



**APLICAȚIE MOBILĂ PENTRU ANALIZA ÎN TIMP
REAL A POSTURII UTILIZATORILOR FOLOSIND
VIZIUNE ARTIFICIALĂ**

LUCRARE DE LICENȚĂ

Absolvent: **Antonia-Maria TARTĂ**

Coordonator **Sl. Dr. Ing. Vlad Cristian MICLEA**
științific:

2025

Cuprins

Capitolul 1 Introducere	1
1.1 Motivație și context	1
1.2 Structura lucrării	2
Capitolul 2 Obiectivele proiectului	3
2.1 Obiectivul principal	3
2.2 Obiectivele secundare	3
2.3 Cerințe funcționale	3
2.4 Cerinte non-funcționale	4
Capitolul 3 Studiu bibliografic	5
3.1 Aplicații pentru analiza posturii	5
3.2 Fundamente ergonomice pentru clasificarea posturii	6
3.2.1 Parametri fizici analizați	6
3.2.2 Postura neutră	7
3.3 Metode de detecție a punctelor cheie de pe corp	7
3.4 Metode de analiză a posturii	8
3.4.1 Metoda 1: Clasificarea posturală pe baza unghiurilor și a etichetării fuzzy	9
3.4.2 Metoda 2: Evaluare posturală automată cu selecție de scor ergonomic	10
3.4.3 Metoda 3: Analiza posturii corporale cu ajutorul MobileNetV2	12
Capitolul 4 Analiză și fundamentare teoretică	14
4.1 Fundamente de învățare automată	14
4.1.1 Tipuri de învățare automată	14
4.1.2 Învățarea profundă	15
4.1.3 Rețele neuronale convoluționale	16
4.1.4 Subînvățare, Supraînvățare și Generalizare	19
4.1.5 Optimizarea rețelelor neuronale	21
4.1.6 Metrici de evaluare a performanței	24
4.1.7 Modelele de clasificare utilizate	25
4.2 Setul de date	29
4.2.1 MediaPipe	30
4.2.2 Etichetarea setului de date	31
4.3 Diagrama cazurilor de utilizare (use-case)	34
4.4 Tehnologii folosite pentru dezvoltarea aplicației mobile	35
Capitolul 5 Proiectare de detaliu și implementare	36
5.1 Arhitectura conceptuală a aplicației	36
5.2 Modulele aplicației	37
5.2.1 Interfața utilizatorului	37
5.2.2 Modulul de achiziție a imaginilor	37
5.2.3 Modulul de preprocesare a imaginilor	38
5.2.4 Modulul de clasificare	38
5.2.5 Modulul de interpretare a rezultatului	39

5.3	Diagrama de clase	39
5.4	Modelul DenseNet121	40
5.5	Diagrama de secvență	42
5.6	Diagrama de flux	42
5.7	Preprocesarea datelor	43
Capitolul 6	Testare și validare	46
6.1	Validarea etichetării setului de date	46
6.2	Evaluarea performanței modelului	47
6.3	Testarea aplicației mobile	49
6.3.1	Testare pe emulatoare	49
6.3.2	Testare pe dispozitive reale	50
6.3.3	Testare în condiții reale de utilizare	50
Capitolul 7	Manual de instalare și utilizare	52
7.1	Instalarea aplicației pe un dispozitiv mobil	52
7.2	Utilizarea aplicației	53
7.2.1	Lansarea aplicației	53
7.2.2	Interfața aplicației	53
7.2.3	Fluxul aplicației	54
Capitolul 8	Concluzii	56
Bibliografie		57

Capitolul 1. Introducere

1.1. Motivație și context

În ultima perioadă, din ce în ce mai multe persoane petrec un număr ridicat de ore desfășurând activități statice, la birou (pentru muncă, studiu sau divertisment). Menținerea unei posturi incorecte pentru un număr ridicat de ore poate avea efecte negative asupra sănătății. Printre consecințele unei posturi incorecte se află durerile de spate și afectiunile coloanei vertebrale. Modul în care stăm așezăți ne poate afecta atât sistemul muscular, cât și cel osos, dar poate avea efecte negative și asupra stării noastre emoționale, din cauza disconfortului creat.

De-a lungul timpului, s-au dezvoltat soluții pentru monitorizarea și corectarea posturii. Din păcate, multe dintre acestea presupun costuri ridicate sau necesitatea utilizării unor dispozitive suplimentare precum senzori, camere sau chiar scaune ergonomice. În general, persoanele sunt reticente când sunt nevoie să apeleze la o soluție care presupune să ai mereu un echipament suplimentar la îndemâna. Totodată, prețul acestor sisteme face ca acestea să nu fie disponibile publicului larg.

În acest context se creionează nevoiea unor alternative accesibile, ușor de folosit și care să nu necesite dotări suplimentare. O soluție este reprezentată de utilizarea viziunii artificiale. Acest domeniu îmbină procesarea de imagini cu inteligența artificială. Sistemele care folosesc viziunea artificială pot interpreta ceea ce văd în același fel în care o fac și oamenii.

Viziunea artificială a devenit o componentă esențială în numeroase aplicații. Ea este utilizată pentru diferite sarcini, precum: detectie facială, diagnostic medical, conducere autonomă. Sistemele pot identifica obiecte, urmări mișcări, recunoaște expresii faciale și chiar evalua stări emoționale. Tehnologiile actuale oferă rezultate precise și vin în ajutorul utilizatorilor, reducând timpul necesar procesării manuale a informațiilor vizuale. În cele mai multe cazuri, aceste sisteme elimină subiectivitatea și erorile care pot apărea în evaluările realizate de oameni. În domeniul sănătății, viziunea artificială este tot mai des utilizată pentru evaluarea imaginilor radiologice, analiza posturii sau monitorizarea utilizatorilor în timpul efectuării exercițiilor fizice. Specialiștii pot consulta aceste rezultate generate automat când stabilesc un diagnostic.

Capacitățile tot mai avansate ale viziunii artificiale pot fi valorificate pe dispozitive mobile, fără să mai fie nevoie de echipamente dedicate. Dacă în trecut, astfel de aplicații necesitau hardware specializat, în prezent camerele telefoanelor mobile pot furniza imagini suficient de calitative pentru procesare în timp real. Utilizarea telefoanelor dotate cu cameră foto a devenit un lucru obișnuit, ceea ce favorizează dezvoltarea unor aplicații care monitorizează postura utilizatorilor. În acest fel, utilizatorii ar beneficia de o soluție care să îi ajute să își evaluateze postura.

Modelul de clasificare utilizat în cadrul aplicației se bazează pe rețele neuronale convoluționale, antrenate pentru a recunoaște tipare de postură din imagini. Un avantaj al acestei abordări este faptul că procesarea se face pe dispozitiv (local), fără să fie nevoie de conexiune la internet pentru comunicarea cu alte servere. Astfel, rezultatele sunt primite în timp real și confidențialitatea este asigurată. Utilizatorii beneficiază de

o soluție care să le analizeze postura, fără să fie nevoie să suporte costuri suplimentare. Prin feedback-ul oferit, aplicația îi determină pe utilizatori să conștientizeze obiceiurile pe care le au când stau la birou și îi încurajează să adopte o postură corectă din punct de vedere ergonomic. Această conștientizare poate ajuta la prevenirea problemelor asociate cu o postură incorectă și are un impact pozitiv asupra sănătății.

Îmbinarea dispozitivelor mobile cu viziunea artificială prezintă o soluție accesibilă pentru monitorizarea posturii. Această lucrare își propune să dezvolte un astfel de sistem și să analizeze performanțele acestuia.

1.2. Structura lucrării

Lucrarea este împărțită în opt capitole, fiecare având un rol specific în analiza subiectului abordat. Mai jos se regăsește o prezentare a acestora:

- **Capitolul 1. Introducere**

Prezintă contextul și motivația temei alese, creionează problema abordată;

- **Capitolul 2. Obiectivele proiectului**

Detaliază scopul general al lucrării, expune obiectivele urmărite și conturează atât cerințele funcționale, cât și cerințele non-funcționale ale sistemului;

- **Capitolul 3. Studiu bibliografic**

În acest capitol se analizează soluțiile similare existente, incluzând o comparație între acestea;

- **Capitolul 4. Analiză și fundamentare teoretică**

Descrie noțiunile esențiale despre învățarea automată și despre modelele utilizate. Tot în acest capitol se prezintă tehnologiile folosite și setul de date;

- **Capitolul 5. Proiectare de detaliu și implementare**

Detaliază modul în care a fost proiectată aplicația, incluzând diagrame care prezintă structura, dar și modul de funcționare;

- **Capitolul 6. Testare și validare**

Oferă o perspectivă asupra metodelor de testare utilizate, adăugând rezultatele și interpretarea acestora;

- **Capitolul 7. Manual de instalare și utilizare**

Ghidează utilizatorii pas cu pas pentru a putea instala și utiliza aplicația, prezintă inclusiv situații neprevăzute care pot apărea;

- **Capitolul 8. Concluzii**

Recapitulează rezultatele obținute și obiectivele îndeplinite, propune direcții de dezvoltare ulterioară.

Această lucrare își propune să ofere o soluție practică pentru o problemă tot mai des întâlnită în viața de zi cu zi: menținerea unei posturi corecte în timpul activităților desfășurate la birou. Prin folosirea viziunii artificiale și a unei aplicații mobile, proiectul îmbină tehnologia cu nevoile reale ale utilizatorilor.

Capitolul 2. Obiectivele proiectului

Proiectul își propune dezvoltarea unei aplicații mobile capabile să analizeze postura utilizatorilor în timp real, utilizând tehnici de viziune artificială. Scopul principal al aplicației este de a contribui la prevenirea afecțiunilor coloanei vertebrale cauzate de o postură incorrectă.

2.1. Obiectivul principal

Obiectivul principal al proiectului este realizarea unei aplicații mobile care utilizează viziunea artificială pentru a analiza postura utilizatorilor în cadrul activităților desfășurate la birou. Această aplicație trebuie să fie capabilă să analizeze imagini, să identifice postura utilizatorului și să ofere feedback în timp real, fără a fi necesară conexiunea la internet sau echipamente suplimentare.

2.2. Obiectivele secundare

Îndeplinirea obiectivului principal este condiționată de realizarea obiectivelor secundare. Acestea acoperă procesul de colectare a datelor, antrenarea modelului și integrarea acestuia într-o aplicație mobilă Android. Proiectul abordează următoarele obiective secundare:

- 1) Crearea unui set de date;
- 2) Etichetarea setului de date;
- 3) Antrenarea unui model de clasificare bazat pe rețele neuronale convoluționale;
- 4) Evaluarea performanței modelului;
- 5) Implementarea aplicației mobile;
- 6) Integrarea modelului în aplicația mobilă;
- 7) Testarea aplicației în condiții reale de utilizare.

Pentru ca aplicația să funcționeze corespunzător și să răspundă nevoilor utilizatorilor, este importantă stabilirea unor cerințe funcționale și non-funcționale care vor sta la baza dezvoltării acesteia.

