acuratețe mai mare în limba română față de limba engleză.

1.2 Rețele neuronale artificiale

1.2.1 Instruire asistată de calculator

Termenul de „Instruire asistată de calculator” provine din sintagma în limba engleză „Machine Learning” (ML) și reprezintă un subdomeniu al Inteligenței Artificiale (eng. „Artificial Intelligence” – AI). În ultimii 30 de ani, acest domeniu a devenit unul dintre cele mai puternice centre de interes în sfera tehnologiei informației și, prin aplicațiile și sistemele dezvoltate, o parte integrantă a vieții de zi cu zi. Date fiind atât cantitatea, cât și diversitatea datelor care ne înconjoară și care, implicit, ne sunt disponibile spre analiză, procesarea și vizualizarea datelor sunt, și vor continua să fie, o parte esențială a dezvoltării societății din care facem parte.

Necesitatea existenței unor sisteme implementate prin intermediul noțiunilor de ML vine în urma apariției unor probleme insuficient definite, probleme care necesită o rezolvare. Un exemplu concludent în acest sens îl reprezintă ordonarea rezultatelor oferite de un motor de căutare. Este evident că această sortare trebuie făcută în funcție de relevanța fiecărei pagini web, relativ la cuvintele cheie pe care utilizatorul le-a folosit pentru a realiza interogarea. Gradul de relevanță poate fi influențat de mai multe caracteristici, precum numărul de accesări ale paginii la căutări cu cuvinte cheie similare, conținutul articolelor spre care motorul de căutare indică sau numărul de apariții în pagină al termenilor folosiți de utilizator. Este deci necesară implementarea unui algoritm cu un grad mare de adaptabilitate, capabil să furnizeze un rezultat cât mai bun pentru o problemă, problemă a cărei rezolvare nu urmărește un set clar de reguli.

Apare astfel noțiunea de ML, explicată de Kovahi și Provost [9] ca reprezentând „o disciplină științifică care explorează construcția și studiul algoritmilor care învață din date”. Li se oferă astfel calculatoarelor puterea de a învăța și de a răspunde la anumite probleme puse, fără a urmări instrucțiuni explicit oferite, ci folosind un set de date pe baza cărora sistemul oferă decizii ori predicții. Este eliminată astfel necesitatea aducerii îmbunătățirilor în mod manual de către programator. Dat fiind faptul că sistemele de tip ML realizează modele bazate pe intrările puse la dispoziție, ieșirile unor astfel de sisteme se vor modifica corespunzător la apariția unui nou tip de intrări, fără a fi nevoie de o modificare ulterioară a codului scris.

În literatura de specialitate, domeniul ML este divizat în trei categorii vaste, ale căror scheme de principiu sunt prezentate în Figura 1.1. Învățarea supervizată se bazează pe un set de date de intrare etichetate de către programator, setul de antrenare conținând astfel atât datele de intrare furnizate sistemului, cât și răspunsurile corecte aferente lor. Putem deci compara acest principiu de funcționare cu minimizarea unei funcții de eroare, calculată între răspunsurile pe care sistemul le produce și răspunsurile corecte. În comparație cu prima metodă, învățarea nesupervizată folosește un set de antrenare lipsit de etichete, astfel că sistemul nu cunoaște ieșirile pe care ar trebui să le furnizeze. Pornind de proprietățile statistice ale datelor de intrare, metoda se bazează pe calitatea modelului pe care îl extrage din date, calitate care trebuie să crească pe parcursul antrenării, astfel încât să se realizeze gruparea datelor în funcție de gradul de similitudine dintre acestea. În opoziție cu strategiile

prezentate anterior, învățarea cu întărire se bazează pe feedback-ul primit de sistem de la mediu, în urma realizării unor sarcini oferite spre execuție. În funcție de corectitudinea cu care sistemul și-a îndeplinit sarcina, el primește fie o recompensă (i.e. întărire pozitivă), fie este penalizat (i.e. întărire negativă), scopul final fiind maximizarea recompenselor.

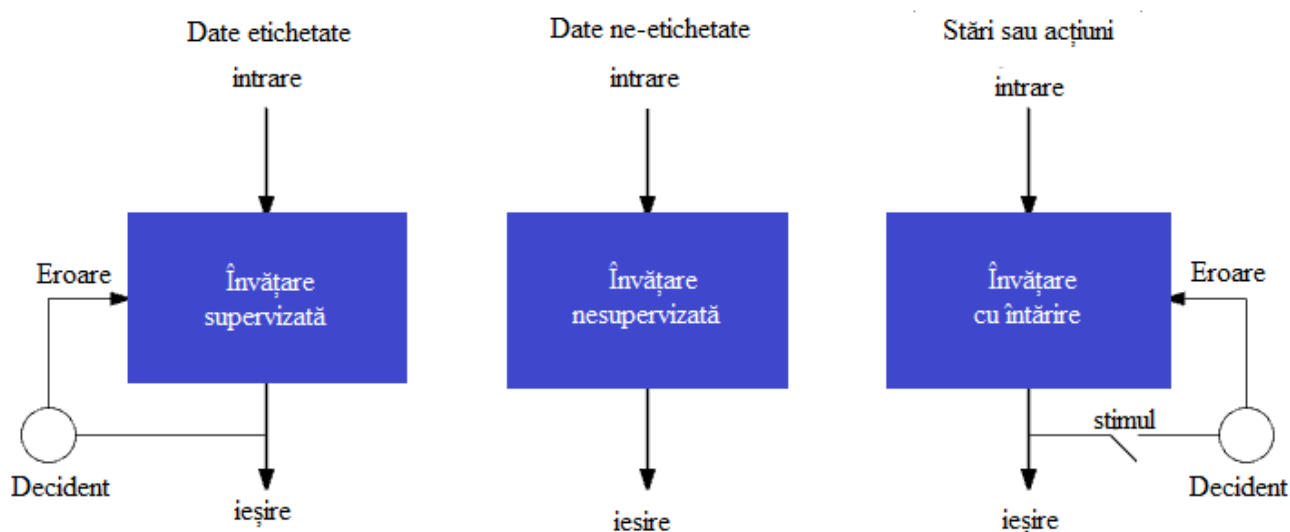


Figura 1.1: Strategii de învățare automată

Datorită modului de funcționare, învățarea supervizată prezintă un set de avantaje ce merită menționat. Astfel, definirea claselor se poate face cu foarte multă precizie prin etichetarea datelor de intrare conform cu sarcina pe care sistemul o are de îndeplinit. Algoritmul poate fi deci antrenat să distingă cu acuratețe între clasele care îi sunt furnizate, numărul acesta fiind stabilit de programator. Nu în ultimul rând, odată antrenat, sistemul nu mai are nevoie de setul de date de intrare care i-a fost pus la dispoziție, ci doar de relația matematică ce diferențiază clasele. Pe de altă parte, trebuie descrise dezavantajele pe care această metodă de antrenare le prezintă. Astfel, cantitatea datelor de intrare influențează puternic calitatea modelului obținut de sistem, fiind necesar un set de date etichetate cât mai vast pentru a realiza o clasificare cu acuratețe mare. De asemenea, pe un set de date de dimensiuni reduse, sistemul este foarte probabil să ajungă la supraantrenare, devenind astfel incapabil să clasifice corect date noi de intrare. În plus, pentru intrări ce nu aparțin claselor deja definite, strategia de învățare supervizată va produce întotdeauna răspunsuri greșite. Analizând avantajele și dezavantajele acestei metode, considerăm oportună alegerea ei ca principiu de bază pentru implementarea unui sistem automat de labiolectură, capabil să recunoască un set redus de cuvinte cheie.

1.2.2 Principiul de funcționare al rețelelor neuronale artificiale

Rețelele neuronale artificiale reprezintă un subdomeniu al instruirii asistate de calculator, un subdomeniu care a luat amploare în ultimii ani ca urmare a avansului tehnologic pe care producătorii de echipamente hardware l-au realizat. Cunoscute în engleză sub termenul de Artificial Neural Networks (ANN), ele reprezintă o unealtă puternică de clasificare și predicție. Privite ca un *black*

box, ANN primesc la intrare un set de date etichetat, ieșirea lor fiind reprezentată de un vector de predicții, exprimate ca o distribuție de probabilități ale fiecărei clase. Analizând principial, ele pot fi descrise ca reprezentând o rețea cu multiple straturi, pe fiecare strat aflându-se neuroni artificiali. Astfel, o rețea va fi constituită dintr-un strat de intrare, stratul prin care setul de intrare este furnizat rețelei, un număr oricât de mare de straturi ascunse și un strat de ieșire, cel prin care rețeaua oferă mediului rezultate. Pentru oricare strat, se poate afirma că modul de funcționare este similar și poate fi descris ca fiind format din trei etape: intrarea datelor în rețea, calculul unei sume ponderate pe baza intrărilor oferite și, în final, aplicarea unei funcții matematice rezultatului obținut la pasul anterior. Dat fiind faptul că pașii unu și doi sunt ambii realizați de către o singură unitate funcțională, denumită neuron, și că rezultatul furnizat de a treia etapă poate reprezenta fie ieșirea sistemului, fie intrarea într-un alt strat, Kon Mamadou Tadiou, citat de Maqableh [10], propune următoarea schemă principială a unui ANN, schemă expusă în Figura 1.2.

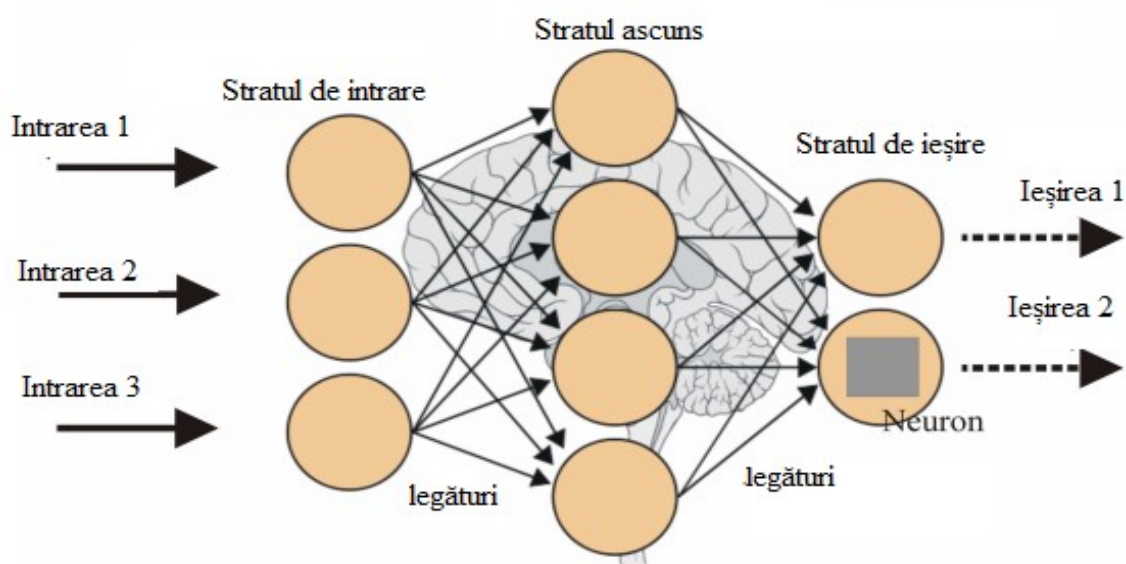


Figura 1.2: Schema de principiu a unei rețele neuronale artificiale [10]

1.2.3 Istoricul rețelelor neuronale artificiale

Istoria rețelelor neuronale artificiale este strâns legată de descoperirile realizate în domeniul neurologiei. Abilitățile pe care creierul uman le posedă au determinat cercetătorii care aveau ca arie de interes tehnologia informației să realizeze un sistem capabil să emuleze proprietățile creierului. Astfel, este important de menționat caracterul asociativ al memoriei umane, accesul la informație făcându-se, spre deosebire de o arhitectură clasică, tip von Neumann, printr-o adresare la prin conținut. Discutând tot despre modul în care informația este stocată, neurologii au descoperit că există o distribuție a cunoștințelor în întreaga structură neuronală. Analizând modul în care se realizează transferul informației, s-a observat faptul că există un număr extrem de mare, raportat la numărul de magistrale de adrese și/sau date al unui calculator, de conexiuni interneuronale, acestea fiind în număr de aproximativ 10^{15} [11]. O altă comparație relevantă dintre creierul uman și o arhitectură clasică a unui sistem de calcul se poate face din punctul de vedere al numărului unităților de procesare

disponibile. Astfel, în timp ce un calculator poate prezenta un număr relativ redus de procesoare, creierul uman dispune de aproximativ 10^{11} [11] unități funcționale, cunoscute ca neuroni. S-a încercat, așadar, replicarea acestei structuri cât mai fidel cu putință, astfel încât să se obțină rezultate comparabile cu cele ale creierului uman.

Primele rezultate notabile s-au obținut în anul 1943, atunci când Warren McCulloch și Walter Pitts reușesc să formuleze primul model de rețea neuronală artificială cu capabilități computaționale. Acela a fost, de altfel, primul model matematic al unui neuron biologic. Șase ani mai târziu, în 1949, Donald Hebb oferă baza teoretică ce descrie modul în care neuronii răspund adaptiv pe parcursul procesului de învățare. El arată că activarea simultană a celulelor presinaptică și postsinaptică conduce la creșterea permeabilității sinaptice, implicit la creșterea eficienței cu care informația este transmisă. Pornind de la teoriile formulate de cercetătorii menționați anterior, Frank Rosenblatt introduce în anul 1957 prima implementare a unei ANN: perceptronul. Acesta reprezintă, în fapt, cea mai simplă implementare a unei rețele neuronale artificiale și este baza de la care au pornit toate celelalte realizări în acest domeniu. Din perspectiva ML, perceptronul funcționează după principiul învățării supervizate, fiind astfel, de asemenea, prima implementare a unei ANN capabile să învețe.

Având în vedere că, principal, o rețea neuronală artificială poate fi explicată prin intermediul perceptronului simplu, vom descrie în cele ce urmează modul de funcționare al acestuia, folosindu-ne de topologia sa, prezentată în Figura 1.3. Fie setul de date de intrare $in(t)$, definit de intrările de la x_1 la x_n , denumite și trăsături. Fie de asemenea setul de ponderi definit prin mulțimea $\{w_1, w_2, \dots, w_n\}$. Fiecare pondere corespunde câte unei unități funcționale, mulțimea acestor unități reprezentând stratul de intrare. La nivelul unității funcționale, se realizează însumarea intrărilor, ponderate cu ponderile aferente. Pentru a se obține comportamentul unei funcții de gradul I, se introduce de asemenea un termen de prepolarizare (i.e. *bias*), notat în topologie cu θ . Astfel, prin modificarea termenului de prepolarizare, rezultatul funcției se deplasează pe axa absciselor:

$$y = f(t) = \sum_{i=1}^n x_i * w_i + \theta \quad (1.1)$$

Rezultatul obținut în urma însumării trece mai apoi printr-o funcție de activare, a cărei ieșire va reprezenta ieșirea neuronului. Funcția de activare, în cazul neuronului descris de Rosenblatt, este reprezentată de o funcție treaptă al cărei prag de activare este dat chiar de termenul de prepolarizare. Alegerea unei funcții treaptă s-a făcut ca urmare a faptului că perceptronul simplu a fost construit pentru a realiza doar clasificări binare. Se obține astfel ieșirea neuronului:

$$out(t) = \begin{cases} 0, & y < 0 \\ 1, & y \geq 0 \end{cases} \quad (1.2)$$

Deoarece se bazează pe metodologia învățării supervizate, ieșirea unității funcționale este mai apoi comparată cu ieșirea dorită. Rezultatul obținut în urma comparației este utilizat pentru ajustarea ponderilor rețelei, astfel încât rezultatele sistemului să se îmbunătățească. Acest proces se repetă până când datele de intrare sunt clasificate corect, cu o eroare rezonabilă.

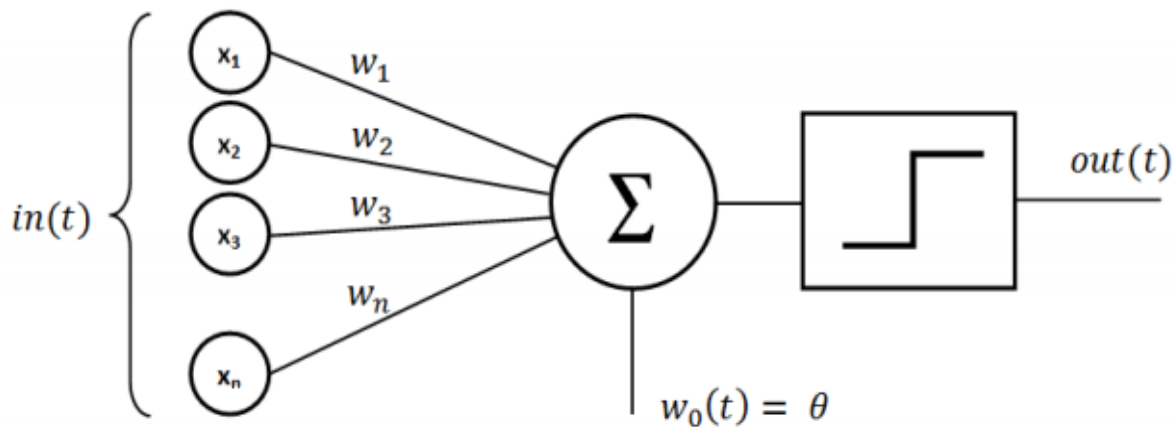


Figura 1.3: Topologia perceptronului [12]

Rezultatele obținute cu ajutorul perceptronului simplu au determinat apariția unei noi structuri de tip ANN. Astfel, în anul 1960, Bernard Widrow și Ted Hoff propun arhitectura ADALINE (i.e. neuron cu adaptabilitate liniară). Astfel, funcția treaptă folosită de Rosenblatt este transformată într-o funcție de gradul I, algoritmul de învățare fiind acum bazat pe minimizarea unei funcții de eroare. Cu toate acestea, arhitectura nou dezvoltată prezintă o limitare demonstrată în 1969 de Minski și Papert [11]. Aceștia arată faptul că ADALINE nu poate generaliza în orice context, în particular nu poate clasifica date de intrare care nu aparțin unor clase liniar separabile. Acest rezultat conduce la o stagnare a dezvoltării ANN, stagnare care se menține până la mijlocul anilor '80.

Unul dintre factorii care a condus la revenirea interesului pentru domeniul ANN este o consecință a activităților desfășurate în perioada de stagnare. Astfel, principalele progrese în respectiva perioadă s-au concretizat exclusiv în aria procesării simbolice. Evoluția în această arie era, însă, lentă și limitată la simulări. Un alt aspect îl reprezintă avansul tehnologic și puterea computațională crescută a *hardware*-ului disponibil la acea vreme. Acest factor continuă să fie unul important chiar și în zilele noastre, date fiind cerințele computaționale pe care antrenările ANN-urilor le au, antrenări imposibil de realizat în acele timpuri. Un al treilea aspect care a condus la această revenire pornește chiar de la sugestia lui Minski și Papert, care arătau că utilizarea unei rețele neuronale artificiale cu mai multe straturi ar putea rezolva problema claselor care nu erau liniar separabile. Astfel, atenția cercetătorilor s-a îndreptat înspre posibilitatea realizării unor rețele neuronale artificiale multistrat, cu straturi interconectate. Se obțin astfel rezultate notabile, un exemplu concludent în acest sens fiind reprezentat de LeCun et. al [13] în 1989, care implementează un sistem de recunoaștere codurilor poștale scrise de mână folosind ANN-uri multistrat. Acesta este, de altfel, unul dintre primele lucrări ce pun în comun vastul domeniu al instruirii asistate de calculator și aria recunoașterii imaginilor.

În anii '90 încep să fie utilizate clasificatoarele mașini cu vector suport (SVM). Folosite în numeroase contexte, precum recunoașterea scrisului sau a vorbirii, au avantajul simplității comparativ

cu ANN și, în același timp, obțin rezultate bune. Dezavantajul lor este că, în timp ce implementarea este simplă pentru clasificarea a două clase, algoritmi pentru discriminarea între multiple clase sunt dificil de implementat. Dată fiind creșterea puterii computaționale chiar și a calculatoarelor de uz general, cât și numărul tot mai mare de informații ce se cer a fi procesate, cercetătorii și-au îndreptat atenția spre un nou subdomeniu al ML, subdomeniu care a luat amploare în ultimii ani: rețelele neuronale adânci (eng. Deep Learning).

1.2.4 Rețele neuronale adânci

Așa cum reiese și din denumirea pe care o au, rețelele neuronale adânci vin ca o completare a rețelilor neuronale artificiale, un caz particular a acestora, cu diferența că prezintă două sau mai multe straturi ascunse neliniare. În acest context, neliniaritatea face referire la tipul funcției de activare pe care o folosesc unitățile funcționale pentru calculul ieșirii. Folosite în numeroase domenii, ca de pildă *computer vision* sau recunoașterea automată a vorbirii, ele au avantajul că pot determina în mod independent trăsăturile intrării ce trebuie clasificată. Astfel, pe baza exemplelor oferite la intrare, ele pot fi antrenate, fără a mai fi nevoie de o prelucrare manuală a datelor de intrare în vederea obținerii trăsăturilor. Un exemplu concludent în acest sens îl reprezintă analiza și recunoașterea imaginilor folosind ANN. Prin intermediul procesării pixelilor care constituie imaginea oferită ca intrare a sistemului, algoritmul stabilește ce combinații ale pixelilor reprezintă trăsături specifice unei imagini, trăsături ce se regăsesc de multiple dați în cadrul respectivului cadru. Pe măsură ce acesta parcurge straturile din care rețeaua este alcătuită, aceste trăsături își schimbă reprezentarea, devenind reprezentări cât mai fidele și concise ale imaginii inițiale.

Popularitatea domeniului DL a crescut odată cu rezultatele spectaculoase obținute în varii domenii, rezultate mai bune decât stadiul actual al tehnologiei de la acel moment. Un exemplu concludent în acest sens îl reprezintă lucrarea publicată de Krizhevski et al. [14] în 2012, lucrare ce are ca scop clasificarea imaginilor pe baza setului de date ImageNet. Rezultatele acestui grup de cercetători se bazează pe utilizarea unei noi tehnici de tip DL, cunoscută drept rețele neuronale convoluționale (eng. Convolutional Neural Networks – CNN). Cu ajutorul sistemului astfel construit, ei au obținut o eroare de 15.3% pentru primele 5 predicții, în condițiile în care o altă echipă a obținut, pe același set de date, o eroare de 26.2%. Aceste rezultate ne-au determinat să apelăm la CNN în cadrul prezentei lucrări.

Capabilitățile DL sunt strâns legate de două aspecte pe care le vom discuta în acest subcapitol. În primul rând, trebuie analizată funcția de activare folosită de unitățile funcționale ce alcătuiesc rețeaua. În cadrul subcapitolului 1.2.3, am analizat perceptronul simplu, care folosea ca funcție de activare o funcție treaptă. Din cauza faptului că această funcție ia doar două valori, o analiză bună a erorii este imposibil de implementat. Astfel, ieșirea unui neuron va fi tratată ca fiind fie corectă, fie incorectă, fără a se analiza gradul de corectitudine în vederea ajustării ponderilor. Așa cum am menționat, un prim pas în acest sens a fost făcut de Widrow și Hoff, care au utilizat o funcție liniară în construcția perceptronului. Cu toate acestea, rezultatele au arătat că nici o astfel de implementare nu este mulțumitoare pentru realizarea unei clasificări eficiente.

În vederea rezolvării acestui impediment, s-a luat decizia utilizării unor funcții sigmoidale. Avantajele acestui tip de funcții se văd în special în cadrul rețelelor cu un set redus de straturi. Bazate pe algoritmul gradientului negativ și, implicit, pe minimizarea erorii pătratice, rețelele ce implementează astfel de funcții au anumite dezavantaje, precum viteza mică de convergență, apariția unor oscilații ale erorii sau stagnarea procesului de învățare, cauzată de întâlnirea unui punct de minim local. Deși aceste probleme pot fi rezolvate, utilizarea funcțiilor sigmoidale nu este însă eficientă în situații în care avem rețele de dimensiuni mari [14]. În principal, acest lucru este cauzat de faptul că funcțiile sigmoidale au în compoziție funcții exponențiale, ale căror rezultate pot fi adesea prea mari și, deci, dificil de gestionat. Un alt aspect negativ al acestor valori mari obținute îl reprezintă valoarea gradientului negativ. Astfel, pe măsură ce argumentul funcției crește, valoarea gradientului negativ scade, fapt ce conduce la încetinirea procesului de învățare.

Pentru a depăși această problemă, s-a luat decizia utilizării unui alt tip de funcții, cunoscută ca funcția semi-liniară (eng. Rectified Linear Unit – ReLU) și definită ca fiind:

$$f(x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases} \quad (1.3)$$

Spre deosebire de funcțiile sigmoidale, problema încetinerii învățării din cauza valorii gradientului negativ dispare odată cu utilizarea ReLU. Acest lucru se întâmplă deoarece, în această situație și pentru argumente pozitive, valoarea gradientului va fi mereu constantă, dată de panta funcției, așa cum se poate observa în Figura 1.4. Utilitatea acestei proprietăți este demonstrată de Krihevski et al., care arată că numărul de iterații necesare pentru obținerea convergenței în cazul utilizării gradientului negativ stochastic a scăzut de 6 ori, comparativ cu utilizarea funcțiilor sigmoidale.

Un alt avantaj al folosirii funcției semi-liniare îl reprezintă distribuția rezultatelor funcției. Astfel, pentru un argument negativ al funcției, ReLU produce o ieșire nulă, după cum se poate observa în Figura 1.4.

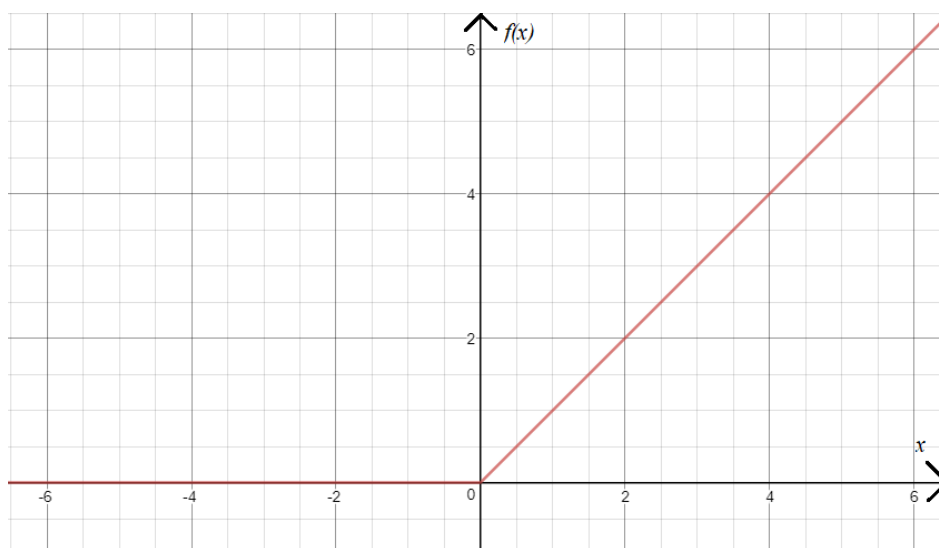


Figura 1.4: Funcția semi-liniară (ReLU)

Construind o rețea cu mai mulți neuroni ce folosesc ReLU, ieșirile rețelei vor avea un grad mare de împrăștiere, ceea ce conduce la o clasificare mai bună. Acest lucru se află în opoziție cu utilizarea funcției sigmoidale, ale cărei rezultate sunt dense și, implicit, dificil de interpretat. Această proprietate vine însă și cu un dezavantaj. În cazul în care argumentele funcției sunt distribuite în jurul lui 0 sau a unei valori negative, se ajunge într-un punct în care neuronii nu mai sunt excitați de nicio dată de intrare pe parcursul antrenării.

Cel de-al doilea concept care a influențat puternic dezvoltarea domeniului rețelelor neuronale adânci a fost avansul tehnologiilor *hardware* și, în particular, al unităților de procesare grafică (eng. Graphic Processing Unit – GPU). Principalul avantaj al acestor echipamente este dat de faptul că îmbunătățesc foarte mult viteza de antrenare a unei ANN, comparativ cu unitățile de procesare centrală (eng. Central Processing Unit – CPU). Rezultatele privitoare la timpul de antrenare pe diferite topologii de rețele neuronale adânci arată că viteza poate crește de până la 10 ori odată cu utilizarea GPU în locul CPU. Acest fapt se bazează în principal pe modul în care unitățile de procesare grafică execută instrucțiunile. CPU sunt, prin definiție, secvențiale și controlează strict fluxul datelor ce trebuie prelucrate. Acest lucru se traduce prin capacitatea de a executa instrucțiuni complexe, având însă o putere computațională redusă, dat fiind numărul de asemenea redus de nuclee de procesare.

În comparație cu ele, GPU prezintă un înalt grad de paralelism. Realizate inițial pentru crearea și prelucrarea imaginilor în vederea afișării lor pe ecran, ele au început să deservească domeniul DL odată cu experimentele realizate cu diferite topologii de rețele. GPU sunt utilizate pentru prelucrări masive de date, fiind capabile să realizeze schimburi rapide între firele de execuție aflate în desfășurare. Deși au un set mai redus de instrucțiuni decât CPU, unitățile de procesare grafică sunt mai potrivite pentru aplicații intensiv computaționale, deoarece numărul mare de nuclee de procesare le permite să execute în paralel instrucțiuni simple pentru fiecare element. Astfel, nu este nevoie de un algoritm complex de control al fluxului execuției și nici de memorii *cache* masive, întârzierile cauzate de accesul la memorie putând fi mascate prin realizarea altor operații matematice.

Dat fiind faptul că scopul acestui proiect îl reprezintă implementarea unui sistem automat de labiolectură și deci, implicit, prelucrarea unor imagini, este firească utilizarea unor circuite de tip GPU, grație capacității acestora de a procesa vectori sau matrice.

1.2.5 Rețele neuronale convoluționale

Parte a subdomeniului mai vast reprezentat de rețelele neuronale artificiale, cele convoluționale au ca inspirație o zonă specifică a creierului uman, și anume cortexul vizual primar, localizat în lobul occipital. Considerat un „detector de reprezentări” [15], asocierea cu ANN este imediată, dat fiind faptul că acestea au ca vectori de intrare reprezentările, sau trăsăturile, datelor ce trebuie clasificate. Analiza structurală ne arată ca există două tipuri de celule în cortexul vizual: celule simple și celule complexe. Primele dintre acestea „răspund la bare de lumină sau întuneric proiectate pe retină sub un anumit unghi” [15]. Ele funcționează, așadar, ca un detector de muchii, observând linia de separație între obiecte. Celulele complexe, pe de altă parte, acoperă un spectru mai larg de obiecte observabile, nefiind dependente de zona de pe retină pe care lumina cade. Puse împreună,

aceste două categorii de celule au capacitatea de a edifica harta captată de câmpul vizual, ținând seama de corelația dintre obiectele observate și de formele acestora.

Încercările de a emula structura cortexului vizual primar au început în 1980, odată cu realizările lui Kunihiko Fukushima. Acesta implementează „neocognitronul”, o un model de rețea neuronală ale cărei straturi au ca inspirație celulele simple și pe cele complexe menționate anterior. Stratul de intrare al modelului propus este urmat de un strat similar celulelor simple, strat care realizează detecția contrastului. Urmează apoi straturi succesive similare celulelor complexe, cu un al doilea strat care implementează detecția muchiilor. Restul straturilor până la ieșire simulează proprietatea de invarianță locală și, în plus, fiecare celulă complexă urmărește un singur model. Legăturile sunt unu-la-unu, ceea ce face ca respectivul model să fie urmărit de o singură grupare de celule pe întreaga lungime a rețelei parcurse. Se obțin astfel rezultate bune comparativ cu structurile propuse anterior, existând însă și un dezavantaj. Procesul de învățare este unul extrem de simplist, deoarece, la acea vreme, nu exista o metodă care să permită ajustarea ponderilor în funcție de o eroare globală, măsurată pe întreaga rețea neuronală.

Rezolvarea acestei probleme vine odată cu lucrarea lui Rumelhart et al. [11], lucrare care propune o implementare a algoritmului *BackPropagation*. Descoperirea acestui algoritm a fost făcută de Paul Werbos [11] în 1975, el oferind o soluție care admitea utilizarea unei rețele neuronale artificiale cu mai multe straturi, devenind astfel o rețea neuronală adâncă. S-a demonstrat pe baza acestei descoperiri faptul că o rețea neuronală cu un singur strat ascuns și cu număr suficient de mare, dar finit, de neuroni poate aproxima cu o precizie arbitrară orice funcție matematică. Conceptul ce stă la baza algoritmului *BackPropagation* este într-o oarecare măsură similar principiului de funcționare a perceptronului simplu. Astfel, se calculează gradientul erorii în funcție de ponderile fiecărei unități funcționale. Odată cu modificarea ponderilor, se modifică implicit și valoarea gradientului. Acest proces iterativ se repetă în vederea identificării acelor ponderi pentru care gradientul erorii scade, scăzând astfel eroarea de clasificare a rețelei. Folosirea unor algoritmi de optimizare în relație cu acest procedeu conduce la îmbunătățirea rezultatelor tehnicii, exemple concrete în acest sens fiind reprezentate de algoritmul gradientului negativ (eng. Gradient Descent – GD) sau cel pe baza gradientului negativ stohastic (eng. Stochastic Gradient Descent – SGD). În timp ce primul ține cont de eroarea de pe întregul lot de antrenare, cel din urmă ia în considerare eroarea după fiecare iterație a unui bloc a datelor de intrare, dimensiunea blocului fiind definită de utilizator.

Primele rezultate obținute prin intermediul algoritmului *BackPropagation* au fost cele ale lui Yann LeCun [13], în cadrul proiectului ce avea ca scop recunoașterea scrisului de mână. La momentul respectiv, acest proiect a reprezentat cea mai de succes utilizare a unor CNN, dată fiind cantitatea mare de date de intrare necesare pentru antrenarea rețelei, astfel încât aceasta să ofere o acuratețe bună în clasificare. CNN continuă să fie utilizate și în prezent, astfel că este necesară o analiză a componentelor ce definesc o astfel de rețea neuronală.

1.2.6 Proprietățile rețelelor neuronale convoluționale

Un prim aspect ce trebuie analizat este operația de convoluție pe care se bazează CNN în vederea extragerii trăsăturilor. Din punct de vedere matematic, operația de convoluție este definită ca reprezentând combinația a două semnale în vederea obținerii unui al treilea. Comportamentul ei este specific unui filtru, putând fi utilizată la detecția muchiilor ori la creșterea sau scăderea contrastului unei imagini. Aplicând imaginii un filtru convoluțional, se obține o nouă imagine ce conține schema trăsăturilor. Numărul de transformări rezultate reprezintă intrările rețelei neuronale, capabile astfel să învețe trăsăturile datelor de intrare. Tot în contextul operației de convoluție trebuie menționată prezența sau absența unui tip de bordare. Astfel, în anumite situații se preferă renunțarea la pixelii de pe marginea imaginii, deoarece pentru aceștia nu se poate realiza o procesare pe vecinătăți. În alte contexte, este preferabilă bordarea imaginii cu valori nule. Ambele variante au ca scop reducerea trăsăturilor pe care rețeaua urmează să le învețe.

O a doua proprietate importantă a CNN este reprezentată de distribuția ponderilor (eng. *weight sharing*). Această trăsătură permite rețelelor neuronale convoluționale să rețină informația utilă și, implicit, să învețe pe baza ei, indiferent de poziția acesteia în cadrul datei de intrare. Distribuția ponderilor permite utilizarea unei proprietăți importante a datelor multidimensionale, și anume invarianța spațială a trăsăturilor. Aceste trăsături, obținute cu ajutorul operației de convoluție menționate anterior, sunt organizate într-un plan, iar fiecare plan, obținut prin intermediul unui nucleu (eng. *kernel*) diferit, conține informații cu privire la o anumită trăsătură. Astfel, se pot reține toate proprietățile dorite din datele inițiale, indiferent de poziția acestora, grație schemelor trăsăturilor. Procedul de extragere a lor și de organizare în scheme este cunoscut ca o operație convoluțională, de unde reiese și denumirea acestui tip de rețele.

Un alt parametru al rețelelor neuronale convoluționale este reprezentat de dimensiunea kernelului, denumit și *local receptive field*. El reprezintă interfața de legătură dintre fiecare unitate funcțională și datele multidimensionale. Principiul care stă la baza acestui parametru este dat tot de structura cortexului vizual [15]. Astfel, se realizează o conectivitate locală (eng. *locally-connected areas*), ținând seama de lățimea și înălțimea datelor de intrare, cea de-a treia dimensiune folosindu-se de o conectivitate completă (eng. *fully-connected*). Astfel, pe două dimensiuni datele de intrare sunt reduse în funcție de dimensiunea kernelului. Cea de-a treia dimensiune rămâne nealterată și ieșirea la finalul rețelei are aceeași adâncime cu data de intrare, dată de numărul de filtre utilizate. Această tehnică permite extragerea unor trăsături cu un nivel superior de abstractizare și, în plus, se reduce riscul de supraantrenare.