2.3. Cerințe funcționale

Cerințele funcționale descriu ce face sistemul. Acestea definesc funcționalitățile pe care aplicația trebuie să le îndeplinească pentru a răspunde obiectivelor propuse.

- Aplicația trebuie să permită utilizatorului să selecteze din ce perspectivă este capturată imaginea pe care o va analiza (din față sau lateral);
- Aplicația trebuie să permită utilizatorului să achiziționeze o imagine cu ajutorul camerei dispozitivului;
- Aplicația trebuie să permită utilizatorului să selecteze o imagine din galerie;
- Sistemul trebuie să identifice postura din imaginea furnizată de utilizator;
- Aplicația trebuie să afișeze rezultatul printr-un mesaj;
- Dacă postura este incorrectă, aplicația trebuie să ofere o explicație (tot sub formă de mesaj);

- Aplicația trebuie să îi prezinte utilizatorului niște sfaturi legate de poziția corectă și să precizeze ce obiceiuri trebuie evitate;
- Aplicația trebuie să afișeze mesaje de eroare în cazul în care utilizatorul nu furnizează toate informațiile necesare.

2.4. Cerinte non-funcționale

Cerințele non-funcționale definesc calitatea sistemului. Acestea nu se referă la ce face aplicația, ci la condițiile în care funcționează și modul în care valorifică resursele disponibile.

- Timpul de procesare al unei imagini trebuie să fie redus, iar rezultatele să fie afișate aproximativ instant;
- Aplicația trebuie să funcționeze fără conexiune la internet;
- Aplicația trebuie să funcționeze chiar și fără acces la cameră;
- Aplicația trebuie să fie compatibilă cu majoritatea dispozitivelor care rulează sistemul Android;
- Aplicația trebuie să aibă o interfață simplă, care să permită utilizatorilor neavizați să își analizeze postura fără să întâmpine probleme;
- Datele trebuie procesate local. Trebuie să se respecte confidențialitatea datelor utilizatorului, imaginile nu trebuie trimise spre alte servere;
- Sistemul trebuie să fie ușor de instalat;
- Aplicația trebuie să fie optimizată astfel încât să nu consume multe resurse (de exemplu să nu consume multă memorie);
- Acuratețea modelului de clasificare trebuie să fie de minim 80%;
- Lansarea aplicației nu trebuie să depășească mai mult de câteva secunde.

Prin urmare, definirea clară a obiectivului principal și a obiectivelor secundare, stabilirea cerințelor funcționale și non-funcționale, conturează direcția de dezvoltare a aplicației propuse. Prin cerințele funcționale s-au detaliat principalele acțiuni pe care aplicația trebuie să le permită utilizatorilor, iar prin intermediul cerințelor non-funcționale s-au evidențiat standardele de calitate pe care aceasta trebuie să le respecte.

Capitolul 3. Studiu bibliografic

Studiul bibliografic este una dintre cele mai importante etape care trebuie parcuse pentru realizarea unui proiect. Abordările existente fie conferă un punct de pornire care ajută la elaborarea obiectivelor stabilite, fie ajută la descoperirea lacunelor și la dezvoltarea unor noi strategii de soluționare a problemelor. Acest capitol se concentrează atât pe prezentarea aplicațiilor software care sunt deja utilizate, cât și pe metodele existente folosite pentru detectia punctelor și pe algoritmii de clasificare discutați în alte articole științifice sau cărți. În articolele studiate se află și niște studii ergonomice care au clarificat ceea ce înseamnă o postură corectă și ce abateri sunt acceptate și considerate normale.

3.1. Aplicații pentru analiza posturii

La momentul actual, există o multitudine de aplicații software care sunt folosite pentru analiza și menținerea unei posturi corecte din punct de vedere ergonomic. Multe dintre acestea nu oferă feedback utilizatorilor, ci doar prezintă sfaturi generale pentru întreținerea coloanei vertebrale. Unele soluții, deși eficiente, nu pot fi utilizate de către oricine, deoarece necesită echipamente suplimentare, care sunt costisitoare. Problema ridicată de prețul echipamentelor limitează accesul publicului larg la o soluție care monitorizează sănătatea.

Aplicația *Upright Go* se folosește de un dispozitiv purtabil pentru a oferi feedback utilizatorilor. Acest dispozitiv se plasează pe spatele persoanei și oferă un răspuns haptic când se detectează o deviație de la postura neutră, de obicei, sub formă de impuls sau vibrație [1]. În mod similar, funcționează și *Lumo Lift*, dar senzorul se plasează pe haine, în zona pieptului [2]. În figurile 3.1 și 3.2 sunt prezentate dispozitivele purtabile care trebuie achiziționate pentru utilizarea aplicațiilor menționate anterior.



Figura 3.1: Dispozitivul Upright Go

Sursă:

<https://www.amazon.in/Adhesive-Upright-trainer-depending-adhesives/dp/B08DYY1X9C>
(Accesată în 19-02-2025)



Figura 3.2: Dispozitivul Lumo Lift

Sursă:

<https://www.thetestpit.com/2015/08/review-lumo-lift.html>
(Accesată în 19-02-2025)

Tabelul 3.1 evidențiază diferențele dintre aplicațiile existente și soluția propusă (SpinalX). Comparația surprinde doar aplicații mobile și se concentreză pe caracteristicile de bază, precum cost, latență, accesibilitate. Se observă că soluția propusă se remarcă atât prin faptul că e gratuită și oferă feedback, cât și prin timpul scăzut de procesare a datelor. Aceasta nu necesită echipamente suplimentare, iar singurul minus ar fi că este disponibilă doar pentru Android.

Tabela 3.1: Analiză comparativă a aplicațiilor existente

Soluție	Echipamente?	Folosesc AI?	Cross platform?	Latentă?	Feedback?	Gratuit?
Upright GO [1]	Dispozitiv purtabil	Nu	Nu, iOS	Scăzută	Vibrații	Nu
Lumo Lift [2]	Dispozitiv purtabil	Nu	Nu, iOS	Scăzută	Vibrații	Nu
Posture Screen [3]	Cameră telefon	Da	Da	Ridicată	Da	Nu
Posture Correction Exercises [4]	Telefon	Nu	Da	Scăzută	Nu	Da
NLMeasurer [5]	Cameră telefon	Da, PoseNet	Nu, Android	Medie	Da	Doar prototip
SpinalX (abordarea propusă)	Cameră telefon	Da, DenseNet	Nu, Android	Scăzută	Da	Da

3.2. Fundamente ergonomice pentru clasificarea posturii

Postura umană este evaluată prin intermediul unor repere ergonomice care stabilesc modul optim de aliniere a corpului. O poziție corectă nu implică doar respectarea unor reguli pentru menținerea corpului într-o stare neutră care nu exercită presiune asupra mușchilor și oaselor. Numeroase lucrări, printre care se numără articolul [6] și cartea [7], sugerează că sedentarismul sau adoptarea unei poziții care nu este schimbată timp de mai multe ore (nu este relevant faptul că aliniamentul corporal este corect), provoacă apariția suprasolicitării și disconfortului la nivelul sistemului musculo-scheletal. Literatura de specialitate definește criteriile ergonomice care stau la baza dezvoltării sistemelor de analiză posturală.

3.2.1. Parametri fizici analizați

Analiza posturii se efectuează ținând cont de relațiile stabilite între punctele detectate de pe corp. Aceste relații evaluatează unghiurile definite de puncte și simetria acestora față de axele anatomici principale. Parametrii folosiți sunt:

- Unghiul dintre trunchi și coapse: utilizat pentru a determina dacă trunchiul este înclinat în față sau prea relaxat, lăsat pe spate;
- Unghiul dintre sapte și gât sau cap: utilizat pentru a examina zona cervicală;
- Unghiul dintre coapse și gambe: utilizat pentru a investiga gradul de flexie al picioarelor;
- Alinierea umerilor: utilizată pentru a detecta asimetrii, mai exact, detectează prezența dezechilibrelor musculare;

- Poziția genunchilor: utilizată pentru a identifica asimetrii provocate de statul picior peste picior (statul cu picioarele încrucișate) [8].

3.2.2. Postura neutră

Postura neutră reprezintă alinierea corporală optimă din punct de vedere ergonomic, tensiunea exercitată asupra coloanei vertebrale și țesuturilor de susținere fiind minimă. Adoptarea acesteia reduce riscul de apariție a durerilor și a problemelor de sănătate corelate cu leziuni care au efecte de lungă durată. Conform ghidurilor creionate în cartea *An Ergonomics Guide to Computer Workstations* [8] și elaborate de *Occupational Safety and Health Administration* (OSHA) [9], o postură neutră, pentru momentul în care persoana stă așezată, trebuie să respecte următoarele indicații:

- Trunchiul trebuie să fie orientat vertical;
- Coapsele trebuie să formeze un unghi de aproximativ 90° cu trunchiul și să fie paralele cu solul;
- Gâtul și capul trebuie să fie aliniati cu trunchiul (să nu fie îndreptate în exces spre partea anterioară sau posterioară);
- Umerii trebuie să fie simetrici față de axa mediană a corpului;
- Genunchii îndoiti, trebuie să formeze un unghi drept;
- Tălpile ar trebui să fie sprijinite pe un suport sau pe podea.

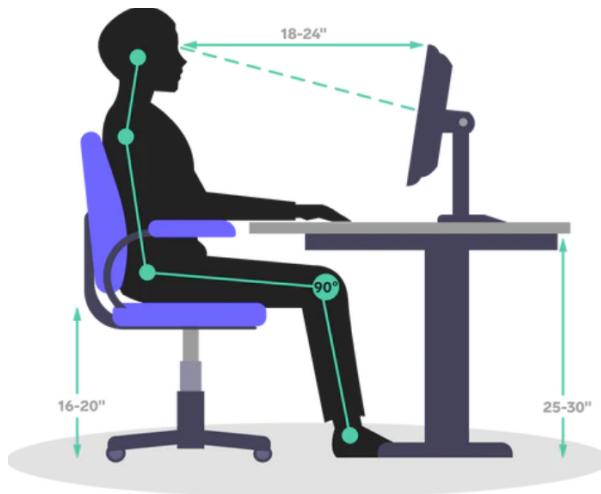


Figura 3.3: Postura neutră

Sursă: <https://ordernexstand.com/blogs/news/how-to-set-up-your-desk-ergonomically> (Accesată în 22-02-2025)

Figura 3.3 conturează un mod corect de a sta la birou. Orice deviere de la indicațiile anterior menționate poate să genereze dezechilibre musculare și poate să pună presiune pe coloana vertebrală. De obicei, cele mai des întâlnite repercușiuni ale unei posturi incorecte sunt: capul este proiectat anterior, spatele este cocoșat, umerii nu se mai află la același nivel [10, 11].