Un alt aspect al CNN care conduce la reducerea dimensiunii datelor de intrare este reprezentată de tehnica subeșantionării spațiale (eng. *pooling*). Realizarea acestei operații se face prin reducerea valorilor trăsăturilor din schema obținută în urma convoluției la o singură valoare, considerată cea mai semnificativă. Această valoare rezultată poate fi fie media valorilor din aria dată, fie valoarea maximă. Ca și în cazul operației de convoluție, și aici se poate realiza bordarea intrării, depinzând de rezultatul dorit. Avantajele subeșantionării spațiale sunt reducerea schemei trăsăturilor,

dar și faptul că împiedică rețeaua să învețe poziția unor trăsături, oferindu-i astfel capacitatea de a realiza o învățare invariantă spațial.

O ultimă proprietate ce merită avută în vedere o reprezintă rata de regularizare (eng. dropout rate). Această abordare presupune dezactivarea aleatoare a unor unități funcționale pe parcursul antrenării, fapt ce simulează antrenarea mai multor rețele, diferite între ele, dar conectate prin intermediul unor ponderi comune. Regularizarea se aplică doar în procesul propagării înainte a datelor, nu și în cazul algoritmului *BackPropagation*. Prin excluderea anumitor neuroni, valoarea intrărilor lor scade, fiind necesară o creștere a ponderilor care să compenseze acea scădere. În final, este necesară o ultimă ajustare a ponderilor, relativă la rata de regularizare aplicată. Se permite astfel tuturor neuronilor să învețe independent anumite trăsături, conform unei structuri ideale, în caz contrar ajungându-se la fenomenul de coadaptare, în care mai mulți neuroni rețin aceeași trăsătură. În plus, procedeul regularizării previne fenomenul de supraantrenare, prin obținerea unor rezultate cu grad mai mare de generalitate.

Rețelele neuronale convoluționale reprezintă, așadar, unelte foarte puternice pentru extragerea trăsăturilor dintr-un set de date de antrenare. Trebuie însă amintit că, pentru realizarea unei clasificări, este necesară definirea unor rețele interconectate (i.e. un strat *fully-connected*), care să poată propaga trăsăturile determinate în scopul învățării. Se obține astfel o schemă similară cu cea prezentată în Figura 1.5. Astfel, se realizează o legătură între stratul de intrare și fiecare strat convoluțional, capabil să reunoască o anumită trăsătură. Următoarea etapă este reprezentată de procesul de *pooling*, aplicat fiecărei hărți a trăsăturilor, în vederea reducerii datelor ce urmează să se propage în rețea. O ultimă etapă de procesare este reprezentată de stratul *fully-connected*, care procesează datele în adâncime, urmând apoi a fi propagate rezultatele spre ieșirea sistemului. Avantajul acestui tip de rețele îl reprezintă posibilitatea utilizării algoritmului *BackPropagation* pentru ajustarea erorii folosind SGD, ajustând astfel și ponderile rețelei neuronale convoluționale. Se obține astfel un sistem de clasificare și predicție foarte puternic, capabil să realizeze procesări multiple în vederea determinării apartenenței unei date la o anumită clasă

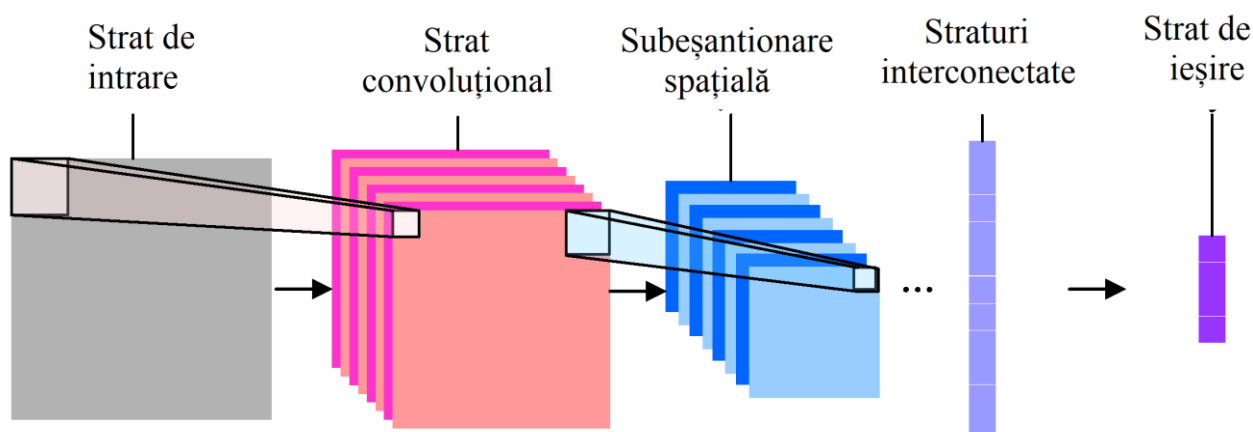


Figura 1.5: Schema principală a unei rețele neuronale convoluționale [17]

1.3 Biblioteci specifice

Cunoașterea uneltelor *software* puse la dispoziție și a bibliotecilor disponibile este o etapă importantă în realizarea unui sistem robust și cu o acuratețe ridicată. Se cere astfel analiza teoretică a proprietăților mediilor de lucru ce urmează a fi folosite, în vederea folosirii acestora conform cu specificațiile și potențialul lor. Vom realiza, așadar, o analiză succintă a principalelor biblioteci utilizate în prezenta lucrare: TensorFlow, Keras și OpenCV.

1.3.1 TensorFlow

Tensorflow (TF) [18] reprezintă o librărie *open source* realizată în vederea realizării unor sisteme de tip ML, librărie scrisă în C++ și Python. Lansată în noiembrie 2015, ea a fost dezvoltată de Google Brain Team folosită inițial de aceștia cu scopul de a îmbunătăți anumite produse comercializate de Google, precum recunoașterea automată a vorbirii în serviciul Google Now. Unul dintre principalele avantaje ale TF îl reprezintă ușurința cu care poate fi folosit atât pentru prototipare, cât și pentru producție, nefiind necesare compilări suplimentare sau modificări ale codului sursă. În plus, faptul că echipa Google îl folosește în continuare atât pentru producție, cât și pentru cercetare, îi demonstrează flexibilitatea. Un alt avantaj al acestei biblioteci este reprezentat de faptul că nu este necesară utilizarea unui alt *Application Programming Interface* (API) pentru a face transferul de la rularea proiectului pe o arhitectură CPU la o rulare pe GPU.

O parte cheie a librăriei TensorFlow este modul în care este reprezentat fluxul datelor. Descrierea operațiilor matematice prin intermediul nodurilor, pentru operațiile matematice în sine, și a dreptelor, pentru reprezentarea legăturilor dintre noduri, sugerează modul în care informația parcurge rețeaua ca un vector multidimensional, un tensor. În plus, nodurilor li se alocă memorie în dispozitivul *hardware*, operațiile pe care aceștia le reprezintă fiind executate în paralel după alocarea tuturor resurselor.

Principalul argument pentru alegerea librăriei TensorFlow în detrimentul unor variante similare, precum Caffe sau Theano, îl reprezintă comunitatea *online* foarte activă. Acest lucru se traduce prin faptul că orice problemă este identificată în scurt timp de la apariție și, mai apoi, rezolvată. Este deci un *framework* stabil și robust. În plus, date fiind resursele de care Google dispune pentru a continua dezvoltarea librăriei, cât și disponibilitatea acestora de a oferi o documentație bine structurată a API-ului Python, TensorFlow reprezintă o alegere potrivită pentru majoritatea proiectelor realizate în domeniul Deep Learning.

Un alt argument în favoarea utilizării TensorFlow ca *framework* pentru prezenta lucrare este reprezentat de decizia celor de la Google de a utiliza TF în propriile proiecte, ceea ce va asigura dezvoltarea continuă a sa. De asemenea, faptul că o bună parte dintre dezvoltatorii acestei librării au luat parte la realizarea Theano sau Torch7 arată experiența echipei ce stă în spatele TF, experiență cu influență directă în calitatea produsului final. O consecință a acestui aspect o reprezintă performanțele ridicate, ușurința cu care codul sursă poate fi înțeles, cât și flexibilitatea *framework*-ului, indiferent

de platformă. Analizând argumentele favorabile librăriei TensorFlow, am luat decizia utilizării ei în cadrul proiectului curent.

1.3.2 Keras

Keras [19] este un API de nivel înalt, creat pentru implementarea rețelelor neuronale. Scris în Python, el este compatibil cu multiple *framework*-uri dedicate domeniului ML, printre care și TensorFlow. Ușor de folosit, pentru implementarea modelelor rețelelor prin intermediul acestui API nu sunt necesare fișiere suplimentare de configurare. Un alt avantaj îl reprezintă gradul mare de modularitate, modelele ce se doresc a fi implementate fiind simplu de conectat, atât din perspectiva arhitecturii, cât și din punctul de vedere al proprietăților respectivelor modele. Un alt avantaj important, alături de documentația pusă la dispoziție utilizatorilor, îl reprezintă posibilitatea de a implementa rețele neuronale convoluționale, pe de o parte, și rețele neuronale recursive, pe de altă parte, cât și diferite structuri ce le combină pe cele două.

Aflat în strânsă legătură cu TensorFlow, reacțiile rapide ale comunității la problemele care apar în dezvoltarea versiunilor Keras conduce la rezolvarea în timp util a acestor probleme. Principalul avantaj îl reprezintă simplitatea cu care se pot implementa modelele dorite, fiind un API de nivel înalt, spre deosebire de TF, care funcționează ca un *backend*. Această îmbinare dintre cele două a condus la faptul că, la momentul actual, TensorFlow și Keras au cele mai multe citări [19] în lucrările scrise în domeniul DL. La fel ca TF, Keras permite rularea sistemelor dezvoltate atât pe unități de procesare centrale, cât și pe unități de procesare grafice, fără a fi nevoie de modificarea codului sursă. Date fiind avantajele pe care utilizarea API-ului Keras le prezintă, am luat decizia firească de a îl utiliza în cadrul acestei lucrări pentru implementarea rețelelor neuronale artificiale dorite.

1.3.3 OpenCV

OpenCV [20], acronim pentru *Open Source Computer Vision*, reprezintă o bibliotecă construită pentru a realiza operații de *Computer Vision* (CV), altfel spus vedere computațională. Operațiile pe care această bibliotecă le poate realiza sunt multiple, astfel că dintre ele amintim doar recunoașterea gesturilor sau a fețelor, identificarea obiectelor sau a mișcării ori chiar implementări ale realității augmentate. În plus, avansul domeniului rețelelor neuronale adânci (eng. Deep Neural Networks – DNN) a făcut ca introducerea unei librării specializate pentru ML să devină o necesitate, apărând astfel capabilități noi, precum implementarea ANN-urilor sau a unor algoritmi tipici pentru calculul erorilor.

Apărută în anul 2000 și dezvoltată la acel moment de Intel Research, biblioteca OpenCV este scrisă în limbajul C++, însă prezintă numeroase API-uri care permit dezvoltarea proiectelor în Python, Matlab sau Java. Principalul avantaj al acestei biblioteci, pe lângă numeroasele funcționalități pe care le implementează, este dat de faptul că poate fi implementată pe numeroase sisteme de operare, atât mobile, cât și de tip *desktop*. În plus, dată fiind dezvoltarea sistemelor GPU, s-a introdus

posibilitatea accelerării aplicațiilor dezvoltate cu funcții ale OpenCV prin intermediul unei interfețe pentru GPU-urile construite după specificul *Compute Unified Device Architecture* (CUDA). Aceste argumente în favoarea bibliotecii OpenCV au condus la utilizarea acesteia în prezenta lucrare.

1.4 Rezultate anterioare

Cercetările în domeniul labiolecturii au o lungă istorie la nivel internațional, drumul implementărilor unor astfel de sisteme fiind deschis de Petajan [3] în 1984. El a folosit informația vizuală alături de cea acustică, realizând astfel un sistem dual. Intrarea algoritmului bazat pe mișcarea buzelor era reprezentată de modelul dat de cele mai bune două rezultate ale modelului acustic. Această construcție a adus o îmbunătățire sistemelor de recunoaștere automată a vorbirii, dar a și deschis calea implementărilor algoritmilor bazați exclusiv pe mișcarea buzelor.

Așa cum arată Zheng et. al [21], prima utilizare a rețelelor neuronale artificiale pentru realizarea labiolecturii aparține lui Yuhas, care a realizat extragerea trăsăturilor pe baza pixelilor în vederea identificării vocalelor. Conform [21], în 1992 s-au utilizat primele rețele neuronale care țineau cont de întârzierea în timp, iar în 1994 apare primul sistem capabil să identifice litere și cuvinte izolate, neintegrate într-o propoziție. În anii ce urmează apar numeroase modificări și optimizări în cadrul procesului de extragere a trăsăturilor sau de identificare a conturului buzelor și a modului în care acestea se mișcă, cât și optimizări în vederea reducerii trăsăturilor necesare pentru antrenarea sistemului. Așa cum arată Zheng [21], în 2002 apar primele sisteme utilizate în producție, capabile să transforme mișcarea buzelor vorbitorilor în sunete, dar și să realizeze procesul invers. Apar apoi lucrări care prezintă sisteme ce pot distinge limba utilizată de vorbitor, cât și aplicații ce permit urmărirea buzelor pe măsură ce acestea se mișcă, grație unui senzor de adâncime.

Principala problemă în acest domeniu o reprezintă absența unor seturi de date publice potrivite pentru astfel de aplicații. Datele disponibile în acest moment sunt insuficiente pentru realizarea unui model robust, scalabil, capabil să facă generalizări în afara condițiilor unui mediu controlat. Vom realiza, așadar, o descriere a seturilor de date existente, împreună cu cele mai bune rezultate obținute pe fiecare set, conform [22], prezentată în Tabelul 1.1.

Din punctul de vedere al lucrării curente, cele mai interesante rezultate sunt reprezentate de cele ale lui Assael et al. [22] și ale lui Chung & Zisserman [21]. Primii au implementat în 2016 un sistem automat de labiolectură antrenat la nivel de propoziție, cu o acuratețe de 95.2% pe GRID. Acesta este, de altfel, primul astfel de sistem antrenat *end-to-end*, ceea ce înseamnă că nu se realizează o extragere a trăsăturilor preliminară antrenării, ci acest proces este integrat în cel de antrenare. Principial, lucrarea lor se bazează pe o rețea convoluțională, urmată de una recursivă în vederea extragerii trăsăturilor temporale, ieșirea fiind precedată de o interconectare a rețelei precedente pentru analiza erorii.

Tabel 1.1: Seturi de date disponibile pentru antrenarea sistemelor automate de labiolectură

<i>Nume</i>	<i>Mediul înregistrării</i>	<i>Ieșire</i>	<i>Tipul înregistrării</i>	<i>Numărul claselor</i>	<i>Numărul subiecților</i>	<i>Cel mai bun rezultat</i>
AVICAR	Mașină	Cifre	Înregistrare continuă	10	100	37.9 %
AVLetter	Laborator	Alfabet	Cuvinte izolate	26	10	64.6 %
CUAVE	Laborator	Cifre	Cuvinte izolate	10	36	83%
GRID	Laborator	Cuvinte	Înregistrare continuă	8.5	34	95.2%
OuluVS1	Laborator	Fraze	Cuvinte izolate	10	20	91.4%
OuluVS2	Laborator	Fraze	Cuvinte izolate	10	52	94.1%
BBC TV	TV	Cuvinte	Înregistrare continuă	333/500	>1000	65.4%

Lucrarea lui Chung & Zisserman [21] prezintă un sistem antrenat la nivel de cuvânt, compus, de asemenea, dintr-un set de rețele neuronale convoluționale. La fel ca [21], sistemul este capabil de a recunoaște cuvinte dintr-o înregistrare continuă, astfel încât să se adapteze situațiilor din viața de zi cu zi, acolo unde pot exista modificări față de cuvintele segmentate, date de pronunția cuvintelor învecinate. Sistemul astfel dezvoltat a obținut o acuratețe de 91.4% pe OuluVS1, 94.1% pe OuluVS2 și 65.4% pe BBC TV.

Cercetările noastre au arătat că, până la acest moment, nu există niciun sistem construit pentru a realiza procesul de labiolectură în limba română. Acest aspect și utilitatea unui astfel de sistem, cât și rezultatele anterioare obținute la nivel internațional, sunt motivele care ne-au determinat să propunem în prezenta lucrare un sistem automat de labiolectură capabil să identifice cuvinte cheie în limba română.

Capitolul 2

Arhitectura sistemului propus

În acest capitol vom prezenta structura propusă pentru realizarea unui sistem automat de labiolectură în limba română. Date fiind concluziile deduse în subcapitolul 1.1.3, cât și sistemele realizate de [22] și [23], luăm următoarele decizii:

1. Sistemul va fi antrenat la nivel de cuvânt.

În cazul sistemelor de recunoaștere a vorbirii, atât pe baza informației acustice, cât și utilizând informație vizuală, se pot realiza trei tipuri de antrenare: la nivel de fonem/*viseme*, la nivel de cuvânt, respectiv la nivel de propoziție. Există două motive pentru care excludem prima variantă. În primul rând, așa cum am arătat în secțiunea 1.1.2, numeroase foneme nu pot fi identificate exclusiv pe baza labiolemelor, pronunția lor fiind obținută fără participarea buzelor. În al doilea rând, o analiză exhaustivă a *visemelor* limbii române nu a fost realizată încă, ceea ce conduce la imposibilitatea realizării unui sistem de labiolectură pe baza labiolemelor în absența unei baze teoretice.

Decizia de a nu realiza antrenarea la nivel de propoziție am luat-o din perspectiva raportului complexitate-rezultate. Astfel, am arătat în cadrul secțiunii 1.1.2 că limba română, fiind o limbă fonemică, prezintă un număr redus de omofone. Astfel, sunt rare situațiile în care este necesară cunoașterea contextului în vederea identificării unui anumit cuvânt. De asemenea, dat fiind faptul că scopul lucrării îl reprezintă identificarea unui set specific de cuvinte, cuvinte ale căror labioleme și reprezentări fonemice nu sunt identice, cunoașterea contextului nu este necesară. Apelul la contextul în care un anumit cuvânt a fost utilizat va fi făcut în vederea

îmbunătățirii rezultatelor rețelei, fără însă ca acest aspect să fie parte integrantă a procesului de antrenare a rețelei neuronale utilizate.