3.3. Metode de detecție a punctelor cheie de pe corp

Mijloacele automatizate de analiză posturală se bazează pe identificarea unor landmark-uri (puncte cheie) de pe corpul uman. Aceste puncte corespund unor zone anatomici, de exemplu: umeri, glezne, solduri, genunchi, urechi, nas, ochi. În aplicațiile moderne, a devenit esențială detectia automată a acestor puncte deoarece astfel se permite monitorizarea posturii în timp real. În trecut, pentru a efectua analiza posturii era

nevoie de senzori sau alte echipamente costisitoare care să detecteze punctele cheie. La momentul actual, pentru identificarea acestor puncte, există framework-uri și biblioteci open-source care se folosesc de rețele neuronale conveoluționale preantrenate pe diverse seturi de date. În soluțiile software sunt frecvent utilizate, datorită eficienței, următoarele:

- **PoseNet**

Este o bibliotecă potrivită atât pentru aplicațiile mobile, cât și pentru cele web. A fost dezvoltată de *Google* și detectează 17 puncte 2D din video sau imagini.

- **MediaPipe Pose**

Acest framework este dezvoltat tot de *Google* și identifică 33 de landmark-uri 3D. Este frecvent utilizat în aplicații unde este nevoie de un răspuns în timp real pentru că latența este scăzută.

- **BlazePose**

Este o extensie a modulului MediaPipe Pose. Acest framework a fost optimizat pentru a oferi rezultate mai precise chiar și în condiții slabe de iluminare. El detectează tot 33 de puncte și este potrivit pentru aplicații mobile.

- **OpenPose**

Această bibliotecă este dezvoltată de *Carnegie Mellon University* și are printre cele mai bune rezultate. Este destul de lentă, dar poate detecta între 25 și 135 de puncte de pe întreg corpul. Nu este optimă pentru mobile deoarece necesită resurse hardware de nivel ridicat.

Principalele diferențe dintre metodele de detecție a punctelor amintite anterior sunt prezентate în Tabelul 3.2. Această comparație conturează atât avantajele, cât și limitările fiecărei soluții.

Tabela 3.2: Analiză comparativă a mijloacelor de detecție a punctelor existente

Metodă	PoseNet [12]	MediaPipe Pose [13]	BlazePose [14]	OpenPose [15]
Număr puncte detectate	17	33	33	25-135
Dimensiune	2D	3D	3D	2D/3D
Acuratețe	Medie	Bună	Foarte bună	Foarte bună
Latență	Mică	Foarte mică	Mică	Ridicată
Suport mobile?	Da	Da	Da	Nu

3.4. Metode de analiză a posturii

Una dintre etapele esențiale în procesul de analiză posturală este clasificarea. În studiile de specialitate, se prezintă diverse abordări care variază în funcție de algoritmii de clasificare utilizați (de exemplu: machine learning, sisteme fuzzy), de tipul trăsăturilor extrase sau de tipul datelor prelucrate (imagini, videoclipuri, date provenite de la senzori). În această secțiune sunt creionate câteva metode, accentul fiind pus pe:

- tipul datelor de intrare;
- modul de etichetare;
- clasificatorii utilizați;
- performanțele obținute.

3.4.1. Metoda 1: Clasificarea posturală pe baza unghiurilor și a etichetării fuzzy

În lucrarea sa [16], Coskun propune o metodă de clasificare automată a posturilor în timpul statului pe scaun, prin utilizarea unghiurilor dintre punctele cheie detectate de pe corp și un sistem de etichetare bazat pe inferență fuzzy. Această abordare evită problemele asociate cu etichetarea manuală a datelor, deoarece totul se face într-un mod obiectiv, construit pe baza criteriilor stabilite de experții în ergonomie și reduce timpul de lucru.

Datele sunt preluate de la o cameră Kinect care oferă coordonatele tridimensionale pentru un set de 25 de puncte de pe schelet. Participanții acestui studiu au menținut timp de 30 de secunde câte o postură indicată pe un ecran. Fiecare postură este ilustrată în Figura 3.4 și este definită de una din etichetele:

- Postura standard – spatele sprijinit complet, mâinile pe cotiere, genunchii la 90°;
- Înclinat în față – trunchiul proiectat înainte, fără contact cu spătarul;
- Înclinat spre stânga – greutatea mutată pe partea stângă, cu picioarele încrucișate;
- Înclinat spre dreapta – similar cu poziția anteroiară, dar pe partea dreaptă;
- Înclinat pe spate – alunecare în scaun, cu reducerea contactului în zona lombară.

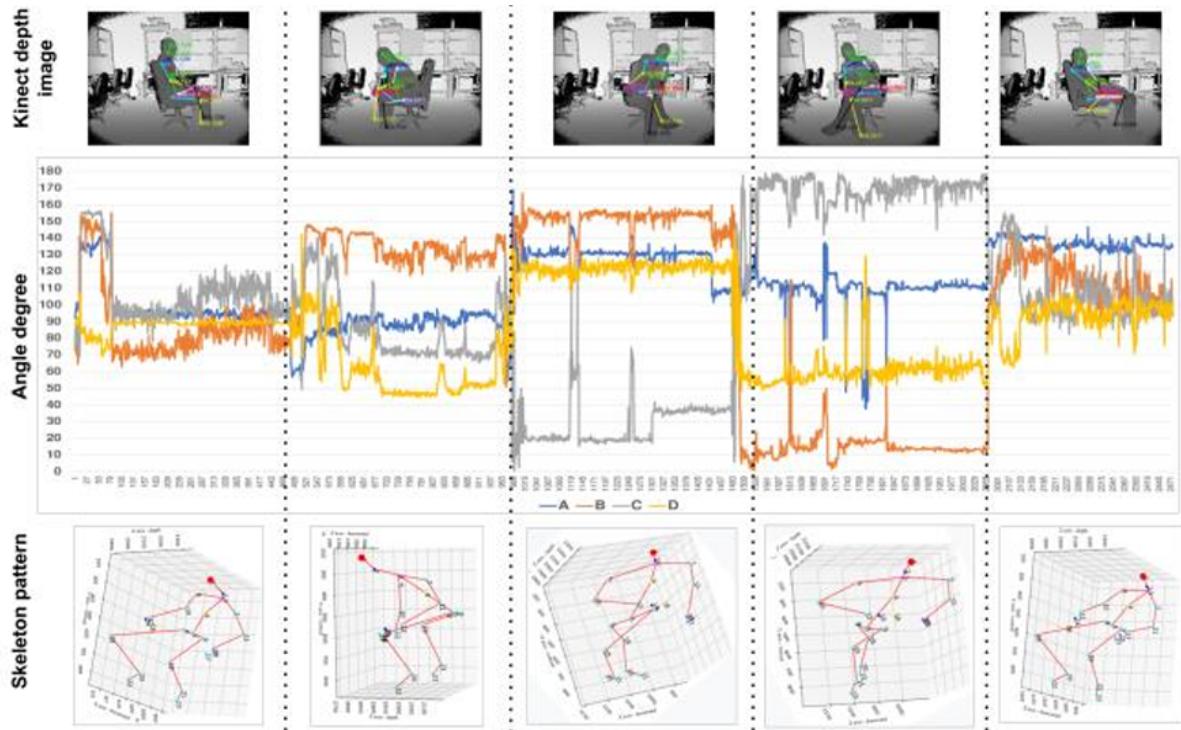


Figura 3.4: Datele Kinect corespunzătoare celor cinci posturi (Sursa [7])

Cu ajutorul punctelor determinate, autorul a calculat niște unghiuri între segmente corporale (unghi trunchi-cap, unghi trunchi-coapse, etc.). Fiecare unghi a fost asociat cu o postură. Pentru calcularea acestora, a fost nevoie de două etape. În prima etapă, s-a folosit Distanța Euclidiană (ecuația 3.1) pentru a calcula distanța dintre două puncte.

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2} \quad (3.1)$$

Această distanță a fost calculată pentru a putea aplica ulterior Teorema Cosinusului, prin care se determină măsura unghiului determinat de două segmente. Înmulțirea cu $\frac{180}{\pi}$

transformă rezultatul în grade din radiani (ecuația 3.2).

$$A^\circ = 180 - \arccos \left(\frac{|AB|^2 + |AC|^2 - |BC|^2}{2 \cdot |AB| \cdot |AC|} \right) \cdot \frac{180}{\pi} \quad (3.2)$$

A fost folosit un sistem fuzzy bazat pe funcții de apartenență Gaussiane pentru a eticheta instanțele. Ieșirea sistemului a fost o valoare între 0 și 5 care a fost ulterior mapată la una dintre clasele corespunzătoare celor cinci tipuri de postură. Clasificarea automată a fost realizată prin două metode: algoritmul K-Nearest Neighbors(KNN) și o rețea neuronală. Aproximativ 15% din date au fost utilizate pentru testare. Rețeaua neuronală a obținut o acuratețe de peste 97%, iar KNN a atins o precizie de aproape 99%. Aceste rezultate au confirmat relevanța unghiurilor folosite și eficiența etichetării. Diagrama întregului sistem este expusă în Figura 3.5.

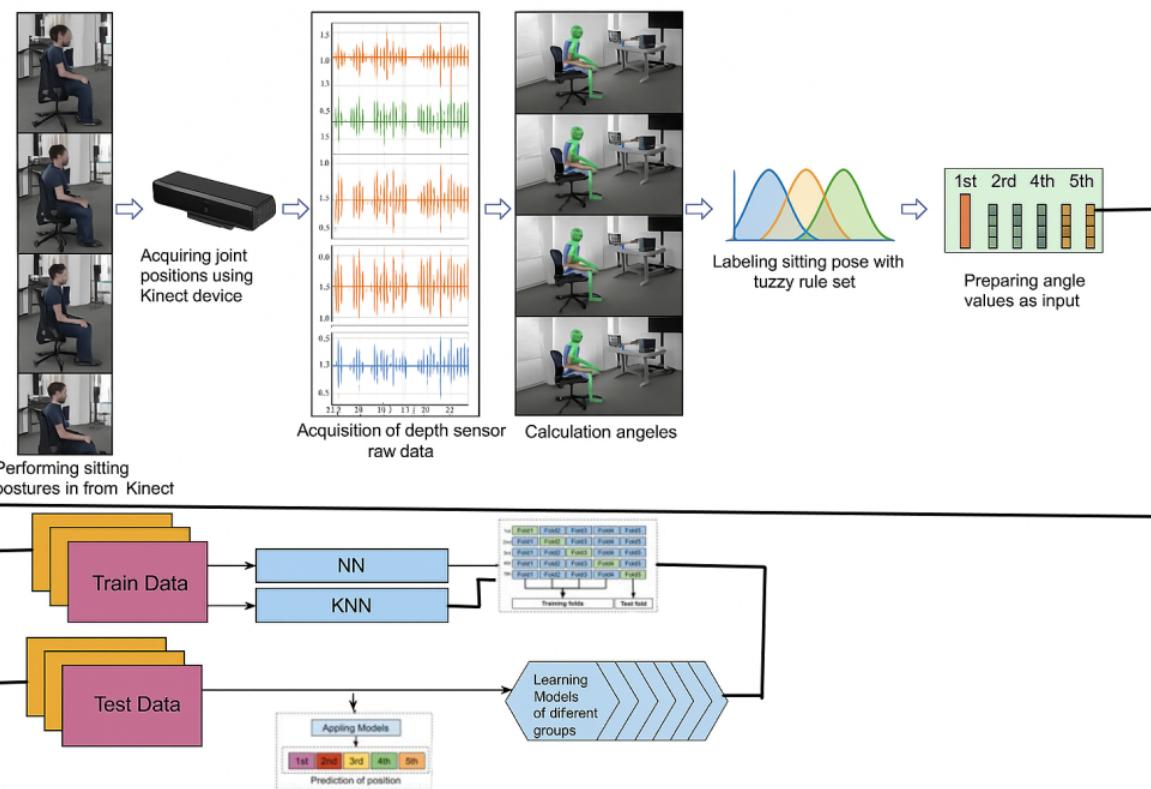


Figura 3.5: Diagrama de etichetare și clasificare (Sursa [7])

3.4.2. Metoda 2: Evaluare posturală automată cu selecție de scor ergonomic

Această metodă propune analiza posturală fără să utilizeze algoritmi de inteligență artificială. Autorii se bazează doar pe reguli ergonomicice și pe un arbore de decizie care selectează cea mai potrivită scală de evaluare a solicitării posturale. Procedura este sintetizată în Figura 3.6. Etapele conturate în figură definesc o secvență logică de procesare a datelor achiziționate prin intermediul OpenPose, urmată de calculul unghiurilor relevante. Ulterior, sistemul clasifică activitatea în funcție de tip, durată și efort.