2. Sistemul va fi capabil să identifice cuvinte din înregistrări continue.

În cazul modului în care cuvintele pot fi folosite pentru antrenarea rețelei, există două posibilități: cuvinte izolate și înregistrări continue. Prima variantă conduce la rezultate mai bune în condiții de laborator, dată fiind segmentarea exactă a cuvintelor. Pe de altă parte, în condiții reale este improbabilă existența unor cuvinte pronunțate independent, astfel că un sistem practic trebuie să fie capabil să identifice cuvintele spuse de vorbitor în cadrul unor fraze complete. Vom folosi, așadar, înregistrări complete, din care sistemul va extrage fiecare cuvânt, apoi îl va identifica.

3. Sistemul va avea o arhitectură independentă de limba folosită.

Reducând problema labiolecturii la o problemă de recunoaștere a mișcării, considerăm realizabilă implementarea unui sistem care să nu aibă o dependență directă de limba folosită de vorbitori. Astfel, dată fiind folosirea unei antrenări la nivel de cuvânt, setul de cuvinte folosit va determina limba la care sistemul este capabil să răspundă. Pentru îmbunătățirea rezultatelor, vom adăuga un bloc suplimentar ce conține un set de reguli lingvistice pentru limba română, fără ca acest bloc să definească rezultatele brute ale sistemului.

Propunem, așadar, în Figura 2.1 schema bloc a sistemului de labiolectură:

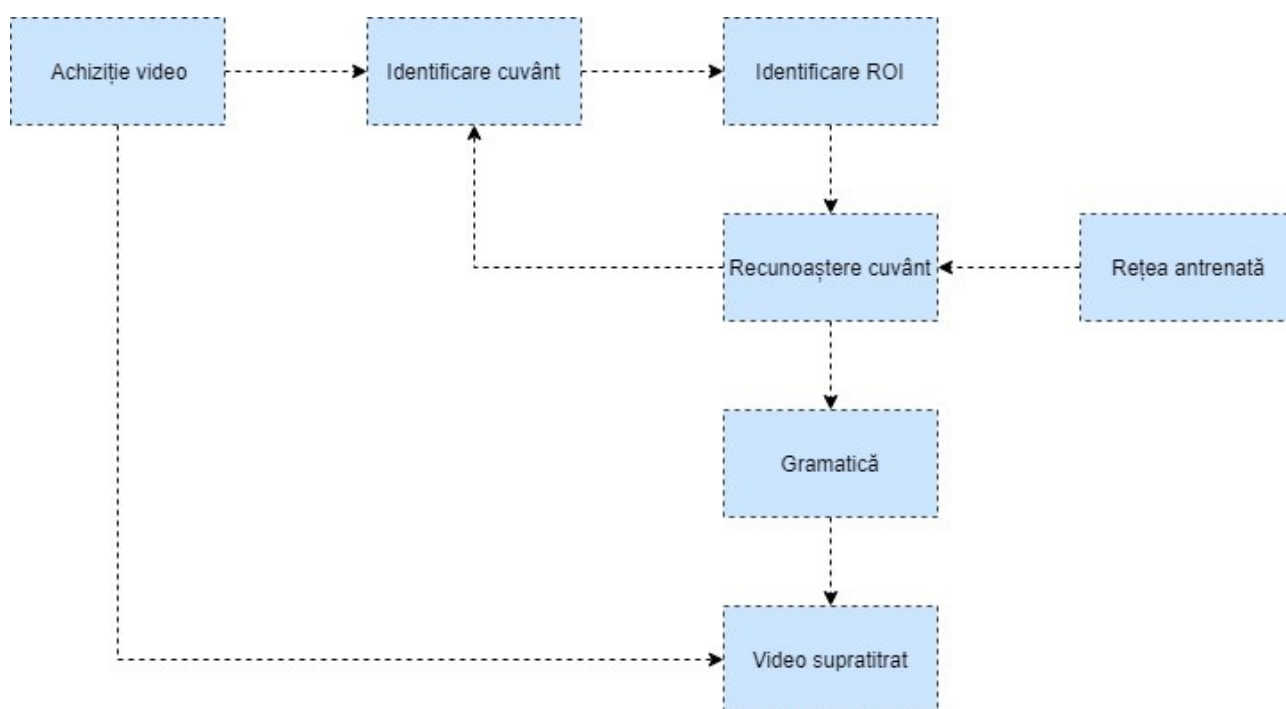


Figura 2.1: Organigrama sistemului automat de labiolectură propus

Vom analiza în cele ce urmează blocurile componente ale sistemului propus, observând structura acestora și modul lor de funcționare. În situațiile în care propunem mai multe implementări ale unui singur bloc, le vom prezenta pe fiecare, urmând ca rezultatele lor să fie analizate în Capitolul 3.

2.1 Achiziția video

Pentru achiziția fișierului video folosim două canale, unul pentru informația vizuală, al doilea pentru informația acustică. Cel de-al doilea canal, reprezentat de microfonul laptopului, este utilizat exclusiv pentru conformitate, având un scop pur academic, și anume cel de a verifica veridicitatea clasificării realizate de sistem. Înregistrarea audio este mai apoi suprapusă peste cea video folosind biblioteca FFmpeg [24].

Pentru obținerea informației vizuale în vederea procesării ei, se preia imaginea de la camera web a laptopului. Vom înregistra timp de 5 secunde, sau până la apăsarea tastei „q”, cu o viteză de 25 de cadre pe secundă (eng. Frames per second – FPS). În acest caz, vom obține până la 125 de cadre, cu rezoluția de 640 x 480 pixeli. Deoarece scopul proiectului este reprezentat de interpretarea mișcării buzelor, păstrarea imaginii întregi înseamnă procesarea unei informații redundante pe parcursul cadrelor. Vom încerca deci să păstrăm exclusiv zona ce conține buzele din acele cadre care reprezintă un cuvânt.

2.2 Identificarea unui cuvânt

În literatura de specialitate se prezintă mai multe modalități prin care se poate ocoli necesitatea identificării fiecărui cuvânt rostit. Vom analiza cele mai des întâlnite metode, urmând a propune o serie de variante fezabile. Astfel, o primă variantă este reprezentată de abordarea lui Assael et. al [22] care, realizând antrenarea sistemului la nivel de propoziție, nu are nevoie de separarea fiecărui cuvânt în vederea recunoașterii. Ținând cont că vom antrena sistemul propus la nivel de cuvânt, o astfel de abordare devine, de la început, inutilizabilă. O altă abordare este reprezentată de utilizarea ferestrelor de cadre, aceasta fiind, de asemenea, o variantă în care segmentarea devine redundantă. Această metodă presupune stabilirea unui număr fix de cadre, x , și selecția primului set de x cadre: $\{1...x\}$. Pe acest set se va realiza clasificarea, urmând a se reține rezultatul clasificării. Următorul set analizat este reprezentat de $\{2...x + 1\}$, pentru care se face din nou clasificarea și se obține un nou rezultat. Dintre cele două rezultate, se reține acela pentru care gradul de siguranță al rețelei este mai mare. Procedeu se repetă apoi până la parcurgerea întregului set de cadre obținut din înregistrare. Această metodă are două dezavantaje majore: numărul fix de cadre și complexitatea de calcul. În primul rând, numărul fix de cadre înseamnă fie segmentarea unor cuvinte lungi, pierzându-se astfel înțelesul lor, fie omiterea cuvintelor scurte, ducând la imposibilitatea înțelegerii frazei. Cea de-a doua problemă care apare în această situație este dată, așa cum aminteam, de complexitatea de calcul. Presupunând un număr de n cadre ce compun secvența video și o fereastră de x cadre, se obține o complexitate de:

$$O(n) = (n - x) + 1 \quad (2.1)$$

care înseamnă, în fapt, $O(n)$. Am avea, aşadar, n clasificări de realizat, ceea ce va creşte puternic timpul de rulare şi va scădea, implicit, utilitatea practică a unui astfel de sistem.

Propunem, aşadar, 3 metode pentru identificarea cuvintelor, urmând a analiza performanţele fiecăreia în Capitolul 3.

2.2.1 Metoda conturului

Pentru implementarea acestei metode am utilizat funcţii predefinite din biblioteca OpenCV [20], după cum se poate observa în Listarea 2.1 din Anexa 2. Este în primul rând necesar să identificăm regiunea în care se află gura vorbitorului. Pentru aceasta, vom folosi iterativ metoda Viola-Jones [25]. Într-o primă etapă, vom extrage regiunea în care se află faţa vorbitorului, folosind un clasificator specific, urmând ca apoi să selectăm zona în care se află gura subiectului, utilizând un alt clasificator. Vom presupune pentru simplitate că `frame` reprezintă regiunea din imagine în care se află buzele vorbitorului. Prima etapă este reprezentată de aplicarea unui filtru de estompare (eng. blur) imaginii, în vederea reducerii zgomotului existent. Imaginea este apoi transformată într-o imagine cu tonuri de gri, reducându-se astfel imaginea utilă de la trei canale la un singur canal. Se determină în următoarea etapă contururile (*edges*) din imagine, pe baza relaţiei dintre valoarea pixelilor şi valoarea pragului (*threshold*) aplicat. Ultima etapă este dată de preluarea tuturor conturilor şi a relaţiei dintre ele. Ne aşteptăm să observăm un singur contur, cel al gurii, şi graţie modificării coordonatelor punctelor ce alcătuiesc conturul să putem observa mişcarea gurii.

2.2.2 Metoda histogramei

Implementarea acestei metode are ca sursă de inspiraţie o observaţie făcută în urma încercărilor de a determina parametrii potriviţi pentru Metoda conturului. S-a observat faptul că, odată cu pronunţarea unui cuvânt, indiferent dacă acesta conţine consoane dentale sau vocale deschise, imaginea „se întuneacă”. Acest lucru ne-a făcut să determinăm histograma pixelilor ce formează regiunea în care se află gura, după cum se poate observa în Listarea 2.2 din Anexa 2. Ca în cazul expus la 2.2.1, folosim metoda Viola-Jones [25] pentru selecţia gurii subiectului, aplicăm un filtru de estompare pe imagine, apoi convertim imaginea în tonuri de gri. Următorul pas îl reprezintă determinarea vectorului `histogram`, care va conţine numărul de apariţii a fiecărei valori între 0 şi 255, numărul maxim fiind dat de numărul de pixeli din imagine. În final, vom determina valoarea medie a pixelilor din `frame` prin parcurgerea histogramei şi înmulţirea fiecărei valori cu numărul său de apariţii, ponderând această valoare cu dimensiunea imaginii pentru a menţine media în intervalul [0, 255]. Prin intermediul mediei, calculate pentru primele n cadre, putem stabili un prag superior şi un prag inferior care, odată depăşite, semnalează prezenţa unui cuvânt. Tot cu ajutorul mediei vom determina apariţia unei pauze în vorbire, prin stabilizarea mediei în jurul unei valori.

2.2.3 Metoda ariei

Această metodă a pornit ca răspuns la dependența pe care metodele prezentate la 2.2.1 și 2.2.2 o au față de iluminarea cadrului, cât și față de vorbitor. Astfel, în timp ce clasificatorii existenți pentru identificarea fețelor umane dau rezultate bune în condiții reale, cei utilizați în identificarea gurii conduc la rezultate slabe în anumite cazuri, precum prezența unor subiecți umani cu păr facial. O posibilă variantă în această situație este realizarea unor noi clasificatoare pe baza caracteristicilor de tip Haar, antrenate pe o varietate mai mare de subiecți. Deoarece acest aspect nu face obiectul prezentei lucrări, ne-am îndreptat atenția asupra altor modalități de detecție a gurii și, implicit, de identificare a cuvintelor.

Am ales, în final, să folosim metoda propusă de King [26], bazată pe histograma orientărilor gradientelor (eng. Histogram of Oriented Gradients – HOG). Prin intermediul acestei metode, fața umană, cât și principalele puncte de interes din aceasta (sprâncene, ochi, nas, contur, buze), pot fi identificate cu ajutorul unor repere faciale. Vom utiliza un detector deja antrenat, capabil să detecteze reperele faciale cu ajutorul a 68 de puncte, așa cum se poate observa în Figura 2.2. Folosind punctele 49, 52, 55 și 58, vom determina aria exterioară a gurii, aproximând forma conturului cu o elipsă, așa cum se poate observa în Listarea 2.3 din Anexa 2.

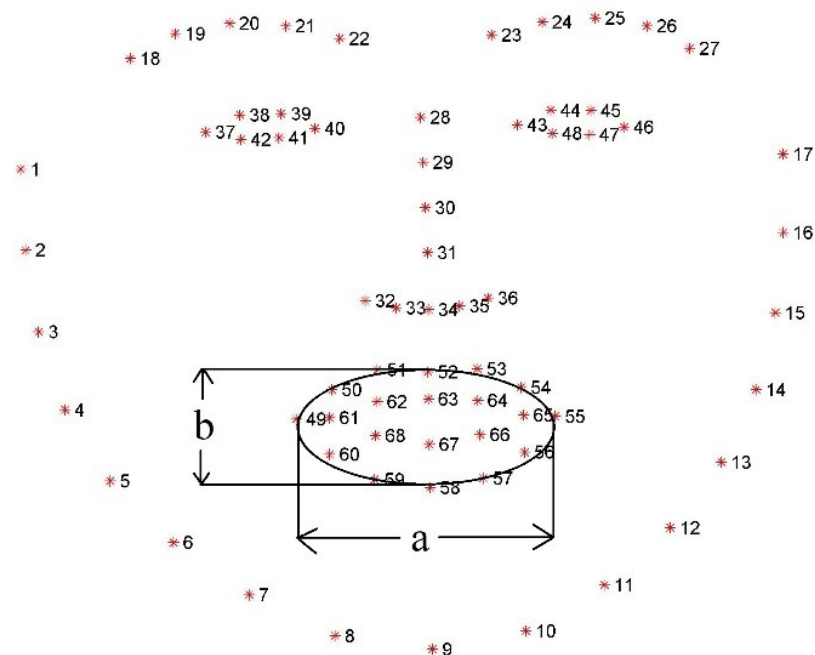


Figura 2.2: Determinarea ariei gurii cu ajutorul celor 68 de repere faciale. Punctele 52 și 58 sunt utilizate pentru determinarea semiaxe mici a elipsei, iar punctele 49 și 55 pentru semi-axa mare. Pe baza lor se poate calcula aria gurii.

Se obține astfel aria gurii:

$$A = \pi * \frac{a}{2} * \frac{b}{2} \quad (2.2)$$

care poate fi utilizată pentru determinarea începutului și sfârșitului unui cuvânt. Astfel, se pot stabili praguri inferioare și superioare la depășirea cărora putem afirma că subiectul pronunță un cuvânt. Sfârșitul cuvântului poate fi identificat prin stabilizarea ariei în jurul unei anumite valori. În acest mod, putem asigura selecția exclusivă a cadrelor care conțin cuvinte ce urmează a fi recunoscute.

2.3 Identificarea regiunii de interes

Regiunea de interes (eng. Region Of Interest – ROI sau Area Of Interest – AOI) reprezintă acea zonă din imagine care conține informație utilă sistemului. Deoarece ne propunem să implementăm un sistem automat de labiolectură, regiunea de interes o reprezintă gura vorbitorului. Informație utilă poate fi găsită și în restul feței, dată de ridurile de expresie la pronunțarea anumitor foneme, însă această informație este mult mai puțin relevantă comparativ cu cea oferită de mișcarea buzelor. De aceea, vom selecta din cadrele segmentate doar zona care conține gura vorbitorului.

La fel ca în cazul blocului prezentat în cadrul subcapitolului 2.2, și pentru această etapă propunem mai multe implementări posibile, urmând a le analiza din punct de vedere experimental în Capitolul 3. Implementările sunt inspirate de structurile din subcapitolul anterior, ținând însă cont de faptul că detecția feței folosind caracteristicile de tip Haar este mai rapidă decât cea prin intermediul celor 68 de repere faciale. În plus, structura prezentată în acest subcapitol reprezintă de asemenea un sub-bloc al blocului „Rețea antrenată”, ceea ce face ca presupunerea că deja cunoaștem poziția gurii din blocul „Identificare cuvânt” să nu fie întotdeauna validă, căci acest bloc este absent în cadrul antrenării rețelei.

2.3.1 Metoda directă

Această metodă presupune utilizarea soluției dezvoltate de Viola-Jones [25] în vederea detecției gurii, prin utilizarea unui clasificator specific acestui proces. Pentru reducerea erorilor, vom impune căutarea gurii exclusiv în jumătatea inferioară a cadrului, fără a aplica nicio altă constrângere. Ne așteptăm ca această variantă să dea cele mai slabe rezultate, dată fiind aria mare în care posibila gură poate fi identificată.

2.3.2 Metoda indirectă

Metoda indirectă pornește de la dezavantajele anterior menționate în cadrul metodei directe, și anume aria mare în care sistemul este nevoit să identifice buzele vorbitorului. Acest aspect conduce atât la întâzieri în fluxul de procesare, cât și la creșterea ratei de eroare în identificarea gurii. Pentru a reduce această rată, introducem un pas suplimentar, anterior detecției gurii: identificarea feței. Date fiind rezultatele bune ale clasificatorilor existenți pentru detecția fețelor subiecților umani, vom realiza acest pas într-o primă etapă, urmând a căuta gura în subcadrul nou obținut. Restrângând aria de căutare, ne așteptăm să restrângem implicit și posibilitățile de eroare.

2.3.3 Metoda indirectă cu memorie

Apărută ca o completare a metodei indirecte, prezentate în paragraful anterior, metoda indirectă cu urmărire își propune să elimine situațiile în care sistemul detectează corect gura în *frame*-ul n , dar nu reușește să o identifice cu acuratețe în *frame*-ul $n+1$. Această situație apare, în general, în momentele în care subiectul uman începe să vorbească, fapt ce conduce la creșterea unghiului de deschidere a maxilarului. Clasificatorii ce folosesc caracteristici de tip Haar sunt sensibili la astfel de situații, fapt ce conduce la apariția erorilor. Pentru a preîntâmpina aceste cazuri, propunem un algoritm similar celui descris în cadrul 2.3.2, dar care, în cazul în care în cadru nu a fost detectată gura vorbitorului, preia un sub-cadru folosind coordonatele determinate la pasul anterior. Astfel, sintagma „cu memorie” din denumirea metodei provine din faptul că algoritmul memorează coordonatele determinate la pasul precedent, în vederea utilizării lor la pasul curent.