Trăsăturile au fost extrase prin intermediul bibliotecii OpenPose. Datele de intrare utilizate au fost cincisprezece videoclipuri în care oamenii erau surprinși făcând diverse activități: lucru la birou, sport, conducere. Pe baza punctelor cheie extrase (gât, solduri, ge-

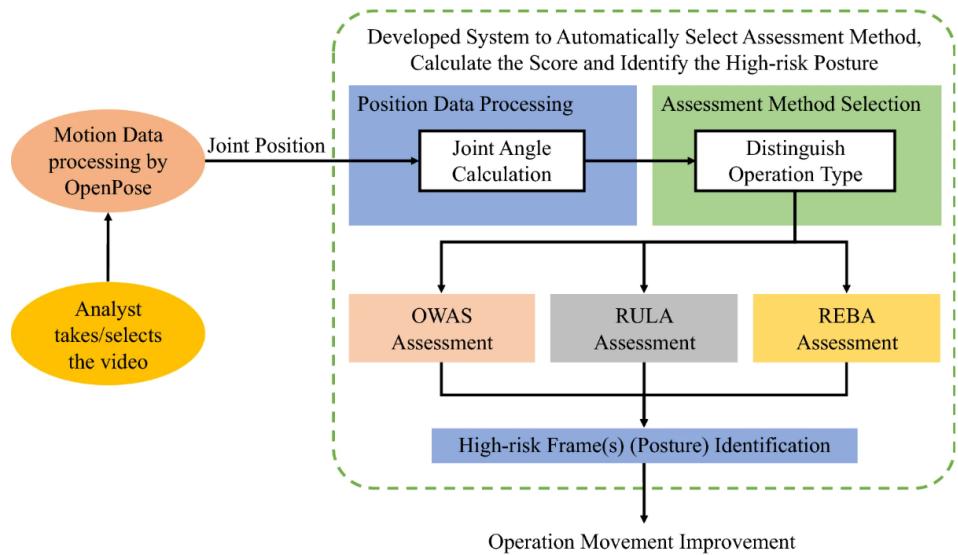


Figura 3.6: Schema generală a sistemului (Sursa [17])

nunchi, coate) se calculează niște unghiuri între zone ale corpului: unghiul dintre trunchi și coapsă pentru evaluarea înclinării spatelui, unghiul dintre trunchi și braț, unghiul dintre trunchi și gât pentru monitorizarea zonei cervicale, etc. Aceste unghiuri au fost relevante pentru a clasifica activitatea ca fiind statică, dinamică sau de manipulare a obiectelor. Nicio metodă ergonomică nu este valabilă pentru absolut toate contextele, iar din acest motiv, scăla este aleasă automat prin utilizarea unui arbore de decizii [17]. În acest fel, fiecare cadru din videoclip beneficiază de criteriile potrivite de evaluare. Acest arbore este ilustrat în Figura 3.7. El ia în considerare următorii indici: natura activității (statică sau dinamică), durata și articulațiile implicate. În acest fel, analiza este personalizată în funcție de context și de activitatea desfășurată.

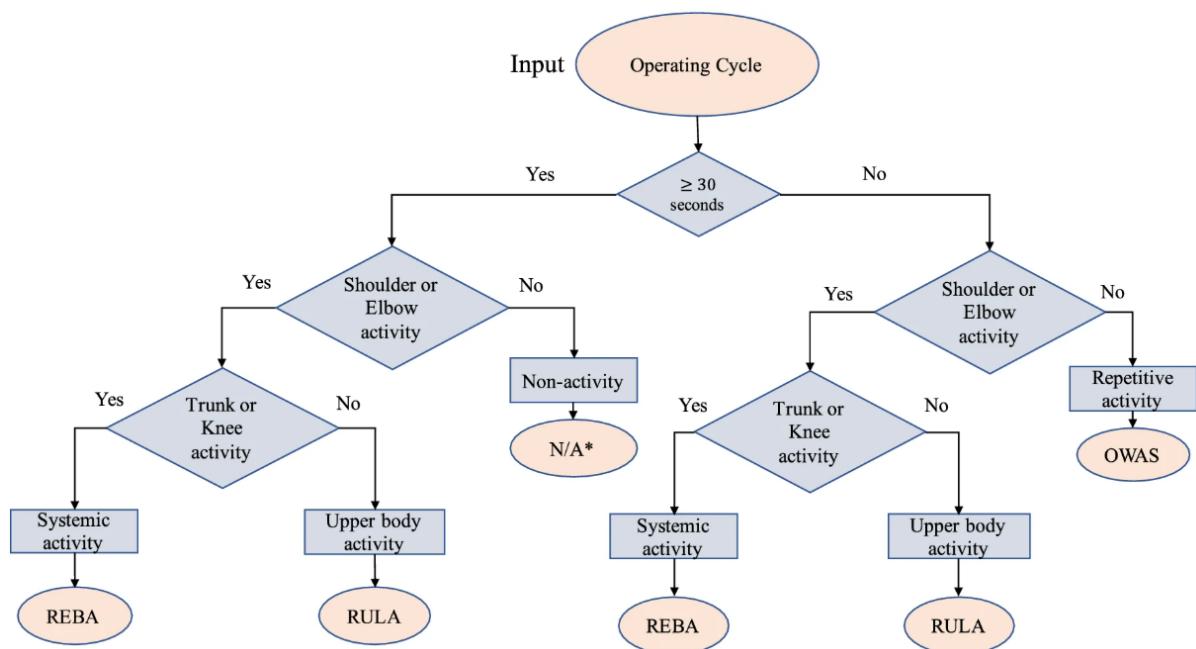


Figura 3.7: Arborele decizional utilizat pentru selectarea scalei ergonomice (Sursa [17])

Cele trei scale utilizate de autori sunt:

- **Ovako Working Posture Analysis System (OWAS)** - este o metodă care se concentrează pe alinierea spotelui, poziția membrelor și efortul fizic depus. Rezultatul evidențiază cât de urgentă este necesitatea unei corecții ergonomice și este reprezentat de un scor de risc între 1 și 4 [18];
- **Rapid Entire Body Assessment (REBA)** - examinează întregul corp în contexte dinamice. Scorul de risc este între 1 și 15. Aceasta ia în considerare frecvența sarcinilor, poziția articulațiilor și forța depusă [19];
- **Rapid Upper Limb Assessment (RULA)** - este potrivită pentru sarcinile statice care antrenează, în mod special, partea superioară a corpului (trunchi, cap, brațe, gât). Analog, rezultatul este un scor de risc între 1 și 7, care indică nevoia corecții posturale [20].

Performanța sistemului a fost determinată prin compararea rezultatelor obținute cu expertizele oferite de specialiști în ergonomie. Aproape în toate cazurile, scorurile atribuite manual au fost identice cu cele generate automat. Astfel, această metodă se dovedește a fi eficientă în situații care necesită intervenții rapide și monitorizare continuă. Sistemul poate fi utilizat pentru a furniza utilizatorilor alerte cu privire la adoptarea unei posturi adecvate, în timp real.

3.4.3. Metoda 3: Analiza posturii corporale cu ajutorul MobileNetV2

În cadrul studiului realizat [21], autori abordează analiza posturii statice a corpului uman în timpul activităților efectuate stând la birou. Aceștia remarcă lipsa unui set de date potrivit deoarece aveau nevoie atât de informație de culoare RGB, cât și de adâncime. Cercetătorii au decis să își colecteze propriul set de date. Acest lucru a fost realizat cu ajutorul unei camere Intel care a capturat imagini și înregistrări.

Datele au fost adunate cu ajutorul a unsprezece persoane care au efectuat o gamă de posturi, stând la birou. Acestea au abordat atât posturi neutre, cât și posturi incorecte și s-au expus la condiții diferite de iluminat. Pentru a avea un set de date diversificat, autori au decis să poziționeze camera în locuri diferite, astfel încât imaginile folosite să aibă perspective variate. Setul de date a fost etichetat manual, astfel s-a introdus o doză de subiectivitate, iar autori au subliniat dificultatea de a clasifica unele posturi apropiate vizual. În Figura 3.8 se ilustrează gruparea claselor în trei categorii principale. Această restrângere a claselor a fost efectuată din cauza posturilor similare care nu puteau fi incluse cu precizie doar într-o singură clasă.

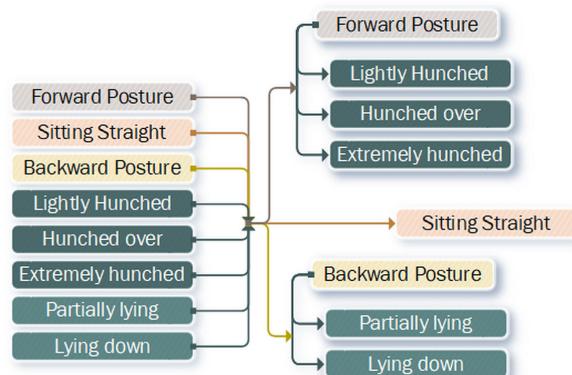


Figura 3.8: Arborele decizional utilizat pentru selectarea unei scale ergonomice (Sursa [21])

Etichetele utilizate pentru a oferi explicații mai detaliate cu privire la postură sunt:

- **Drept** - poziția corectă, neutră, spatele drept, fără asimetrii remarcabile față de axa mediană a corpului;
- **Ușor cocoșat** - o înclinație subtilă și involuntară a trunchiului spre față care proiectează capul ușor înainte;
- **Cocoșat** - trunchiul este curbat, iar gâtul și capul sunt împinse în față;
- **Foarte cocoșat** - capul este deplasat înainte, coloana se asemănă cu litera C;
- **Întins parțial pe scaun** - trunchiul este înclinat spre spate, dar nu total, iar zona lombară nu mai are susținerea adecvată;
- **Întins complet pe scaun** - capul este sprijinit de scaun, persoana este aproape întinsă.