2.3.4 Metoda cu cadru fix

Această metodă a fost dezvoltată ținând cont de lucrarea lui Khan et. al [27], care arată că există un set de proporții standard ale localizării spațiale a trăsăturilor faciale. Acest fapt ne permite să eliminăm apelul la identificarea gurii prin intermediul caracteristicilor de tip Haar și să avem o altă abordare în vederea selecției regiunii care conține gura vorbitorului. Așadar, prima etapă a acestei metode o reprezintă selecția ariei dată de fața subiectului, identificată prin clasificatorii amintiți anterior. Pentru preluarea sub-cadrului dorit, cel în care se află gura vorbitorului, vom folosi proporțiile identificate de [27], alături de observațiile noastre, așa cum se poate observa în Figura 2.3.

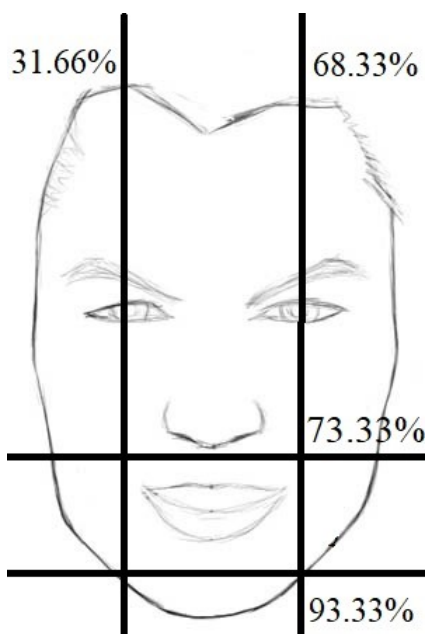


Figura 2.3: Selecția regiunii de interes prin metoda cu cadru fix. Propunem ca gura să fie definită ca o zonă specifică ce ocupă 20% din față pe axa verticală și 36.66% pe axa orizontală. Dreptunghiul astfel obținut asigură încadrarea gurii indiferent de gradul de deschidere a acesteia.

Aria de interes pe care o vom selecta este, deci, dată de un set fix de valori relativ la dimensiunile cadrului ce conține fața, în această arie de interes aflându-se gura vorbitorului, cât și o zonă-tampon care acoperă toate cazurile, de la o deschidere amplă a maxilarului la unirea completă a buzelor. Ne așteptăm ca această metodă să aibă cele mai bune rezultate, dată fiind independența ei de clasificatorii utilizați pentru detecția gurii.

2.4 Recunoașterea cuvântului

Odată determinate cadrele în care subiectul pronunță un cuvânt și selectarea acelei porțiuni din cadru în care se pot observa buzele subiectului, se poate realiza recunoașterea cuvântului. Pentru aceasta, este necesară stabilirea unui set de parametri. În primul rând, trebuie specificat numărul de cadre pe baza cărora se va realiza recunoașterea. În cazul în care numărul de *frame*-uri stabilit este mai mare decât cel extras, recunoașterea nu se va putea realiza. O soluție pentru rezolvarea acestei situații o presupune fie bordarea secvenței de cadre cu cadre duplicat ale ultimului *frame*, fie implementarea unei supraeșantionări *software*, altfel spus dublarea fiecărui cadru existent. Pe de altă parte, în cazul în care numărul de *frame*-uri stabilit este mai mic decât cel existent, se va realiza o scalare a listei de *frame*-uri, urmând a se returna fiecare al n -lea element, cu n reprezentând partea întreagă a raportului dintre numărul de cadre existente și numărul de cadre cerute.

Cel de-al doilea parametru ce trebuie luat în considerare este reprezentat de rețeaua neuronală care va fi folosită pentru clasificare. Astfel, pe lângă specificarea modelului antrenat, dat de blocul „Rețea antrenată”, este necesară cunoașterea cerințelor pe care fiecare rețea le are. Deoarece rețelele folosite vor fi analizate în detaliu în secțiunea 2.7, vom menționa acum doar faptul că este necesară fie extragerea trăsăturilor setului de date utilizat pentru recunoaștere, fie specificarea dimensiunii fiecărui *frame* care intră în componența cuvântului ce se vrea a fi recunoscut.

2.5 Gramatica

Deoarece ne propunem realizarea unui sistem automat de labiolectură a cărei arhitectură să fie independentă de limba vorbită de subiectul înregistrat, existența acestui bloc este opțională. El va fi folosit exclusiv pentru îmbunătățirea rezultatelor sistemului la antrenarea cu cuvinte în limba română, dat fiind setul redus de date ce urmează a fi folosit în acel caz. Pentru a realiza această îmbunătățire, vom utiliza două seturi de reguli, un set indulgent și un set strict. Primul va constrânge sistemul să nu aibă repetiții, astfel încât două entități consecutive nu vor fi niciodată reprezentate de același cuvânt. Cel de-al doilea set de reguli va specifica propozițiile posibile ce pot fi alcătuite prin combinații ale cuvintelor existente în setul de date. Pentru ca implementarea acestor reguli să fie posibilă, vom utiliza primele două rezultate oferite de rețea, ordonate descrescător în funcție de nivelul de siguranță. Vom verifica dacă, folosind cel mai bun rezultat pentru fiecare cuvânt, obținem o propoziție care să corespundă cu setul de reguli impus. Dacă acest lucru nu se întâmplă, vom înlocui, pe rând, fiecare cuvânt cu perechea sa, în ordinea nivelurilor de siguranță, până la identificarea unei propoziții care să respecte setul de reguli impus.

2.6 Video supratitrat

Această etapă presupune exclusiv vizualizarea rezultatelor rețelei într-o formă ușor de interpretat. Astfel, folosind rezultatele oferite de blocul 2.4 sau 2.5, cât și secvența audio-video obținută de blocul 2.1, vom crea un film care să înglobeze datele de intrare și pe cele de ieșire ale sistemului. Pentru realizarea acestei operații, vom folosi biblioteca de procesare multimedia FFmpeg. Rezultatul final va fi reprezentat de o secvență video în care subiectul poate fi văzut și auzit rostind un număr de cuvinte, cuvinte ce vor fi redade sub formă de text în cadrul secvenței.

2.7 Rețeaua antrenată

Pentru a putea recunoaște cuvintele rostite de subiectul înregistrat, este necesară preantrenarea unei rețele neuronale folosind un set de date de dimensiuni cât mai mari. Structura rețelei neuronale este definitorie pentru buna funcționare a sistemului, modul de implementare a acesteia conducând la obținerea rezultatelor dorite. În lucrarea curentă am optat pentru implementarea a patru modele de rețele neuronale artificiale adânci pornind de la proiectul lui Harvey [28], urmând a observa în Capitolul 3 care sunt rezultatele fiecăreia dintre ele și, mai apoi, alegând o singură structură care să facă parte din sistemul final. Toate aceste modele țin seama de faptul că scopul proiectului este recunoașterea unor cuvinte reprezentate ca o secvență video, ceea ce înseamnă că este necesară implementarea unor rețele spațio-temporale. Folosirea unor modele convoluționale bidimensionale ar fi, în această situație, insuficientă, așadar vom utiliza, în plus, rețele neuronale recurente (eng. Recurrent Neural Network – RNN). Acestea au proprietatea că oferă rezultate nu doar în funcție de intrarea curentă, ci și în funcție de restul intrărilor furnizate anterior.

2.7.1 Rețeaua Inception-V3 & Multilayer Perceptron

Primul model construit are ca punct plecare lucrarea lui Szegedy et al. [29], care a obținut rezultate mai bune decât cele mai performante sisteme de la acel moment, pentru teste realizate pe setul ILSVRC 2012. Inception-V3 reprezintă, în fapt, o scalare a primei versiuni Inception, obținându-se astfel o rețea neuronală adâncă ce realizează clasificări cu o acuratețe foarte bună (78.8% pe ILSVRC 2012 pentru evaluarea unui singur cadru). Cu ajutorul acestui model, vom extrage trăsăturile din fiecare cadru care alcătuiește cuvântul ce se dorește a fi recunoscut, numărul de cadre per cuvânt fiind stabilit a priori și notat cu *seq* în Figura 2.4.

Următoarea etapă o presupune obținerea unei relații temporale între trăsăturile determinate anterior, motiv pentru care vom realiza o structură recurentă cu intrare vectorială unidimensională. În particular, vom construi o rețea de tip Perceptron Multistrat (eng. Multilayer Perceptron – MLP). Structura acesteia are ca punct de plecare perceptronul simplu, diferența față de acesta fiind dată de numărul de unități funcționale folosite, cât și de stratificarea și interconectarea acestor unități. În acest caz, vom folosi două straturi cu câte 512 neuroni, cu o conexiune de tip *feed-forward*, fără funcție de activare, astfel încât fiecare neuron să realizeze exclusiv însumarea intrărilor ponderate. În plus, vom

introduce o rată de regularizare de 50% pe fiecare strat, motivele fiind cele arătate în subcapitolul 1.2.6.

În final, rețeaua conține un strat suplimentar alcătuit dintr-un număr variabil de neuroni, numărul acestora fiind dat de numărul claselor existente. Acest strat folosește ca funcție de activare *softmax*, cunoscută și drept exponențiala normalizată:

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^K e^{z_k}} \quad (2.3)$$

unde z este un vector K -dimensional, iar j reprezintă indicele elementului curent. Astfel, diferența față de o funcție sigmoidală uzuală este că se poate obține distribuția probabilităților de apartenență a evenimentului curent, în cazul nostru a cuvântului rostit, la fiecare clasă. În plus, rezultatul funcției va fi tot timpul pozitiv și subunitar, fapt ce conduce la o reducere a complexității computaționale, iar suma tuturor probabilităților de apartenență a unui cuvânt la clasele existente va fi mereu 1. În final, se obține modelul prezentat în Figura 2.4, model al cărui comportament va fi analizat în capitolul următor.

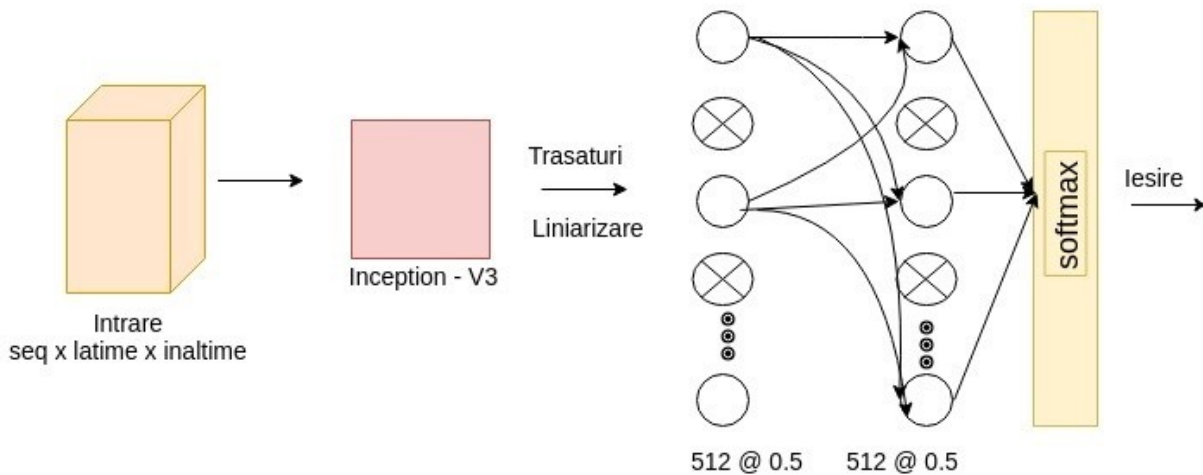


Figura 2.4: Rețeaua Inception-V3 & MLP. Intrarea tridimensională trece prin rețeaua Inception-V3 în vederea obținerii trăsăturilor, apoi acestea sunt folosite pentru antrenarea unei structuri MLP care oferă rezultatele clasificării

2.7.2 Inception-V3 & Long short-term memory

Simplitatea soluției descrise anterior poate conduce la rezultate sub așteptări, drept care propunem o a doua soluție bazată pe aceeași structură. La fel ca în primul model propus, vom folosi rețeaua Inception-V3 [29] pentru determinarea trăsăturilor fiecărui *frame* ce intră în compoziția cuvântului, numărul de cadre per cuvânt fiind fix și notat cu *seq* în Figura 2.5. În acest caz, însă, secțiunea recurentă a rețelei va fi implementată prin intermediul unui *Long short-term memory* (LSTM) definite în biblioteca Keras. Conceptul de LSTM apare în 1997 [30] și propune o rezolvare

a uneia dintre principalele probleme care apar la utilizarea rețelelor neuronale recurente: dependențele pe termen lung. Astfel, în timp ce un RNN este alcătuit în mod uzual dintr-o înlănțuire de module ce conțin un singur strat neuronal, modulul repetitiv al unui LSTM este alcătuit din patru astfel de straturi interconectate. Se obține astfel o dependență mai mare a predicției curente de cele anterioare, relația dintre informația curentă și predicțiile straturilor precedente nefiind definită de o singură funcție matematică, așa cum este cazul unor RNN-uri simple (i.e. funcția tangentă hiperbolică). Rezultatul final al acestei structuri este astfel influențat de toate stările anterioare ale sistemului, acesta prezentând deci o memorie pe termen lung.

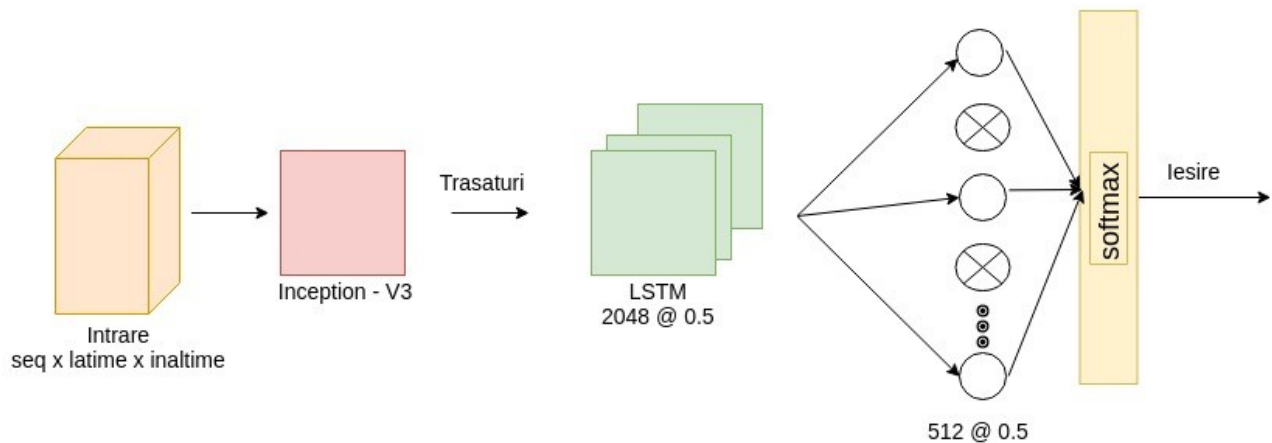


Figura 2.5: Rețeaua Inception-V3 & LSTM. Intrarea tridimensională trece prin rețeaua Inception-V3 în vederea obținerii trăsăturilor, apoi acestea sunt folosite pentru antrenarea unei structuri LSTM, urmată de un strat de unități funcționale care oferă rezultatele clasificării

În modelul propus vom oferi trăsăturile determinate cu ajutorul Inception-V3 ca intrare pentru 2048 de LSTM-uri, fiecare analizând temporal o anumită trăsătură. Stabilim parametrul de regularizare la 50%, iar ieșirile acestui strat sunt mai apoi conectate la 512 unități funcționale cu același *dropout* și cu o funcție de activare de tip ReLU. Acest factor de regularizare face ca, pe parcursul procesului de antrenare a rețelei, anumiți neuroni să se dezactiveze și să se reactiveze aleator, fapt ce conduce la scăderea probabilității de supraantrenare. Stratul de unități funcționale este complet activ doar în procesul de clasificare a lotului de validare, acolo unde rețeaua trebuie utilizată la capacitate maximă în vederea obținerii celor mai bune performanțe. Vom obține în final 256 de valori de care ajung într-un strat cu n neuroni, unde n reprezintă numărul de clase. Folosind o funcție de activare de tip *softmax*, vom determina ieșirea rețelei, probabilitatea de apartenență a cuvântului rostit la fiecare dintre cele n clase. Structura acestei rețele neuronale adânci poate fi observată în Figura 2.5.

2.7.3 Long-term recurrent convolutional network

Cea de-a treia structură propusă are ca punct de plecare lucrarea lui Assael et al. [22], care folosește o rețea antrenată *end-to-end*. Astfel, în timp ce la soluțiile anterioare am extras trăsăturile separat, apoi am antrenat exclusiv partea a doua a rețelei folosind metodologia supervizată, pentru structura prezentă vom aborda procesul ca un întreg. Această aspect înseamnă, în fapt, că diferențele dintre ieșirea dorită și cea reală vor conduce la modificarea tuturor ponderilor existente în sistem. Vom realiza așadar o rețea antrenată *end-to-end*, pornind de la structura prezentată de Donahue et al. [31], modelul propus de noi fiind prezentat în Figura 2.6 și denumit *Long-term Recurrent Convolutional Network (LRCN)*.