Metoda propusă are ca bază un model cunoscut pentru eficiență să în aplicații cu resurse limitate, MobileNetV2. Pentru a examina variații ale poziției corpului, autorii analizează secvențe înregistrate la intervale scurte de timp. Clasificarea posturii a fost realizată în mod ierarhic: mai întâi s-au determinat categoriile generale (înclinat spre față, lăsat pe spate, postură dreaptă), apoi s-a trimis predicția către niveluri mai detaliate (de exemplu: foarte cocoșat sau întins parțial). Prin mecanismul acesta s-a redus riscul de confuzie între clasele apropriate vizual. Arhitectura rețelei propuse este ilustrată în Figura 3.9. Modelul primește la intrare secvențe de imagini pe care le procesează prin straturi convoluționale și recurente. Arhitectura permite extragerea trăsăturilor vizuale eficient, iar modulul cu memorie pe termen lung și scurt, *Long Short-Term Memory* (LSTM), adaugă dimensiunea temporală în analiză pentru a permite detectarea posturilor din cadrele unui video.

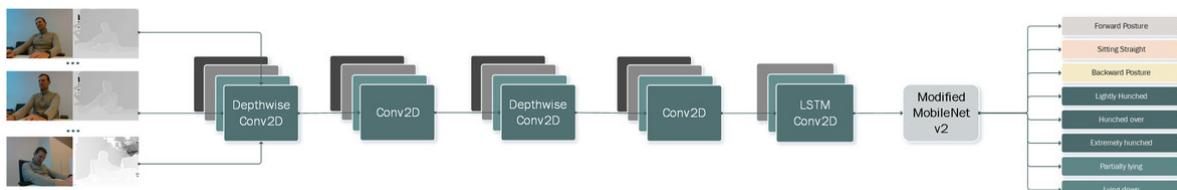


Figura 3.9: Arhitectura rețelei neuronale (Sursa [21])

Rezultatele obținute au fost validate în medii unde nu a fost mereu vizibil întreg corpul. Pentru clasele de bază (postură dreaptă, înclinare înainte, întins), acuratețea obținută este de 91.47%. În cazul clasificării detaliate, unde s-au folosit toate cele 6 clase, acuratețea scade la 68.33%. Deși metoda oferă o acuratețe ridicată pentru clasificarea posturii și funcționează eficient în medii reale, există câteva limitări. Modelul antrenat întâmpină dificultăți când încearcă să diferențieze între clasele „cocoșat” și „foarte cocoșat”. Aceste confuzii se reflectă și în performanța modelului deoarece diferențele subtile de înclinare sunt greu de evaluat. În plus, poziția camerei, iluminatul sau particularitățile anatomicice ale utilizatorilor pot introduce variații necontrolate în date. Astfel, modelul este sensibil la ambiguitățile de etichetare și la variațiile contextuale, ceea ce sugerează necesitatea unor date mai diversificate și a unor criterii mai clare de etichetare pentru îmbunătățirea generalizării. Modelul ar fi compatibil cu aplicații în timp real și ar fi util în sisteme de monitorizare a sănătății posturale pentru că este funcțional chiar dacă este vizibilă doar partea superioară a corpului.

Capitolul 4. Analiză și fundamentare teoretică

Acest capitol detaliază concepțele și fundamentele teoretice care stau la baza aplicației dezvoltate pentru analiza posturii utilizatorilor. Se vor prezenta aspectele esențiale ale procesului de învățare automată aplicat, algoritmii și arhitectura neuronală selectată, modul în care a fost construit setul de date, precum și tehnologiile folosite. Fiecare alegere privind tehnologiile și metodele utilizate va fi motivată prin descrierea avantajelor și limitărilor observate.

4.1. Fundamente de învățare automată

Învățarea automată (Machine Learning) este un subdomeniu al inteligenței artificiale care se ocupă cu dezvoltarea de algoritmi capabili să învețe din date și să ia decizii [22]. Regulile pe baza cărora se iau decizii nu sunt definite manual. Acestea sunt extrase automat din seturile de date.

Fluxul unei soluții de Machine Learning este prezentat în [23, 24] și include:

- conversia datelor brute în reprezentări numerice;
- alegerea modelului adecvat pentru obiectivul propus;
- optimizarea parametrilor;
- evaluarea performanței pe seturi dedicate de date (validare și test).

4.1.1. Tipuri de învățare automată

Învățarea supervizată (Supervised Learning) se bazează pe un set de date etichetate, în care fiecare exemplu este asociat cu o clasă. Scopul modelului este de a învăța relația dintre intrări și ieșiri, astfel încât să poată generaliza corect atunci când întâlnește date noi. Este potrivită pentru probleme de clasificare și regresie. Clasificarea atribuie datelor o clasă, iar regresia estimează o valoare pe care o va asocia cu datele [25].

Învățarea nesupervizată (Unsupervised Learning) e folosită în situațiile în care datele de intrare nu sunt etichetate. Scopul acesteia este de a descoperi relații între date sau tipare care să furnizeze răspunsurile corecte. O aplicație a învățării nesupervizate este segmentarea imaginilor [26].

Învățarea semi-supervizată (Semi-Supervised Learning) îmbină învățarea supervizată cu cea nesupervizată. Aceasta utilizează un număr restrâns de date etichetate, alături de un volum mai mare de date neetichetate. Abordarea este utilă atunci când etichetarea manuală este costisitoare și există o colecție amplă de date [27].

Învățarea prin consolidare (Reinforcement Learning) presupune existența unui agent care interacționează cu mediul și dezvoltă o strategie bazându-se pe un sistem de recompense și penalizări [28].

Diferențele dintre tipurile de învățare automată sunt surprinse de Figura 4.1. În partea stângă este reprezentată învățarea nesupervizată. Modelul grupează automat datele în clustere pe baza similarităților, fără a avea acces la etichete. În mijloc, se observă învățarea supervizată, unde punctele etichetate sunt separate de o limită de decizie. În dreapta, este ilustrată învățarea prin consolidare. Aceasta este creionată sub forma unui set de acțiuni și recompense.

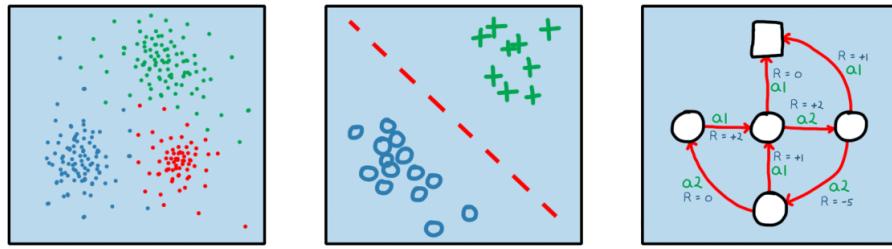


Figura 4.1: Tipuri de învățare automată
 Sursă: <https://www.mathworks.com/discovery/reinforcement-learning.html>
 (Accesată în 14-03-2025)

4.1.2. Învățarea profundă

Învățarea profundă (Deep Learning) este o ramură a învățării automate care se concentrează pe utilizarea rețelelor neuronale pentru a contura relațiile dintre date. Acest tip de învățare s-a remarcat deoarece nu necesită extragerea manuală a caracteristicilor.

Avantajul Deep Learning-ului de a învăța automat caracteristici exclude necesitatea intervenției umane. Modelele care utilizează învățare profundă gestionează eficient datele nestructurate (imagini, sunete, texte). Acestea sunt scalabile și adaptabile deoarece pot fi refolosite și ajustate pentru sarcini noi prin intermediul învățării prin transfer (transfer learning). O limitare a învățării profunde este volumul mare de date necesar. Metoda implică atât colectarea datelor, cât și etichetarea acestora. Un alt dezavantaj ar fi faptul că sunt necesare unități de procesare grafică (GPU) performante și memorie multă. Timpul de procesare este încă un minus al acestei metode [23].

Figura 4.2 prezintă o comparație între învățarea automată și învățarea profundă. În partea de sus, remarcăm cum la învățarea automată, caracteristicile (colțuri, muchii, texturi) sunt extrase manual, iar apoi introduse în modelul de clasificare. În partea de jos, se observă că la învățarea profundă, intervenția umană nu este necesară deoarece imaginiile sunt procesate direct, caracteristicile sunt extrase automat și clasificarea este integrată în același flux.

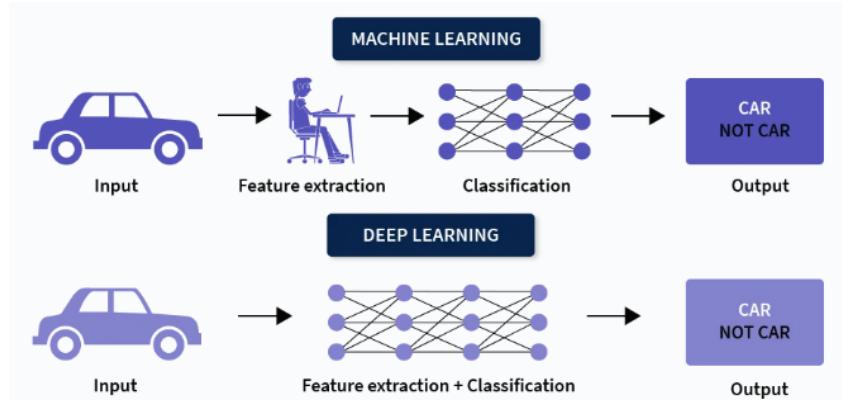


Figura 4.2: Diferența dintre învățarea automată și învățarea profundă
 Sursă: <https://www.scaler.com/topics/what-is-deep-learning/>
 (Accesată în 14-03-2025)

4.1.3. Rețele neuronale convoluționale

Rețele neuronale convoluționale sunt o clasă de modele de învățare profundă. Aceste rețele procesează și analizează în special imagini. Aceste rețele valorifică structura spațială a imaginilor prin intermediul operațiilor de convoluție.

Convoluția este o operație matematică folosită pentru a extrage trăsături locale. Ea presupune aplicarea unui filtru pe porțiuni din imagini. Filtrul se deplasează, parcurge întreaga imagine și generează o hartă de activare (feature map). Harta evidențiază trăsăturile identificate (colțurile, texturile). Filtrele se adaptează tipului de date folosit, nu sunt configurate în prealabil [29].

Arhitectura unei astfel de rețele este prezentată în Figura 4.3. Rețeaua prezentată este utilizată pentru clasificarea imaginilor. La intrare, se primește o imagine, aceasta este procesată cu ajutorul mai multor straturi. Fiecare strat are un rol diferit. Unele straturi se ocupă de extragerea trăsăturilor, iar altele de realizarea predicției. Aceste straturi sunt componente care contribuie la transformarea progresivă a datelor de intrare în abstractizări și vor fi detaliate ulterior.