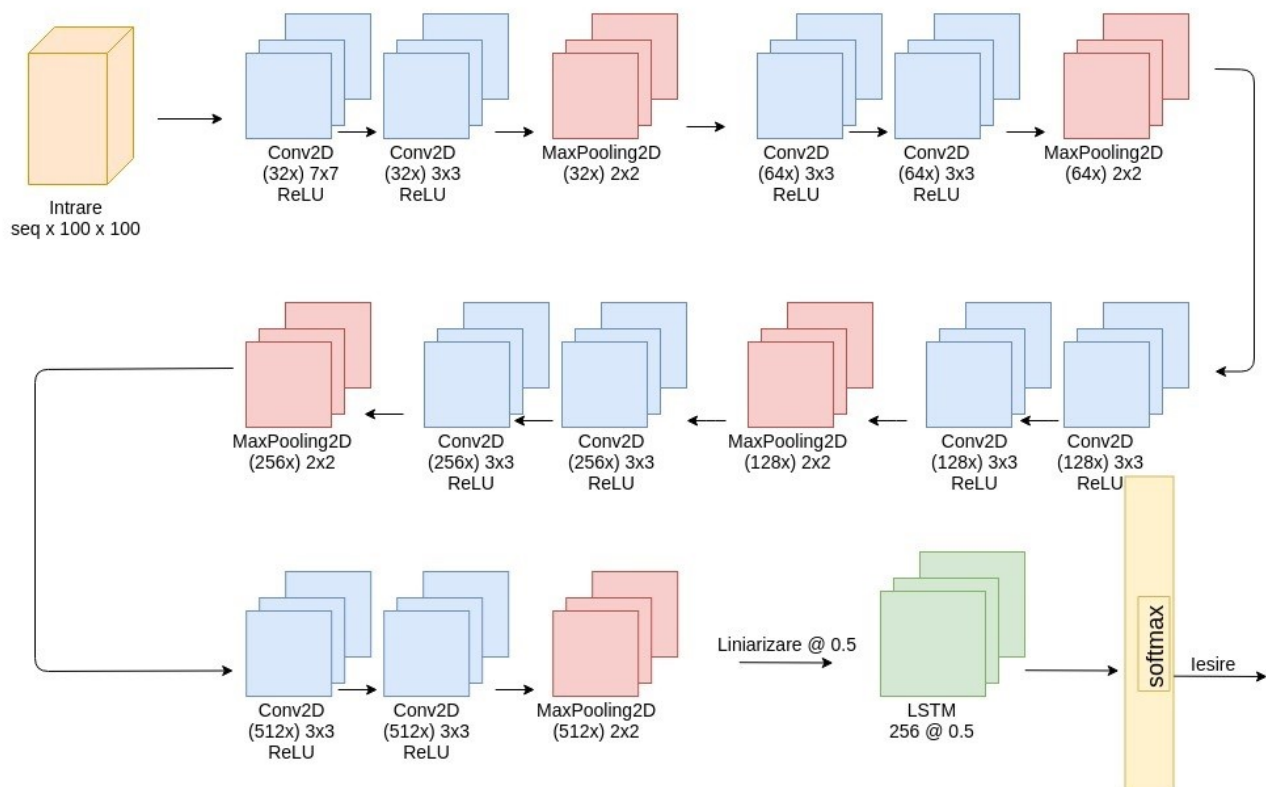


Figura 2.6: Rețeaua LRCN. Intrarea tridimensională este trecută printr-o serie de structuri convoluționale și procese de subeșantionare spațială în vederea obținerii trăsăturilor, care sunt mai apoi utilizate pentru antrenarea unei structuri LSTM. Stratul final realizează punerea în comun a rezultatelor, oferind rezultatele clasificării

Intrarea rețelei este reprezentată de un set de *seq* cadre de dimensiune fixă, stabilită anterior. Se realizează apoi convoluția cu un *kernel* inițializat aleator de dimensiune 7x7, iar rezultatul este apoi dat ca argument unei funcții de activare de tip ReLU. Apoi, 32 de astfel de convoluții se realizează, fiecare urmărind o altă trăsătură. Ele devin intrări ale unei alte structuri convoluționale similare, dar cu *kernel* de dimensiune 3x3. Rezultatelor convoluției li se aplică o subeșantionare spațială prin selecția unor submatrice de 2x2 din care se extrage valoarea maximul, fără a realiza suprapunerea matricelor de parcurgere. Procesul se repetă apoi de patru ori, diferențele apărând doar

în numărul de convoluții realizate simultan și în dimensiunea matricei de convoluție. Rezultatul acestei etape convoluționale este transformat într-un vector unidimensional, datele sunt reduse la jumătate, apoi sunt trecute printr-un set de 256 LSTM-uri paralele cu rata de regularizare de 50%, rezultatul final fiind obținut prin aplicarea unei funcții de activare de tip *softmax* pe ultimul strat al modelului neuronal propus.

2.7.4 Convolutional 3D

Cel de-al patrulea model de rețea neuronală propune o abordare tridimensională, fiind prezentat în Figura 2.7.

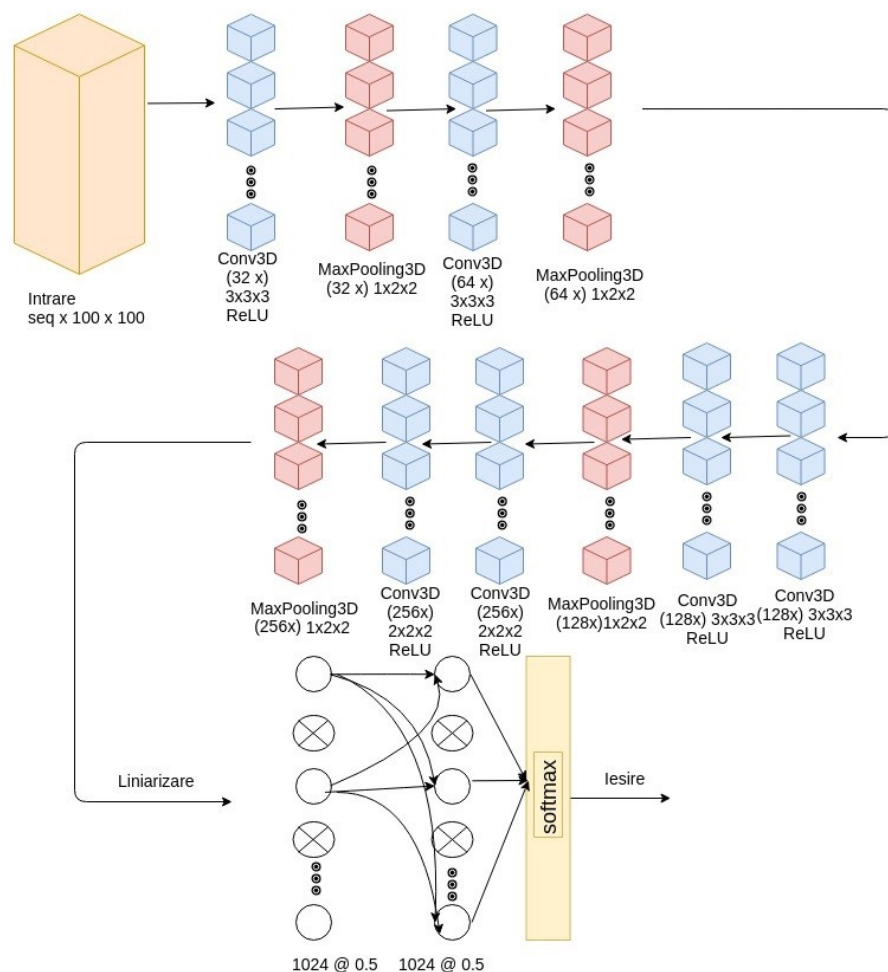


Figura 2.7: Rețeaua C3D. Intrarea tridimensională este trecută printr-o serie de structuri convoluționale și procese de subeșantionare spațială în vederea obținerii trăsăturilor, care sunt apoi folosite pentru antrenarea unei structuri MLP care oferă rezultatele clasificării

Astfel, cunoaștem că imaginile au două dimensiuni: lățime și înălțime. Vom omite cea de-a 3-a dimensiune reală, dată de faptul că o imagine este reprezentată de 3 matrice de culoare (roșu, verde și albastru), deoarece aceasta se absoarbe în rețeaua convoluțională. În situația noastră, ce-a de-a treia dimensiune este dată de timp, reprezentând numărul de cadre ce urmează să fie analizate ca aparținând unui singur cuvânt. Vom porni de la structura prezentată de Tran et. al [32], care au obținut

o acuratețe de 52.8% pe setul de date UCF101, folosit pentru recunoașterea acțiunilor în secvențe video. Adaptând modelul propus de aceștia, obținem structura denumită C3D.

Comportamentul părții convoluționale este similar celui descris în paragraful anterior, diferența fiind dată de apariția celei de-a treia dimensiune. Astfel, avem o secvență de operații de convoluție, pentru care *kernelul* este reprezentat de matrice de dimensiune $3 \times 3 \times 3$ sau $2 \times 2 \times 2$. Acestor convoluții li se intercalează operații de subeșantionare spațială care extrag valorile maxime din datele inițiale. Așa cum se poate observa din figură, operația de *MaxPooling* se realizează efectiv doar pentru două dimensiuni, înălțimea și lățimea imaginii, valorile obținute în urma convoluției nefiind analizate din punct de vedere temporal. Aceste succesiuni de convoluții și subeșantionări sunt urmate apoi de o reducere a rezultatului la o singură dimensiune. Următoarea etapă este similară cu cea implementată pentru rețeaua Inception-V3 & MLP. Ca și în celălalte cazuri, rezultatul final este obținut prin intermediul unui ultim strat, alcătuit din n neuroni, unde n reprezintă numărul de clase. Folosind o funcție exponențială normalizată ca funcție de activare, vom obține n valori între 0 și 1, unde valoarea maximă va fi obținută pentru clasa cu cel mai mare grad de similaritate cu secvența dată la intrare. În capitolul 3 vom analiza performanțele acestui model neuronal, deopotrivă cu cele ale rețelelor descrise anterior, urmând a alege o singură rețea care să devină parte integrantă a sistemului automat de labiolectură pe care dorim să-l implementăm.

Capitolul 3

Rezultate experimentale

În acest capitol vom analiza performanțele sistemului automat de labiolectură propus în această lucrare, urmând a vedea care dintre implementările prezentate în capitolul anterior este cea optimă. În primul rând, vom descrie exhaustiv seturile de date folosite pentru antrenarea rețelelor neuronale. Deoarece ne propunem să realizăm un sistem capabil să recunoască cuvinte în limba română, am decis să realizăm o bază de date redusă cu astfel de cuvinte. O a doua parte a acestui capitol o va constitui descrierea metricilor de evaluare a sistemului. Cea de-a treia etapă o va presupune analiza metodelor propuse în Capitolul 2, cât și a parametrilor ce influențează sistemul, astfel încât să obținem o structură cu performanțe cât mai bune.

3.1 Descrierea seturilor de date

Am afirmat în capitolul anterior faptul că ne propunem realizarea unui sistem de labiolectură independent de limba vorbită de subiectul uman înregistrat. Acest lucru presupune nu doar o construcție cât mai generală a sistemului, ci și antrenarea și analiza acestuia în condiții diverse din punctul de vedere al limbii. Pentru a realiza acest lucru, vom alege pentru testare limbile engleză și română. Motivul alegerii celei din urmă este evident, dat de tema aleasă pentru acest proiect. Pe de altă parte, am ales limba engleză datorită numărului suficient de seturi de date existente, prezentate în Tabelul 1.1.

Dintre seturile de date specificate în tabelul anterior menționat, am ales să folosim pentru antrenare baza GRID, disponibilă pentru utilizare la [33]. Ea a fost realizată în 2006 de Cooke et. al [34] și este în continuare una dintre cele mai utilizate baze de date pentru recunoașterea vorbirii, a vorbitorului sau pentru sisteme automate de labiolectură. Descrierea acestui set de date poate fi observată în Tabelul 3.1, în care vom realiza o comparație cu setul de date pe care l-am realizat pentru această lucrare.

Tabel 3.1: Descrierea seturilor de date utilizate pentru antrenare

* Frazele din GRID au o structură fixă: acțiune (4) - culoare (4) - prepoziție (4) - literă (25) - cifră (10) – adverb (4). Acest lucru face ca pe o poziție din frază să existe, în medie, 8.5 cuvinte.

** Frazele tip conțin 21 de cuvinte, dintre care 9 cuvinte apar o dată și 6 cuvinte de 2 ori.

<i>Setul de date</i>	<i>GRID</i>	<i>GRID redus</i>	<i>LAPI - LIRO</i>
<i>Proprietăți</i>			
<i>Numărul de cuvinte unice</i>	51	15	15
<i>Probabilitatea de apariție a unui cuvânt</i>	11.76%*	6.66%	6.66%
<i>Numărul de eșantioane per cuvânt</i>	4000	300	15/30**
<i>Numărul și sexul subiecților</i>	18M/16F		13M/2F
<i>Gama vârstelor subiecților</i>	18 ani – 49 de ani		20 de ani – 23 de ani
<i>Mediul înregistrării</i>	Laborator		Laborator
<i>Tipul înregistrării</i>	Înregistrare continuă		Înregistrare continuă
<i>Camera video utilizată la înregistrare</i>	Canon XM2		Nikon D5300
<i>Numărul de cadre pe secundă</i>	25 FPS		50 FPS

Dat fiind faptul că un set de date în limba română reprezintă doar un auxiliar al acestei lucrări, cât și dificultatea găsirii unor subiecți în vederea realizării înregistrărilor, am implementat un set-prototip. Acesta are ca scop identificarea unor reguli de bune practici pentru crearea ulterioară a unei baze mai vaste, dar și utilizarea sa pentru antrenarea și testarea sistemului propus. Având ca sursă de inspirație GRID, am realizat alături de Andreea Munteanu, membru al Laboratorului Multimedia din cadrul Centrului CAMPUS, setul de date CAMPUS – α LIRO (i.e. Centrul de Cercetări Avansate pentru Materiale, Produse și Procese Inovative – α LIpreading Romanian). Proprietățile acestui set-prototip pot fi observate în Tabelul 3.1, iar subiecții care au participat la această realizare sunt prezentați în Figura 3.1. Înregistrările au fost realizate la o distanță de aproximativ un metru de subiecți, cu camera video plasată la înălțimea ochilor. Am folosit pentru iluminare lumină naturală, iar fundalul a fost în majoritatea cazurilor alb monocromatic. Fiecare vorbitor a avut o poziție fixă pe întreaga durată a înregistrărilor, însă nu s-a impus ca toți vorbitorii să aibă aceeași poziție. Fiecărui subiect i s-a permis repetarea capturii video până la obținerea unei versiuni lipsite de erori. S-au înregistrat 5 fraze de lungimi diferite pentru fiecare dintre cei 15 participanți, totalizând un număr de 21 de cuvinte, dintre care 15 cuvinte unice (i.e. acesta, anul, electronica, este, eu, imagini, la, licența, mea, pasiunea, pentru, prezint, procesez, student, sunt). Pentru variații ale cuvintelor articulate/nearticulate (e.g. „electronica” și „electronică”), am considerat ambele variante ca fiind același cuvânt, dat fiind gradul de similitudine dintre labiolemele fonemelor „a” și „ă”. De asemenea, cuvinte ce conțin diacritice, precum „licența”, au fost reprezentate fără diacritice (e.g. „licența”), fapt

care nu influențează rezultatele, dat fiind faptul că nu realizăm o antrenare la nivel de fonem. Folosind o unealtă de editare video, am realizat manual adnotările după modelul GRID, specificând pentru fiecare înregistrare momentul de început și momentul de final al fiecărui cuvânt. În plus, am folosit biblioteca FFmpeg pentru a reduce numărul de cadre pe secundă din fiecare înregistrare de la 50 FPS la 25 FPS.



Figura 3.1: Subiecții setului de date CAMPUS – α LIRO

Deoarece ne propunem să realizăm o comparație cât mai relevantă între rezultatele sistemului pentru limbile engleză și română, am folosit o variantă redusă a GRID. Utilizarea întregului set de date ar fi fost neconcludentă, date fiind diferența de două ordine de mărime dintre numărul de eșantioane per cuvânt al GRID și cel al CAMPUS – α LIRO, cât și numărul diferit de cuvinte. Așadar, am restrâns GRID la GRID redus, descris în Tabelul 3.1, set care conține 15 cuvinte (i.e. bin, blue, eight, five, four, nine, now, one, please, seven, six, soon, three, two, zero) pronunțate de 20 de ori fiecare, vorbitorii fiind aleși aleator dintre cei 34 existenți.

Date fiind diferențele dintre cele două seturi de date folosite, ne așteptăm ca rezultatele obținute să fie semnificativ mai bune în cazul utilizării GRID decât în cazul utilizării CAMPUS – α LIRO, motivele fiind multiple. În primul rând, diferența de un ordin de mărime dintre cele două va face ca rețeaua antrenată cu GRID să învețe mai bine. De asemenea, faptul că în CAMPUS – α LIRO există cuvinte ce apar de două ori va face ca sistemul să învețe mai bine acele cuvinte decât pe cele care apar o singură dată. În plus, condițiile de iluminare și fundalul neuniform în cazul setului de date în limba română va produce zgomot în imagine, zgomot ce se va propaga în toată rețeaua. O altă problemă o reprezintă dimensiunea cuvintelor: cuvintele din GRID pot fi pronunțate într-un timp aproximativ egal, în timp ce cele folosite pentru CAMPUS – α LIRO au lungimi ce variază puternic, aspect ce va afecta în sens negativ antrenarea, dat fiind faptul că ea se face folosind un număr fix de

cadre pentru toate cuvintele. Nu în ultimul rând, factorul uman poate conduce la diferențe între rezultatele din limba română, respectiv engleză, prin faptul că adnotările au fost realizate manual, dar și din cauza lipsei de expresivitate a subiecților înregistrați.

3.2 Metrice de evaluare

În analiza comportamentului unei rețele neuronale, există un set standard de variabile care sunt luate în considerare. Cunoașterea acestora și a variației lor conduce la înțelegerea modului de funcționare a unui algoritm DL. În primul rând, trebuie ținut cont de mărimea care stabilește momentul în care considerăm că rețeaua este antrenată. În cazul lucrării curente, vom utiliza ponderile salvate în momentul în care scăderea lui *val_loss* stagnează. Acest parametru este dat de valoarea funcției de cost calculată pentru setul de date de validare, valoare care diferă de cea a funcției de cost calculată pentru setul de date de antrenare, denumită *loss*. O observație ce se cere a fi făcută este că straturile rețelei care prezintă un factor de regularizare nu mai iau în considerare acest *dropout* în calculul lui *val_loss*. De asemenea, trebuie menționat că în această lucrare vom folosi ca funcție de cost *categorical_crossentropy*:

$$L(\theta) = -\frac{1}{n} \sum_{i=1}^n \sum_{j=1}^m y_{ij} * \log(p_{ij}) \quad (3.1)$$

unde n reprezintă numărul de eșantioane per clasă, m este numărul de clase, y este eticheta eșantionului, iar p reprezintă predicția eșantionului. Alegerea acestei funcții s-a făcut prin studierea literaturii de specialitate, date fiind rezultatele bune obținute prin utilizarea ei.

3.2.1 Acuratețea

Principalul mod prin care vom evalua performanța rețelelor neuronale utilizate este reprezentat de acuratețea acestora. Această mărime poate fi descrisă prin intermediul unei comparații cu o clasificare binară, conform procedurii descris de Ionescu și Mironică [35]. Astfel, se definesc patru situații posibile: *false positive (fp)*, *false negative (fn)*, *true positive (tp)*, *true negative (tn)*. Prima reprezintă clasificarea unui obiect ca aparținând clasei A, el fiind inclus în clasa B. Cea de-a doua înseamnă prezicerea unui cuvânt ca fiind reprezentativ pentru clasa B, el aparținând clasei A, cea urmărită. *True positive* înseamnă clasificarea corectă a unui obiect de tip A în clasa A, iar *true negative* se definește ca prezicând apartenența unui obiect de tip B în clasa B. Se obțin astfel două valori importante în analiza clasificării:

$$precision = \frac{tp}{tp + fp}, \quad recall = \frac{tp}{tp + fn} \quad (3.2)$$

În practică, „*precision* este o măsură a falselor detecții iar *recall* o măsură a non-detețiilor”[35]. În cazul lucrării curente, dată fiind natura aplicației, vom folosi o mărime care are ca punct de plecare precizia, mărime denumită acuratețe. Ea se va calcula ca fiind media de apariție a

predicțiilor realizate cu precizie mare. Vom reprezenta acuratețea fiecărei rețele în situații variate pentru a o putea alege pe cea optimă aplicației propuse.