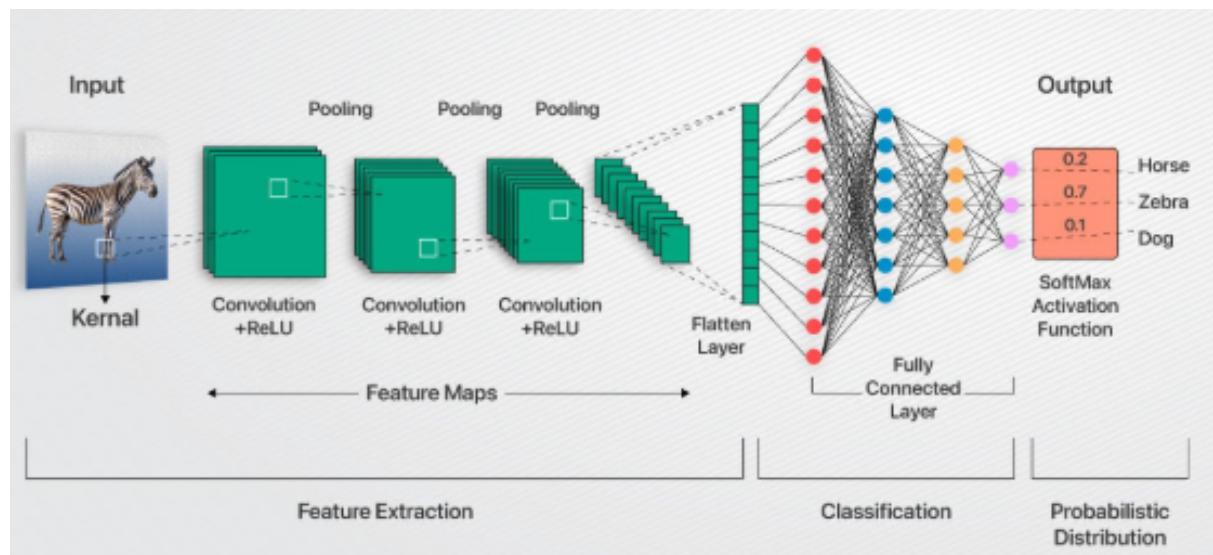


Figura 4.3: Arhitectura unei rețele neuronale convoluționale

Sursă: <https://www.analytixlabs.co.in/blog/convolutional-neural-network/>

(Accesată în 15-03-2025)

Stratul convoluțional (Convolutional Layer) este responsabil cu aplicarea filtrelor amintite anterior. Se pot detecta caracteristici simple precum colțuri, texturi, dar și trăsături complexe cum ar fi forme. Într-un strat se pot aplica în același timp mai multe filtre, astfel vor rezulta mai multe ieșiri, orientate pe tipuri diferite de trăsături. Dimensiunea rezultatului este influențată de modul în care filtrul este aplicat. Pentru a controla această dimensiune se poate utiliza padding sau stride. *Padding-ul* permite adăugarea de pixeli, în jurul marginilor imaginii. Acești pixeli au de obicei valoarea 0 și se adaugă înainte ca filtrul să fie aplicat. Adăugarea pixelilor previne pierderea informației aflate pe margini și asigură că imaginea își păstrează dimensiunea inițială. *Stride-ul* stabilește pasul cu care filtrul se deplasează. Valorile mari reduc dimensiunea ieșirii și durata de procesare, astfel influențând viteza de antrenare și nivelul de compresie a informației [30].

Aplicarea operației de convoluție este ilustrată în Figura 4.4. În partea stângă, se observă o matrice de 3x3. Aceasta e înconjurată de padding (valorile de 0 de pe margini).

În centru, este o matrice de 2x2, mai exact, acea matrice este filtrul. Acesta se deplasează peste matricea din partea stângă, în funcție de stride, și în fiecare poziție se efectuează următorul calcul: se înmulțește fiecare element din filtru cu elementul care îi corespunde în matrice, iar apoi rezultatele se adună. Valoarea finală este adăugată în matricea de ieșire, cea din partea dreaptă. Pentru exemplul din figură, calculul efectuat este $0*0 + 0*1 + 0*2 + 0*3 = 0$.

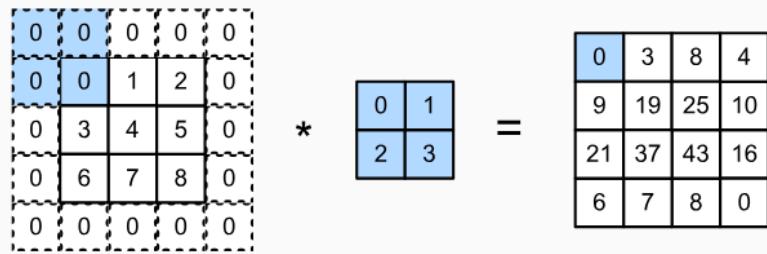


Figura 4.4: Aplicarea operației de conoluție

Sursă: https://classic.d2l.ai/chapter_convolutional-neural-networks/padding-and-strides.html/
(Accesată în 15-03-2025)

Stratul de activare (Activation Layer) introduce non-liniaritate pentru a permite modelului să distingă între datele dificil de diferențiat și pentru a facilita învățarea funcțiilor complexe [31]. Asupra hărților de activare se aplică funcțiile de activare:

- ReLU (Rectified Linear Unit): această funcție elimină valorile negative (4.1).

$$f(x) = \max(0, x) \quad (4.1)$$

În Figura 4.5 se prezintă un exemplu de aplicare a funcției ReLU. Valorile negative sunt înlocuite cu zero, iar cele pozitive sunt păstrate. Dezavantajul acestei funcții este ca poate duce la fenomenul de *dying ReLU*, în cazul în care unii neuroni vor fi constant 0.

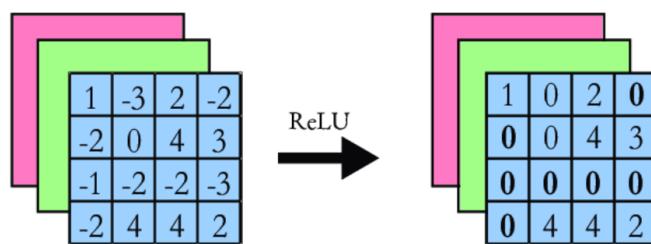


Figura 4.5: Aplicarea funcției de activare ReLU

Sursă: https://www.researchgate.net/figure/Example-of-ReLU-activation-function_fig5_331800773
(Accesată în 15-03-2025)

- Leaky ReLU: o variantă a funcției ReLU care permite o activare și pentru valorile negative (4.2).

$$f(x) = \begin{cases} x, & \text{dacă } x \geq 0 \\ \alpha x, & \text{dacă } x < 0 \end{cases} \quad (4.2)$$

Pentru atenuare se folosește α , un coeficient mic, pozitiv. Această abordare evită problema dying ReLU, permitând transmiterea de valori negative în rețea.

- Sigmoid: transformă valorile, mapându-le în intervalul 0-1. De obicei, este folosită în straturile finale (4.3).

$$\sigma(x) = \frac{1}{1 + e^{-x}} \quad (4.3)$$

- Tanh: este funcția tangentei hiperbolice (4.4). Este similară cu funcția sigmoid, dar ieșirile sunt mapate între -1 și 1.

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad (4.4)$$

Stratul dens (Fully Connected) este zona în care fiecare neuron este conectat la toti vecinii săi, din stratul anterior. Acest strat permite îmbinarea tuturor trăsăturilor extrase și realizează inferența. Rezultatul oferit de acest strat este predicția. Ieșirea furnizată de stratul dens este 4.5:

$$y = f(Wx + b) \quad (4.5)$$

Unde:

- y: este vectorul de ieșire;
- f: este funcția de activare aplicată;
- W: este matricea de ponderi;
- x: este vectorul de intrare;
- b: este vectorul de bias.

Matricea W are dimensiunile $n_{\text{ieșire}} \times n_{\text{intrare}}$ (numărul de neuroni din stratul precedent x numărul de neuroni din stratul curent). Fiecare element w_{ij} exprimă influența neuronului j din stratul anterior asupra neuronului i din stratul curent [31].

Stratul de pooling (Pooling Layer) reduce dimensiunea hărților de activare. În acest fel, complexitatea rețelei scade, se reduce numărul de parametri și se previne supraînvățarea [32]. Tipurile de pooling sunt ilustrate în Figura 4.6. *Max pooling* alege cea mai mare valoare din regiune. Astfel, rămân evidențiate doar cele mai relevante trăsături. O variantă de pooling care nu este la fel de restrictivă ca și max pooling este *average pooling*. Acest tip de pooling calculează media valorilor din regiune. Acest stat contribuie la eliminarea zgomotului.

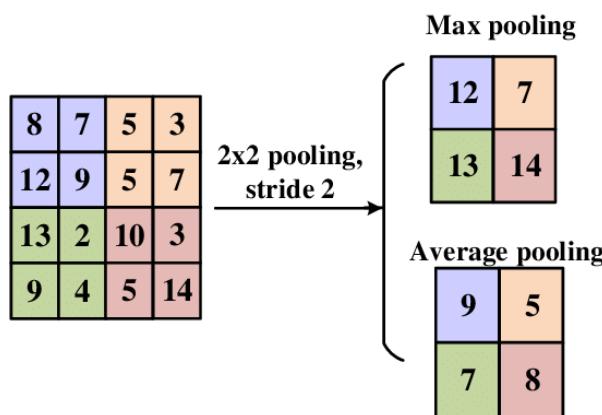


Figura 4.6: Tipuri de pooling

Sursă: <https://www.researchgate.net/figure/>

Pooling-layer-operation-approaches-1-Pooling-layers-For-the-function-of-decreasing-the_fig4_340812216
(Accesată în 15-03-2025)

Stratul de normalizare (Normalization Layer) are rolul de a stabiliza distribuția activărilor. Această distribuție poate fi afectată de variații care încetinesc antrenarea. Normalizarea diminuează probabilitatea ca modelul să ajungă la supraînvățare [31].

Tipuri de normalizare:

- Batch Normalization

Acest fel de normalizare acționează pe mini-batchuri. Ea transformă valorile pentru a ajunge la o medie egală cu zero și la o deviație standard unitară. Prin Batch Normalization se reduc variațiile dintre straturi, adică se evită fenomenul de *internal covariate shift*.

- Layer Normalization

Acest tip de normalizare este folosit pentru stabilizarea activărilor din întreg stratul. Metoda este utilă pentru arhitecturile în care dimensiunea batch-ului este variabilă sau mică.

- Instance Normalization

Această tehnică de normalizare acționează pe fiecare hartă de activare. Ea ajustează distribuția la nivel de canal, este utilă când e nevoie de stabilitate vizuală.

- Group Normalization

Normalizarea aceasta se bazează pe împărțirea canalelor unei hărți de activare în grupuri. Ulterior, fiecare grup este normalizat, individual.

4.1.4. Subînvățare, Supraînvățare și Generalizare

Un model antrenat într-un mod corespunzător trebuie să aibă atât capacitatea să învețe din datele pe care le are la dispoziție, cât și capacitatea de generalizare pentru datele noi. Generalizarea se referă la aptitudinile unui model de a asigura o performanță ridicată în momentele în care este testat cu date noi, pe care nu le-a întâlnit în setul de antrenament. Frecvent, în procesul de antrenare, apar următoarele probleme: subînvățarea și supraînvățarea.

Subînvățarea (underfitting) apare atunci când modelul este simplu și nu reușește să descopere trăsăturile relevante pe care setul de date le expune. În cazul subînvățării, modelul este predispus la erori pe setul de antrenament, dar și pe setul de validare. Printre cauzele acestei probleme se află antrenarea insuficientă, arhitectura prea simplă sau numărul insuficient de trăsături.

Supraînvățarea (overfitting) conținează situația în care modelul învață datele în mod excesiv, memorează chiar și zgomotele din setul de date de antrenament. Acesta își pierde abilitatea de a generaliza. În cazul supraînvățării, modelul are o acuratețe ridicată pe setul de antrenament, dar pe setul de date de validare sau de test, performanțele acestuia sunt scăzute. Această problemă apare dacă setul de date folosit pentru antrenare este prea mic, dacă nu se folosește regularizarea, dacă modelul învață prea mult sau este prea complex pentru datele avute [33].

Supraînvățarea poate fi evitată prin [34]:

- *Regularizare*

Regularizarea constă în adăugarea penalizărilor în funcția de pierdere a modelului. Prin aplicarea acestor penalizări, modelul înțelege să nu ajusteze în mod excesiv parametrii. Astfel, modelul învață o soluție mai simplă și mai generalizabilă.

Exemple de regularizare:

- **L1 (LASSO/Least Absolute Shrinkage and Selection Operator)**: elimină trăsăturile irelevante. L1 adaugă funcției de pierdere inițială, o penalizare corelată

cu valoarea absolută a sumei ponderilor 4.6.

$$J(\theta) = \mathcal{L}(\theta) + \lambda \sum_{i=1}^n |\theta_i| \quad (4.6)$$

Unde:

$\mathcal{J}(\theta)$ - funcția de cost total (funcția de pierdere + penalizare)

$\mathcal{L}(\theta)$ - funcția de pierdere inițială

λ - coeficientul de regularizare (controlează intensitatea penalizării)

θ_i - ponderile modelului

- **L2 (Ridge):** micsorează coeficientii. L2 adaugă o penalizare proporțională cu suma pătratelor ponderilor (4.7) și evită ajustarea exagerată a acestora.

$$J(\theta) = \mathcal{L}(\theta) + \lambda \sum_{i=1}^n \theta_i^2 \quad (4.7)$$

Termenii din ecuația 4.7 au aceeași semnificație ca și în ecuația anterioară (4.6).

- **Dropout:** tehnică care la fiecare epocă de antrenare, presupune dezactivarea unui procent din neuroni. Acești neuroni sunt aleși aleatoriu. Această abordare face ca rețeaua să fie mai flexibilă și să nu depindă mereu de aceeași neuroni.

- *Oprire timpurie (Early Stopping)*

Această procedură previne supraînvățarea prin oprirea modelului în timpul învățării. Se monitorizează performanțele obținute pe setul de date de antrenament și pe cel de validare. Dacă în timpul antrenării eroarea scade, iar la validare aceasta crește, antrenamentul este oprit.

- *Augmentarea datelor (Data Augmentation)*

Prin această metodă se generează date noi. Se pornește de la datele existente. În cazul de față, pentru imagini, se modifică fotografile care fac parte din setul de date prin rotirea acestora, prin scalare, prin schimbarea contrastului sau a luminozității, prin adăugarea de zgomote. Astfel, modelul învață trăsături mai generale deoarece datele sunt diversificate.

- *Creșterea setului de date*

Modelele complexe au nevoie de un volum ridicat de date pentru antrenament. Dacă datele nu sunt suficiente, modelul riscă să ajungă în punctul în care memorează zgomote sau alte trăsături care nu sunt relevante.

- *Validare încrucișată (Cross-Validation)*

Setul de date se împarte în mai multe subseturi (fold-uri) de dimensiuni egale. Acest lucru este efectuat pentru a permite antrenarea și validarea modelului de mai multe ori. De fiecare dată este ales pentru validare un subset diferit, iar restul sunt utilizate pentru antrenare. Capacitatea de generalizare a modelului este apoi estimată prin calcularea unei medii a performanțelor obținute în fiecare etapă.

În Figura 4.7 se prezintă un exemplu de validare încrucișată, unde datele au fost împărțite în 5 subseturi egale. La fiecare pas, un subset este utilizat pentru validare (cel marcat cu albastru), iar celelalte sunt folosite la antrenament. Acest proces se repetă până când fiecare subset este folosit o dată pentru validare. La fiecare pas este calculată o performanță, iar performanța finală este reprezentată de media rezultatelor furnizate de către fiecare iterație.

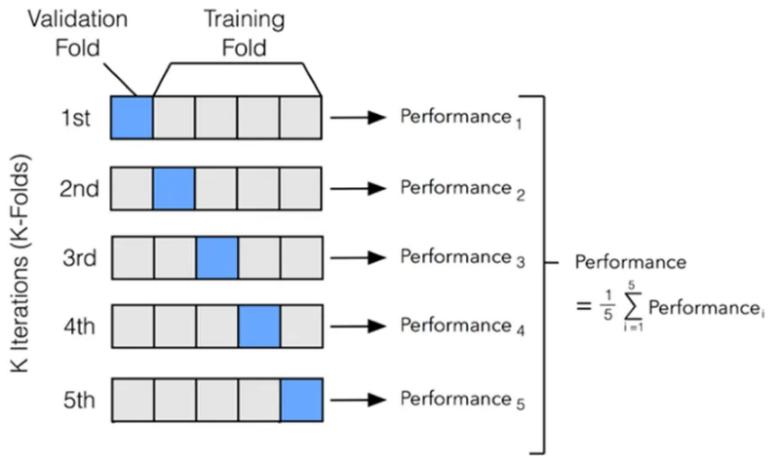


Figura 4.7: Validare încrucișată 5-Fold

Sursă: <https://ompramod.medium.com/cross-validation-623620ff84c2>
(Accesată în 17-03-2025)

- *Reducerea complexității arhitecturii*

Complexitatea unui model poate fi simplificată prin reducerea numărului de straturi sau scăderea dimensiunii acestora, alegerea unor arhitecturi mai simple. Această strategie este utilă atunci când numărul datelor este redus.

Figura 4.8 creionează trei situații care pot apărea în momentul când se antrenează un model: subînvățare, generalizare corectă și supraînvățare. În partea stângă, se poate observa procesul de subînvățare. Modelul nu a reușit să învețe din datele pe care le-a avut la dispoziție și nu a identificat nici măcar un tipar. Distribuția punctelor este mult mai complexă decât curba obținută. În imaginea din centru, se remarcă faptul că modelul generalizează corect. Distribuția punctelor se potrivește cu curba rezultată. Variațiile locale nu sunt urmărite în mod excesiv. În imaginea din partea dreaptă, se observă supraînvățarea. Modelul este complex, iar fiecare variație a datelor de antrenament este urmărită, fapt dovedit de curba identificată. La apariția datelor noi se întâmpină o problemă deoarece modelul nu știe să generalizeze. Acesta a învățat doar datele de antrenament.

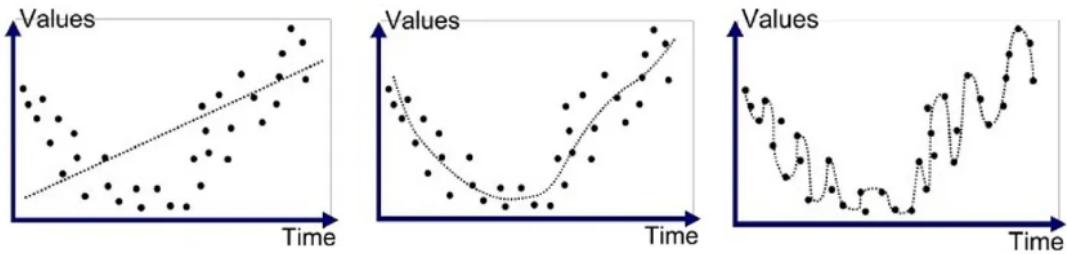


Figura 4.8: Reprezentare vizuală a subînvățării, a generalizării corecte și a supraînvățării

Sursă: <https://medium.com/greyatom/what-is-underfitting-and-overfitting-in-machine-learning-and-how-to-deal-with-it-6803a989c76>
(Accesată în 17-03-2025)

4.1.5. Optimizarea rețelelor neuronale

Optimizarea rețelelor neuronale urmărește minimizarea unei funcții de pierdere și maximizarea abilității de generalizare a modelului [23].

Funcții de pierdere (Loss) Funcțiile de pierdere reprezintă modul în care un model apreciază diferența dintre predicțiile pe care le face și valorile reale care sunt cunoscute din datele de antrenament. Ele oferă indicații despre cum ar trebui ponderile să fie actualizate deoarece se încearcă să se ajungă la erori minime. Funcția de pierdere se alege ținând cont de tipul sarcinii și de datele disponibile.

Mean Squared Error (MSE) este utilizată de obicei în regresie. Această funcție penalizează mai drastic erorile mari 4.8.

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad (4.8)$$

Unde:

- y_i reprezintă valoarea reală (eticheta adevărată);
- \hat{y}_i este valoarea prezisă de model;
- n este numărul de exemple din setul de date.

Binary Cross-Entropy (BCE) este utilizată pentru clasificarea binară. Această funcție penalizează greșelile în funcție de încrederea modelului, ține cont de probabilitățile prezise 4.9. Termenii din ecuația următoare au aceeași semnificație ca și în ecuația 4.8.

$$BCE = -\frac{1}{n} \sum_{i=1}^n [y_i \cdot \log(\hat{y}_i) + (1 - y_i) \cdot \log(1 - \hat{y}_i)] \quad (4.9)$$

Categorical Cross-Entropy (CCE) este utilizată pentru problemele de clasificare care au mai multe clase. Dacă predicțiile au un grad ridicat de veridicitate, valoarea CCE este mică 4.10.

$$CCE = - \sum_{i=1}^n \sum_{j=1}^C y_{ij} \cdot \log(\hat{y}_{ij}) \quad (4.10)$$

Unde:

- n este numărul de exemple din setul de date;
- C este numărul de clase;
- y_{ij} reprezintă valoarea reală, este 1 dacă exemplul aparține clasei j , altfel 0;
- \hat{y}_{ij} este probabilitatea prezisă de model ca exemplul i să aparțină clasei j .