3.2.2 Word Error Rate

Rata de cuvinte eronate (eng. Word Error Rate – WER) este unul dintre cei mai importanți parametri utilizați în descrierea modului de funcționare a unui sistem de recunoaștere a vorbirii, fie că el are ca date de intrare informație vizuală sau acustică. WER se definește ca:

$$WER = \frac{S + D + I}{S + D + C} \quad (3.3)$$

unde S reprezintă numărul de substituții (cuvinte care au fost clasificate eronat), D este numărul de cuvinte eliminate (eng. deletions), I este numărul de cuvinte inserate, iar C reprezintă cuvintele clasificate corect. Astfel, acest parametru poate funcționa ca metrică pentru a descrie comportamentul unui sistem ce recunoaște cuvinte dintr-o frază. Vom folosi WER pentru a arăta rezultatele obținute de întregul sistem, datorită faptului că poate caracteriza atât acuratețea cu care se recunoaște fiecare cuvânt, cât și capacitatea sistemului de a identifica fiecare cuvânt dintr-o frază rostită.

3.3 Analiza rețelelor neuronale adânci

Așa cum aminteam în subcapitolul 2.7, este necesară realizarea unui set de experimente folosind fiecare structură DNN propusă, în vederea selecției celei mai potrivite dintre acestea. Vom analiza structurile standard prezentate, urmând a opera modificări doar asupra DNN-ului cu cele mai bune rezultate. Dată fiind dimensiunea redusă a CAMPUS – α LIRO, vom folosi GRID restrâns pentru analiza comparativă pe care o vom face în acest subcapitol. Ne propunem să realizăm o antrenare de tip 9:1, ceea ce înseamnă că 90% din setul de date va fi folosit pentru antrenare, iar restul de 10% va fi utilizat pentru validare. Acest lucru se traduce prin faptul că 270 de eşantioane per cuvânt vor fi folosite pentru antrenare și 30 pentru testare. Vorbitorii ale căror înregistrări intră în fiecare dintre cele două categorii au fost aleși aleator, astfel încât să obținem o generalizare cât mai bună.

Primul parametru analizat este reprezentat de dimensiunea loturilor (eng. *batch size*) în care este împărțit setul de antrenare. Astfel, dacă avem un set cu 510 eşantioane, setarea unui *batch size* de 100 înseamnă că rețeaua va fi antrenată întâi cu primele 100 de eşantioane din set, apoi cu următoarele 100 ș.a.m.d. Dificultatea apare în cazul ultimelor 10 *sample*-uri, care însă pot fi preluate și utilizate pentru antrenare făcând abstracție de absența celorlalte 90. Avantajul utilizării unui *batch size* de dimensiuni reduse este dat de consumul scăzut de resurse, memoria disponibilă funcționând adesea ca o limitare a antrenării. În plus, un *batch size* mic va conduce la un timp scăzut de antrenare, deoarece ponderile sunt constant actualizate. Pe de altă parte, este important de menționat că odată cu scăderea lotului crește variația gradientului, acesta atingând greu convergența. Vom analiza cele 4

rețele propuse din perspectiva variației parametrului *batch size*, pentru un număr fix de 3 cadre per cuvânt, antrenarea făcându-se timp de 100 de epoci, dar fiind oprită odată cu stagnarea scăderii lui *val_loss*. Rezultatele sunt prezentate în Figura 3.2.

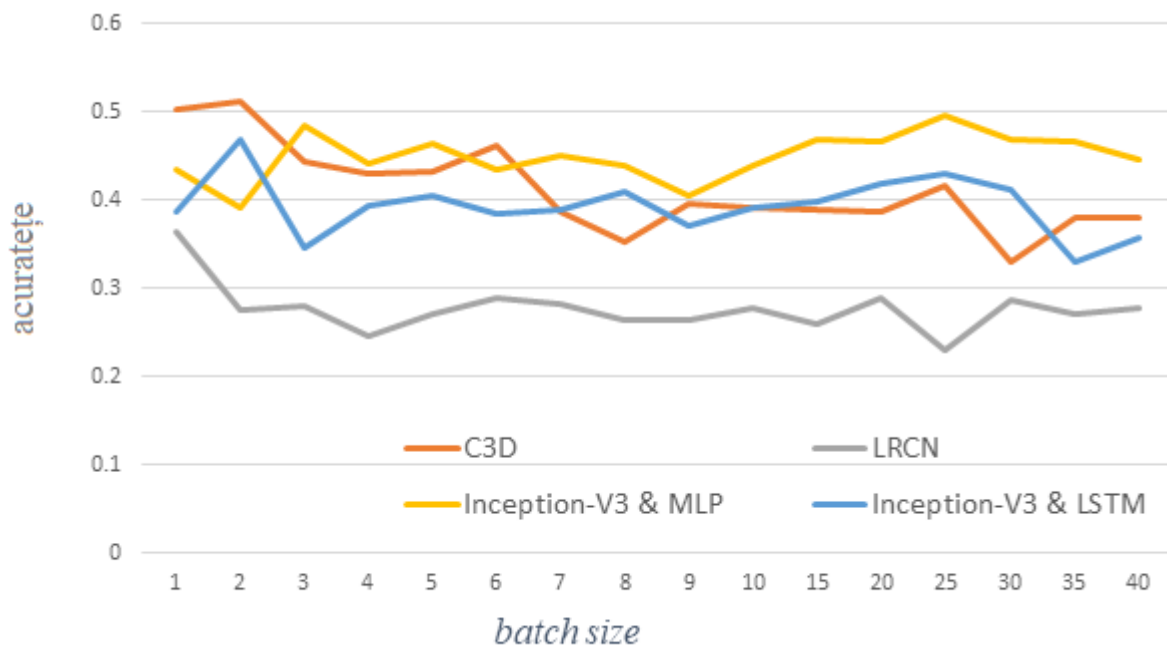
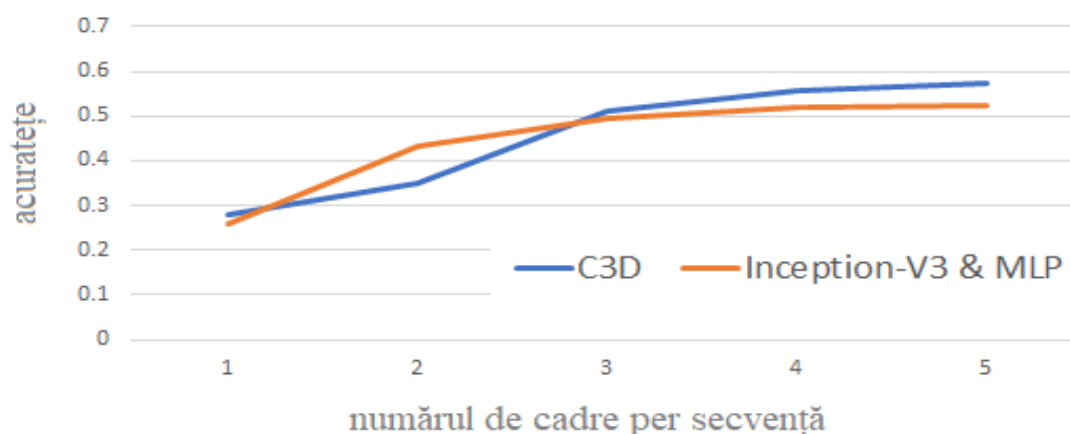


Figura 3.2: Variația acurateței în funcție de batch size

Se observă în figură faptul LRCN obține, în general, rezultate mai slabe decât orice altă rețea. De asemenea, se poate observa că Inception-V3 & LSTM, deși mai bună decât LRCN în condițiile date, nu are niciodată cea mai mare acuratețe, oricare ar fi dimensiunea lotului. Deoarece diferențele dintre punctele de maxim ale C3D și Inception-V3 & MLP sunt infime (51% pentru C3D la *batch size* = 2, respectiv 49.6% pentru Inception-V3 & MLP la *batch size* = 15) și deoarece există o diferență semnificativă între dimensiunile loturilor în cele două situații de maxim, suntem nevoiți să realizăm un alt experiment pentru a determina *batch size* și pentru a selecta o singură rețea pentru antrenarea sistemului.

Așa cum am arătat în subcapitolul 2.7, un alt parametru important al antrenării unor structuri ce țin cont de relația temporală dintre *sample*-uri este reprezentat de lungimea secvențelor. Altfel spus, este important să cunoaștem numărul optim de cadre per cuvânt care conduce la cele mai bune rezultate. Având în vedere informațiile relevate de Figura 3.2, vom elimina din lista de posibili candidați rețelele LRCN și Inception-V3 & LSTM. Pentru a putea face o alegere eficientă între celelalte două DNN-uri, vom reprezenta grafic dependența acurateței de numărul de cadre al unei secvențe. În mod particular, vom reprezenta doar valorile maxime obținute, indiferent de dimensiunea lotului folosit la antrenare. Rezultatele obținute sunt reprezentate în Figura 3.3. Se poate observa că, odată cu creșterea numărului de cadre ce constituie o secvență, rezultatele obținute de C3D sunt mai bune decât cele ale rețelei Inception-V3 & MLP. Este necesar deci să interpretăm semnificația numărului de cadre per secvență. În fapt, acest lucru reprezintă numărul de *frame*-uri pe care le

considerăm necesare și suficiente pentru a caracteriza un cuvânt. Este așadar absurd să setăm acest parametru la valoarea 1, deoarece ar însemna că un cuvânt poate fi complet descris de un singur cadru.



Figură 3.3: Variația acurateței în funcție de lungimea secvenței

Dat fiind faptul că performanțele C3D le depășesc pe cele ale Inception-V3 & MLP odată cu creșterea lungimii secvenței, luăm decizia de a integra în structura sistemului nostru rețeaua neuronală adâncă C3D. Decizia de a utiliza C3D în sistemul prezentat nu este însă suficientă, fiind necesară stabilirea cu exactitate a lungimii secvențelor ce vor fi transmise rețelei spre antrenare. Dat fiind faptul că GRID redus este o bază de date mai vastă decât CAMPUS – α LIRO, vom continua să o utilizăm în vederea stabilirii parametrilor de lucru. Variația acurateței în funcție de numărul de cadre per secvență poate fi observată în Tabelul 3.2, în care reprezentăm doar valorile maxime obținute pentru fiecare valoare a dimensiunii secvenței.

Tabel 3.2: Relația dintre dimensiunea secvențelor folosite la antrenare, acuratețea recunoașterii și setul de date. Se poate observa cum creșterea lungimii secvenței conduce la scăderea datelor disponibile. În tot acest proces, acuratețea are o variație neliniară

<i>Lungimea secvenței</i>	<i>Acuratețea maximă</i>	<i>Dimensiunea setului de antrenare</i>	<i>Dimensiunea setului de validare</i>
1	0.28	4050	448
2	0.35	4050	448
3	0.51	4050	448
4	0.55	4050	448
5	0.57	4047	448
6	0.56	4009	446
7	0.49	3799	423
8	0.52	3408	386
9	0.55	2820	344
10	0.58	2199	303
11	0.7	1636	246
12	0.72	1157	176

Faptul că aceste valori au fost obținute majoritar pentru un *batch size* de dimensiune 1 ne permite să tragem o concluzie suplimentară: setul de date de antrenare va fi analizat independent din punctul de vedere al modului în care ponderile se vor ajusta.

Se observă că există o tendință de creștere a acurateței odată cu creșterea lungimii secvenței folosite la antrenare. Acest aspect trebuie însă corelat cu dimensiunea setului de antrenare, respectiv cu cea a setului de validare. Astfel, în Tabelul 3.2 am menționat numărul total de cadre ce participă la acest proces. Se observă că, odată cu creșterea dimensiunii unei secvențe, numărul de cadre disponibile scade. Această situație era de așteptat, deoarece este influențată de lungimea cuvintelor prezente în GRID. Cuvintele scurte, formate dintr-un număr redus de *frame*-uri, sunt eliminate din procesul de antrenare și validare din cauza măririi secvenței folosite pentru reprezentarea unui cuvânt. Este așadar necesar să stabilim o lungime a secvenței de 5 cadre, acela fiind punctul în care se atinge acuratețea maximă, fără ca setul de antrenare și cel de validare să își reducă semnificativ dimensiunea.

Un ultim aspect ce trebuie luat în considerare este reprezentat de dimensiunea fiecărui cadru ce participă la antrenarea rețelei C3D. Numărul de pixeli va influența direct rezultatele rețelei, aceștia fiind cei care determină trăsăturile obținute de structura convoluțională. Variația dintre dimensiunea imaginii și rezultatele obținute sunt prezentate în Tabelul 3.3. Pentru aceste experimente am fixat parametrii conform concluziilor deduse anterior. Din tabelul de mai jos reiese că dimensiunea optimă pentru a obține o acuratețe maximă este de 80 pixeli x 80 pixeli, o matrice pătratică așadar.

Tabel 3.3: Variația acurateței în funcție de dimensiunea imaginii

<i>Dimensiunea imaginii (pixeli x pixeli)</i>	50 x 50	80 x 80	90 x 50	90 x 90	100 x 100
<i>Acuratețea obținută</i>	0.5	0.57	0.42	0.5	0.44

Observăm, așadar, că cele mai bune rezultate au fost obținute prin utilizarea rețelei neuronale adânci C3D, fără segmentare pe loturi, cu o lungime a secvenței ce reprezintă un cuvânt de 5 cadre, fiecare cadru având 80x80 de pixeli. Aceste rezultate înseamnă, în fapt, o acuratețe de 57.1% pe GRID redus. Ținând cont de faptul că o clasificare aleatoare a cuvintelor are probabilitatea de clasificare corectă de 6.66%, constatăm că rețeaua implementată are rezultate de 9 ori mai bune. Dat fiind faptul că întreg procesul de antrenare nu este dependent de structura lingvistică a limbii folosite, ci doar de cuvintele ce fac parte din lotul de antrenare, ne așteptăm ca, pentru un set de date similar alcătuit din cuvinte în limba română, să obținem aceleași rezultate. Experimental, constatăm că structura similară aplicată setului CAMPUS – αLIRO conduce la rezultate sensibil mai slabe, și anume 51.8%. Ne așteptăm însă ca această valoare să reprezinte, în fapt, o situație de supra-antrenare (eng. *overfitting*) și să nu reflecte capacitățile reale ale sistemului. Acest lucru este dat de dimensiunea redusă a setului de date și se va putea observa în subcapitolul următor.

3.4 Analiza metodelor de identificare a cuvintelor

Așa cum se poate observa în Figura 2.1, este necesară parcurgerea unui întreg proces înainte ca setul de cuvinte înregistrate să poată fi clasificate de rețea. Am prezentat în Capitolul 2 blocurile componente ale organigramei sistemului descris în această lucrare și am menționat pentru identificarea cuvântului propunem mai multe metode. În acest subcapitol urmează să analizăm rezultatele fiecăreia dintre aceste metode, urmând a o selecta pe cea optimă pentru a face parte din sistemul final.

Metoda conturului a avut rezultate mult sub așteptări, dat fiind comportamentul aleator al acesteia, și anume acela de a identifica un număr mai mare de contururi decât cel dorit, cel al gurii. În plus, nici variația parametrilor funcțiilor nu a condus la rezultate mai bune, deoarece setul de parametrii adaptați unui vorbitor nu puteau fi generalizați, astfel încât am fost nevoiți să renunțăm la această abordare. Cea de-a doua metodă, cea a histogramei, a avut comportamentul dorit doar dacă se întrunea un set de precondiții. Astfel, înregistrări cu iluminare difuză sau captarea unor subiecți cu păr facial (v. Figura 3.4) modifica puternic media histogramei. Astfel, se ajunge la situații în care gura întredeschisă a subiectului să fie tradusă ca reprezentând apariția unui cuvânt. Deoarece variația mediei histogramei în funcție de gradul de deschidere a gurii nu poate fi aproximată cu o funcție cunoscută, această abordare a fost abandonată. Cele mai bune rezultate au fost obținute prin utilizarea metodei ariei, deoarece există o dependență directă între gradul de deschidere a gurii și aria elipsei. Cadrele obținute prin intermediul acestei metode sunt prezentate în Figura 3.5, gradul lor de corectitudine determinându-ne să utilizăm metoda ariei în sistemul final.

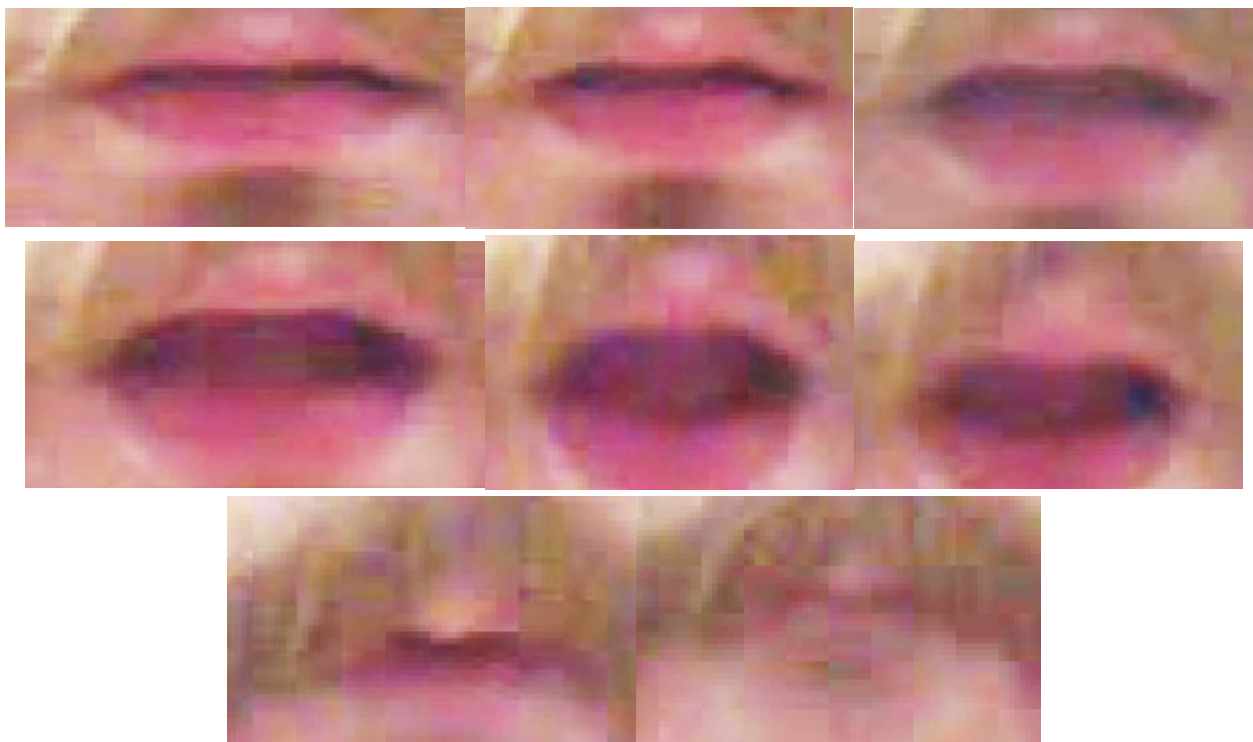
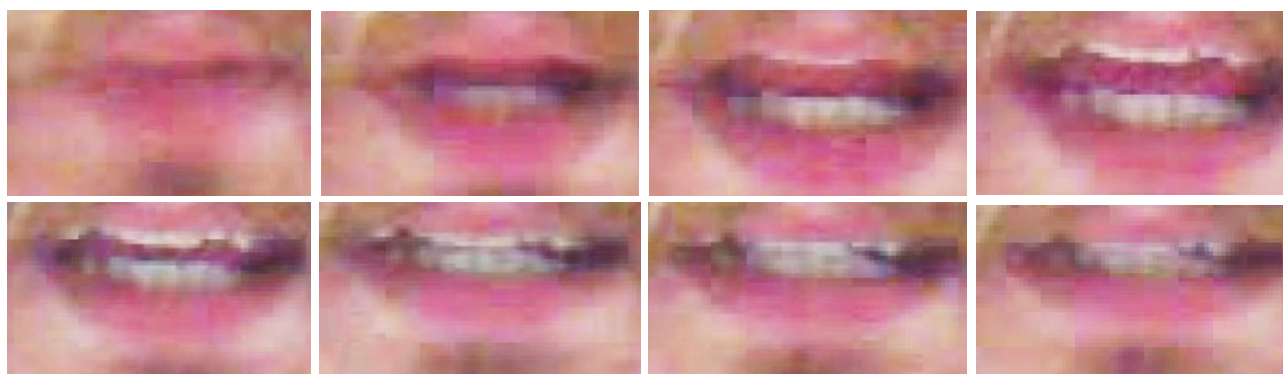


Figura 3.4: Rezultatele utilizării metodei histogramei. Gura întredeschisă a subiectului a fost considerată eronat ca reprezentând un cuvânt



Figură 3.5: Rezultatele utilizării metodei ariei. Cuvântul a fost identificat cu acuratețe deoarece este precedat de o modificare a ariei gurii, o închidere în cazul de față

3.5 Analiza metodelor de identificare a regiunii de interes

Determinarea zonei din imagine în care se află gura vorbitorului este o etapă ce influențează direct atât modul în care rețeaua va fi antrenată, cât și rezultatele oferite de sistem în timpul funcționării. De aceea, ne dorim ca această componentă a sistemului să ofere rezultate cât mai bune. Prin rezultate bune înțelegem ca numărul de cadre care conțin exclusiv buzele să fie egal cu numărul de cadre în care vorbitorul pronunță cuvinte. Acest lucru înseamnă absența rezultatelor de tip *false negative (fn)*, în care nu se identifică gura subiectului deși aceasta este prezentă în cadru, și de tip *false positive (fp)*, în care imaginea finală conține ale *frame*-ului decât cele de interes. Pentru a testa cele patru metode propuse în paragraful 2.3, am ales un număr de 356 de cadre din 30 de secvențe video, înregistrate de 6 vorbitori. În Tabelul 3.4 prezentăm numărul de cazuri *fp* și *fn* pentru fiecare metodă, cât și numărul de cadre utile, cadrele rămase ce conțin gura subiectului. Deoarece cele mai bune rezultate le-am obținut prin utilizarea metodei cu cadru fix, aceasta va fi cea implementată în sistemul final.

Tabel 3.4: Rezultatele metodelor de identificare a regiunii de interes. Obiectivul este alegerea metodei cu cele mai multe cadre finale ce conțin doar gura, determinate dintre cele 356 folosite pentru testare

Metoda testată	Metoda directă	Metoda indirectă	Metoda indirectă cu memorie	Metoda cu cadru fix
Cazuri de tip fn	28	0	0	0
Cazuri de tip fp	32	43	35	0
Cadre utile	296	313	321	356

3.6 Rezultate finale

Am arătat în subcapitolele anterioare care este structura sistemului final. Pentru a definitiva lucrarea, este necesară testarea sistemului automat de labiolectură implementat. Așa cum am

menționat, ne dorim obținerea unui sistem independent de limba vorbită de subiecți, diferența fiind făcută exclusiv de setul de date utilizat la antrenare. Cu toate acestea, ne dorim să implementăm un sistem funcțional pentru limba română. Dat fiind setul redus de date, obținerea unor rezultate concludente este dependentă de utilizarea unor reguli de limbă, definite în subcapitolul 2.5. În particular, vom implementa un algoritm care nu permite ca două cuvinte consecutive să fie identice, pentru a avea astfel rezultate similare cu limba vorbită. În plus, vom considera obligatorie prezența unui verb în propoziție. În Figura 3.6 se poate observa o captură de ecran cu rezultatul final afișat utilizatorului.



Figura 3.6: Etichetarea video-ului inițial folosind rezultatele sistemului

Pentru a testa funcționalitatea sistemului ne vom folosi de WER. În plus, vom analiza dacă sistemul este într-adevăr utilizabil prin compararea lui cu rezultatele unor subiecți umani. Am realizat un studiu folosind 5 voluntari de sex masculin și feminin, cu vârste cuprinse între 20 și 23 de ani. Atât sistemului, cât și participanților la studiu, li s-au oferit 5 fraze spre interpretare. Rezultatele comparative pot fi observate în Tabelul 3.5, acolo unde am menționat și WER-ul mediu total pentru subiecții umani, respectiv pentru sistemul implementat.

Privind ratele de cuvinte eronate din Tabelul 3.5, se poate observa că sistemul oferă rezultate mai bune decât subiecții umani în majoritatea cazurilor, dar și că media erorii sistemului implementat

este mai mică decât cea a participanților la studiu. Acest lucru arată că sistemul este unul performant și că, la limită, el poate înlocui un om neexperimentat în domeniul labiolecturii.

Tabel 3.5: Rezultatele sistemului comparativ cu cele ale unor subiecți umani. Folosim ca metrică de evaluare WER-ul și ne dorim să determinăm valoarea acesteia pentru fiecare frază, cât și media ei pe întreg setul de fraze. Sistemul va fi cu atât mai bun cu cât WER-ul este mai mic

<i>Fraza înregistrată</i>	<i>WER mediu pentru subiecții umani</i>	<i>WER pentru sistemul automat de labiolectură</i>
Anul acesta procesez imagini.	0.55	0.25
Electronica este pasiunea mea.	0.6	0.6
Eu prezint licenta anul acesta.	0.6	0.75
Sunt student la electronica.	0.85	0.5
Procesez imagini pentru licenta.	0.45	0.25
<i>WER global</i>	0.61	0.47

Fiind prima aplicație de acest tip capabilă să recunoască cuvinte în limba română, sistemul automat de labiolectură pe care l-am implementat poate reprezenta un *baseline* pentru realizarea unor sisteme mai robuste și cu performanțe mai bune.

Concluzii și perspective

În această lucrare ne-am propus realizarea unui sistem automat de labiolectură cu arhitectură independentă de limbă, dar care să poată fi utilizat de către vorbitorii limbii române. Lucrarea a fost organizată în două etape: implementarea arhitecturii și implementarea funcționalităților pentru limba română. Pentru prima etapă, am realizat patru rețele convoluționale adânci, ale căror comportamente le-am testat prin antrenare la nivel de cuvânt, utilizând setul de date GRID. Deoarece ne-am propus ca sistemul să poată identifica cuvinte din înregistrări continue, am implementat elementele de preprocesare a datelor, astfel încât propozițiile de la intrarea sistemului să fie separate în cuvinte ce pot fi clasificate de rețeaua pre-antrenată. Toată această parte a fost realizată și testată folosind cuvinte și propoziții în limba engleză, datorită setului vast de date pe care l-am avut la dispoziție. Odată stabilită configurația finală a sistemului, a fost necesară verificarea funcționalității sale în cazul limbii române. Deoarece nu există un set de înregistrări video disponibil, am realizat baza de date CAMPUS – α LIRO, alcătuită din propoziții ce conțin 15 cuvinte unice pronunțate de 15 vorbitori. Acesta a reprezentat protocolul antrenare-testare final pentru sistemul automat de labiolectură implementat în cadrul acestei lucrări.

Am obținut o acuratețe pe setul de date GRID redus de 57.1% și 51.8% pe CAMPUS – α LIRO, cea mai bună performanță pe acest set de date până la momentul curent. În ambele situații, am depășit valoarea de 6.66%, reprezentată de probabilitatea de apariție a unui cuvânt dintr-unul din cele două seturi de date. În plus, am obținut un WER de 47% pe CAMPUS – α LIRO, mai mic decât media de 61% obținută de participanții la un studiu pe care l-am realizat.

Date fiind rezultatele obținute, menționăm că pentru îmbunătățirea acestui sistem sunt necesari o serie de pași. În primul rând, acuratețea obținută pe CAMPUS – α LIRO ne arată că este necesară extinderea acestui set de date. El a reprezentat o variantă *alfa*, care să stabilească regulile de bune practici în realizarea CAMPUS – LIRO. Astfel, este necesară menținerea unei distanțe constante față de subiecții înregistrați, mai mică decât cea folosită de noi, cât și menținerea camerei video la o înălțime fixă. De asemenea, recomandăm iluminarea artificială, constantă pe parcursul tuturor înregistrărilor. Nu în ultimul rând, sugerăm ca fiecare cuvânt din baza de date să apară de un număr egal de ori, cât și folosirea unor cuvinte de lungime constantă sau, în caz contrar, modificarea arhitecturii rețelei neuronale adânci folosită în prezent. Un alt aspect care trebuie îmbunătățit ne este

relevat de valoarea WER-ului. Astfel, deși arhitectura generală a sistemului este una robustă, blocul care realizează identificarea cuvintelor și, implicit, segmentarea propozițiilor, poate propaga erori în sistem. Este necesară implementarea unui algoritm capabil să segmenteze propozițiile dintr-o înregistrare continuă, naturală, algoritmul ariei aplicat în această lucrare având nevoie de scurte pauze între cuvinte pentru a le putea identifica cu acuratețe. Nu în ultimul rând, blocul care impune regulile de gramatică asupra secvenței de cuvinte identificate de rețea poate fi optimizat cu ajutorul unor reguli stricte, prin intermediul unui dicționar care să conțină toate combinațiile posibile, corecte gramatical, ale cuvintelor din setul de date. Implementarea acestor sugestii poate conduce la realizarea primului set de date de mari dimensiuni cu înregistrări în limba română, cât și la obținerea unui sistem automat de labiolectură pentru limba română cu performanțe ridicate, care să iasă din sfera academică și să devină un produs finit în industrie.

Bibliografie

- [1] George Mather. *Essentials of Sensation and Perception. Foundations of Psychology*. Taylor & Francis, pages: 73-90, 2014
- [2] Fry, D.B.: *Theoretical aspects of mechanical speech recognition*, Journal of the British Institution of Radio Engineers, 19, (4), p. 211-218, 1959
- [3] Eric David Petajan. *Automatic Lipreading to Enhance Speech Recognition (Speech Reading)*. Ph.D. Dissertation. University of Illinois at Urbana-Champaign, 1984
- [4] Chung, Joon Son et al. "*Lip Reading Sentences in the Wild.*" 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR): 3444-3453, 2017.
- [5] Neil Midgley. „*New technology catches Hitler off guard*”. <https://www.telegraph.co.uk/news/uknews/1534830/New-technology-catches-Hitler-off-guard.html>. Accesat pe 19-06-2018.
- [6] Alasdair Palmer. „*Lip reader saw Fraser's incriminating conversation*”. <https://www.telegraph.co.uk/news/uknews/1420816/Lip-reader-saw-Frasers-incriminating-conversations.html>. Accesat pe 19-06-2018.
- [7] McGurk H., MacDonald J. "*Hearing lips and seeing voices*". Nature. 264 (5588): 746–8, 1976.
- [8] Corniță Georgeta, „*Fonetica integrată*”, Umbria, 2001
- [9] Ron Kovahi; Foster Provost. "*Glossary of terms*". Machine Learning 30: 271–274. 1998
- [10] Maqableh, M. , Karajeh, H. and Masa'deh, R. "*Job Scheduling for Cloud Computing Using Neural Networks*". Communications and Network, 6, 191-200. 2014
- [11] Ovidiu Grigore. „*Note de curs*”. <http://ai.pub.ro/content/RNSF.htm>. Accesat pe 19-06-2018

- [12] Conner DiPaolo. “*Perceptron*”. <https://github.com/cdipaolo/goml/tree/master/perceptron>. Accesat pe 19-06-2018
- [13] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. “*Backpropagation applied to handwritten zip code recognition*”. *Neural Comput.*, 1(4):541–551. 1989
- [14] Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton. “*Imagenet classification with deep convolutional neural networks*”. *Advances in neural information processing systems.*, pages 1097–1105, 2012
- [15] Marius Ignătescu. “*Lobul occipital și cortexul vizual*”. <https://www.descopera.org/lobul-occipital-si-cortexul-vizual/>. Accesat pe 19-06-2018
- [16] K. Fukushima. “*Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position*”. *Biological Cybernetics*, 36:193–202, 1980
- [17] Peng M, Wang C, Chen T, Liu G. “*NIRFaceNet: A Convolutional Neural Network for Near-Infrared Face Identification*”. *Information*. 7(4):61. 2016
- [18] Documentația Tensorflow. <https://github.com/tensorflow/tensorflow>. Accesat la 19-06-2018
- [19] Documentația Keras. <https://keras.io>. Accesat la 19-06-2018
- [20] Documentația OpenCV. <https://docs.opencv.org>. Accesat la 19-06-2018
- [21] Zheng, G.L., Zhu, M. and Feng, L. “*Review of Lip-Reading Recognition*”. *Computational Intelligence and Design (ISCID)*, 2014 Seventh International Symposium. 1:293-298. 2014
- [22] Yannis M. Assael, Brendan Shillingford, Shimon Whiteson, and Nando de Freitas. “*LipNet: Sentence-level Lipreading*”. *CoRR abs/1611.01599*. 2016
- [23] J. S. Chung and A. Zisserman. “*Lip reading in the wild*”. In *Asian Conference on Computer Vision*. 2016a.
- [24] Documentația FFmpeg. <https://www.ffmpeg.org>. Accesat la 19-06-2018
- [25] Viola, P., Jones, M. “*Rapid Object Detection using a Boosted Cascade of Simple Features*”. In *the Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. 2001.
- [26] D. E. King. “*Max-margin object detection*”. *arXiv preprint arXiv:1502.00046*. 2015
- [27] A. A Khan, A. Aziz, M Dawood. “*Face recognition techniques (FRT) based on face ratio under controlled conditions*”, *IEEE International Symposium on Biometrics and Security Technologies*, pp. 1-6. 2008.

- [28] Matt Harvey. “*Five video classification methods*”. <https://github.com/harvitronix/five-video-classification-methods>. Accesat la 19-06-2018
- [29] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. “*Rethinking the Inception architecture for computer vision*”. arXiv preprint, 1512.00567, 2015
- [30] S. Hochreiter and J. Schmidhuber. “*Long short-term memory*”. Neural computation, 9(8):1735–1780, 1997
- [31] J. Donahue *et al.*, “*Long-Term Recurrent Convolutional Networks for Visual Recognition and Description*”. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(4):677-691, 2017.
- [32] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. “*Learning spatiotemporal features with 3d convolutional networks*”. arXiv preprint arXiv:1412.0767, 2015
- [33] “*The GRID audiovisual sentence corpus*”. <http://spandh.dcs.shef.ac.uk/gridcorpus/>. Accesat la 19-06-2018
- [34] M.P. Cooke, J. Barker, S.P. Cunningham, X. Shao. “*An audio-visual corpus for speech perception and automatic speech recognition*”. Journal of the Acoustical Society of America, 120: 2421-2424. 2006
- [35] B. Ionescu, I. Mironică. “*Conceptul de indexare automată după conținut în contextul datelor multimedia*”. MatrixRom. 2013

Anexa 1

Diplomă obținută la Sesiunea de Comunicări Științifice Studentești



Anexa 2

Implementarea identificării regiunii de interes

```
blurred_frame = cv2.blur(frame, (3,3))
grey_frame = cv2.cvtColor(blurred_frame, cv2.COLOR_BGR2GRAY)
edges = cv2.Canny(grey_frame, threshold, threshold * 3, 3)
contours, hierarchy = cv2.findContours(edges, mode = cv2.CV_RETR_TREE,
method = cv2.CV_CHAIN_APPROX_SIMPLE, offset = (0, 0))
```

Listarea 2.1: Cod Python pentru selectarea conturului gurii

```
blurred_frame = cv2.blur(frame, (3, 3))
grey_frame = cv2.cvtColor(blurred_frame, cv2.COLOR_BGR2GRAY)
x, y, _ = frame.shape
histogram = cv2.calcHist([grey_frame], [0], None, [256], [0, x*y])
mean = 0
for hist_iterator in range (0, len(histogram)):
    mean = mean + hist_iterator * histogram[hist_iterator]/(x*y)
```

Listarea 2.2: Cod Python pentru detecția valorii medii a pixelilor din regiunea de interes

```
detector = dlib.get_frontal_face_detector()
predictor =
dlib.shape_predictor("shape_predictor_68_face_landmarks.dat")
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
rects = detector(gray, 1)
for (i, rect) in enumerate(rects):
    shape = predictor(gray, rect)
    shape = face_utils.shape_to_np(shape)
    for (name, (i, j)) in face_utils.FACIAL_LANDMARKS_IDXS.items():
        k = 49
        if (name == 'mouth'):
            for (x, y) in shape[i:j]:
                if (k == 57):
                    mouth_height_y = y - mouth_height_y
                elif (k == 51):
                    mouth_height_y = y
                elif (k == 48):
                    mouth_width_x = x
                elif (k == 54):
                    mouth_width_x = x - mouth_width_x
                k = k + 1
mouth_area = math.pi * (mouth_height_y / 2) * (mouth_width_x / 2)
```

Listarea 2.3: Cod Python pentru detecția ariei gurii prin aproximarea cu o elipsă