Focal Loss este folosită pentru sarcinile de clasificare unde sunt mai multe clase. Aceasta ia în calcul dezechilibrul datelor din clase (4.11) și previne supraînvățarea. Categorical Cross-Entropy penalizează toate greșelile la fel, iar dacă setul de date este dezechilibrat, se poate ajunge la situația în care modelul ignoră clasele mai rare și învață doar trăsăturile claselor dominante. Focal Loss încearcă să combată problemele care apar când se folosește Categorical Cross-Entropy, prin accentuarea învățării pe exemplele care sunt clasificate incorrect sau cu incertitudine.

$$\mathcal{L}_{\text{focal}} = - \sum_{i=1}^n \sum_{j=1}^C \alpha \cdot (1 - \hat{y}_{ij})^\gamma \cdot y_{ij} \cdot \log(\hat{y}_{ij}) \quad (4.11)$$

Unde:

- $\alpha \in [0, 1]$ este un factor de echilibrare între clase;
- γ este un factor de focalizare care reduce contribuția instanțelor ușor de clasificat;
- restul termenilor au aceeași semnificație, analog formulei 4.10.

În cadrul proiectului, am ales să utilizez această funcție de pierdere deoarece între

clasele analizate sunt dezechilibre. Astfel, Focal Loss reduce influența exemplelor ușor de clasificat și contribuie la îmbunătățirea rezultatelor obținute.

Algoritmi de optimizare

Gradient Descent (GD) actualizează toți parametrii modelului utilizând gradientul calculat pe baza întregului set de antrenament. Rezultatele sunt stabile, dar costurile sunt ridicate.

Stochastic Gradient Descent (SGD) actualizează parametrii pe baza unui singur exemplu sau mini-batch. Performanțele sale sunt influențate de funcția de pierdere aleasă și de rata de învățare.

RMSprop (Root Mean Square Propagation) normalizează gradientul în funcție de valoarea sa anterioară. Aceasta ajustează rata de învățare pentru fiecare parametru.

Adam (Adaptive Moment Estimation) ajustează parametrii modelului în funcție de istoricul gradientului. Acest algoritm se bazează pe momentele de ordin I și II ale gradientului. Printre avantajele care m-au determinat să utilizez algoritmul Adam se numără ajustarea automată a ratelor de învățare pentru fiecare parametru și stabilitatea la variațiile gradientului. Adam este eficient din punct de vedere computațional. Un dezavantaj al acestuia ar fi predispunerea la supraînvățare, de aceea el trebuie folosit alături de unele metode de regularizare.

Rata de învățare

Rata de învățare este un parametru care determină pasul cu care sunt modificate parametrii modelului. O rată de învățare adecvată asigură convergența modelului. Dacă rata de învățare are valori prea mici, antrenarea poate să dureze mult timp deoarece sunt necesare mai multe epoci până când modelul să atingă convergență. În altă ordine de idei, dacă valorile sunt prea mari, modelul poate să ducă la divergență și să nu identifice minimele locale. Cosine Decay este o metodă care ajustează rata de învățare în mod dinamic. În cadrul acestei tehnici, cu cât numărul de epoci crește, se folosește o funcție cosinusoidală pentru a scădea rata de învățare. În acest fel, modelul reușește să generalizeze mai bine și supraînvățarea este diminuată.

Figura 4.9 prezintă evoluția ratei de învățare pe parcursul antrenării folosind strategia Cosine Decay. Inițial, modelul pornește cu o rată de învățare mai mare, care în timp ce numărul epocilor crește, scade urmărind o curbă cosinusoidală. Această scădere controlată ajută la stabilizarea procesului de antrenare. Inițial, pașii sunt mari și permit investigarea eficientă a spațiului de căutare. Spre final, pașii devin tot mai mici, contribuind la ajustarea fină a parametrilor.

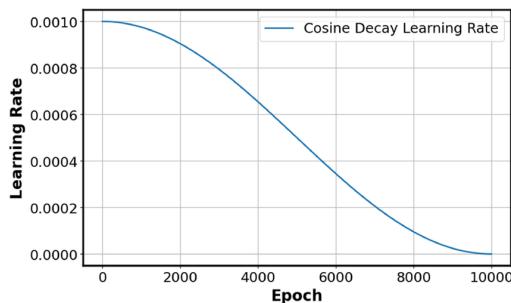


Figura 4.9: Evoluția ratei de învățare folosind Cosine Decay

Sursă: https://www.researchgate.net/figure/Cosine-decay-learning-rate-curve_fig3_387142029
(Accesată în 19-03-2025)

4.1.6. Metriki de evaluare a performanței

Evaluarea performanței unui model de învățare automată presupune folosirea unor metriki care conturează capacitatea de generalizare și calitatea predicțiilor propuse. Alegera metrikiilor potrivite depinde de tipul sarcinii pe care modelul o desfășoară. Pentru clasificare sunt adecvate următoarele metriki:

- **Acuratețea** (4.12, 4.13)

Acuratețea reprezintă raportul dintre numărul de predicții corecte și numărul total de predicții efectuate. Aceasta se obține prin comparația etichetelor prezise cu etichetele reale. Această metrică nu este suficientă, ea trebuie studiată alături de alte metriki (de exemplu: Scorul F1).

$$\text{Acuratețe} = \frac{\text{Număr predicții corecte}}{\text{Număr total exemple}} \quad (4.12)$$

$$\text{Acuratețe} = \frac{TP + TN}{TP + TN + FP + FN} \quad (4.13)$$

Unde:

- TP = True Positives (pozitive corect prezise);
- TN = True Negatives (negative corect prezise);
- FP = False Positives (pozitive incorrect prezise);
- FN = False Negatives (negative incorrect prezise).

- **Precizia** (4.14)

Precizia prezintă câte din exemplele estimate ca aparținând unei clase, sunt într-adevăr din clasa respectivă. De obicei se calculează pentru fiecare clasă. Rezultatele acestei metriki pot fi îmbunătățite prin utilizarea unei funcții de pierdere care penalizează diferit exemplele, în funcție de dezechilibrul claselor din care fac parte.

$$\text{Precizie} = \frac{TP}{TP + FP} \quad (4.14)$$

Unde:

- TP = True Positives (cazuri pozitive corect clasificate);
- FP = False Positives (cazuri negative clasificate greșit ca fiind pozitive).

- **Recall** (4.15)

Măsoară abilitatea modelului de a detecta care exemple dintr-o clasă sunt relevante.

$$\text{Recall} = \frac{TP}{TP + FN} \quad (4.15)$$

Unde:

- TP = True Positives (cazuri pozitive corect clasificate);
 - FN = False Negatives (cazuri pozitive clasificate greșit ca fiind negative).
- Pentru ca această metrică să aibă rezultate mai bune, se pot folosi funcții de pierdere care țin cont de dezechilibrul dintre clase sau se pot augmenta datele.

- **Scorul F1** (4.16)

Scorul F1 reprezintă media armonică dintre recall și precizie. Această metrică este utilă când datele nu sunt echilibrate și acuratețea poate oferi informații eronate.

$$F_1 = 2 \cdot \frac{\text{Precizie} \cdot \text{Recall}}{\text{Precizie} + \text{Recall}} \quad (4.16)$$

- **Matricea de confuzie**

Matricea de confuzie este o reprezentare vizuală a predicțiilor. Fiecare rând corespunde etichetelor reale, iar coloanele sunt asociate cu etichetele prezise. Această matrice este folosită pentru a observa ce greșeli face modelul și ce clase confundă. Pe diagonala principală a matricei apar exemplele prezise corect. Toate valorile care nu se află pe această diagonală reprezintă erori făcute de model.

Performanța obținută este influențată de: calitatea setului de date, dezechilibrul dintre clase, complexitatea arhitecturii utilizate și tehniciile de regularizare aplicate [23].

4.1.7. Modele de clasificare utilizate

Analiza posturală se efectuează cu ajutorul clasificării imaginilor în care oamenii adoptă anumite posturi. Identificarea celui mai bun model pentru această sarcină a presupus studierea și încercarea mai multor arhitecturi de rețele neuronale convoluționale despre care se discută în literatura de specialitate.

ResNet50 face parte din familia ResNet (Residual Networks). Arhitectura modelului este formată din 50 de straturi. Aceasta introduce legăturile reziduale care permit trecerea informației între straturi care nu sunt consecutive. Acest mecanism facilitează antrenarea unor rețele foarte adânci.

În procesul de antrenare al rețelelor neuronale adânci apare fenomenul numit vanishing gradient. Aceasta se referă la situația în care, în timpul procesului de propagare, gradientul devine din ce în ce mai mic pe măsură ce este transmis spre straturile de început ale rețelei. Astfel, aceste straturi nu reușesc să învețe sau învăță foarte lent, afectând performanța și capacitatea de generalizare a modelului.

Pentru a rezolva această problemă, arhitectura ResNet introduce blocurile reziduale ilustrate în Figura 4.10. Acestea permit transmiterea directă a informației de la intrarea în bloc până la ieșirea acestuia, ocolind procesarea intermediară realizată de straturile convoluționale. Dacă straturile intermediare nu adaugă informație relevantă, rețeaua are capacitatea de a sări peste ele, permitând ca datele să fie transmise mai departe nemodificate. Acest mecanism crește stabilitatea și eficiența antrenării rețelei, evită învățarea excesivă a detaliilor, astfel reducând supraînvățarea.

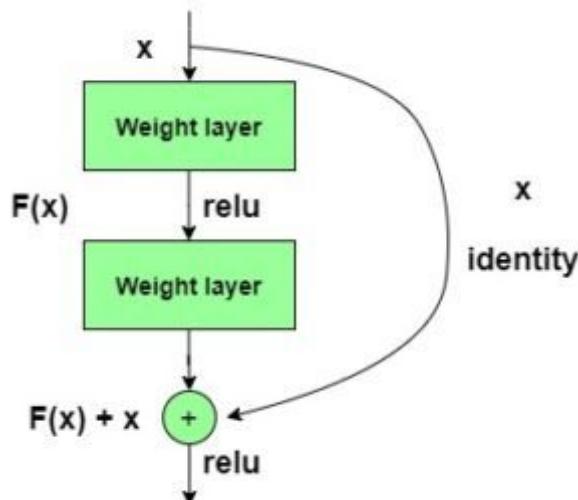


Figura 4.10: Bloc rezidual

Sursă: <https://www.geeksforgeeks.org/machine-learning/introduction-to-residual-networks/>
(Accesată în 07-05-2025)

Unul dintre dezavantajele principale ale arhitecturii ResNet50 este dimensiunea modelului, care cuprinde aproximativ 23 de milioane de parametri. Această complexitate structurală implică un consum ridicat de resurse atât în faza de antrenare, cât și în timpul inferenței, ceea ce îl face mai puțin potrivit pentru dispozitive mobile.

Arhitectura modelului este prezentată în Figura 4.11. Arhitectura este împărțită în mai multe etape care conțin blocuri convoluționale și blocuri reziduale. Modelul începe cu o etapă de preprocesare (zero padding, convoluție, normalizare, activare ReLU și pooling), urmată de patru stadii de învățare. La final, caracteristicile extrase sunt aggregate printr-un strat de pooling și trimise către un strat dens pentru realizarea predicției finale.

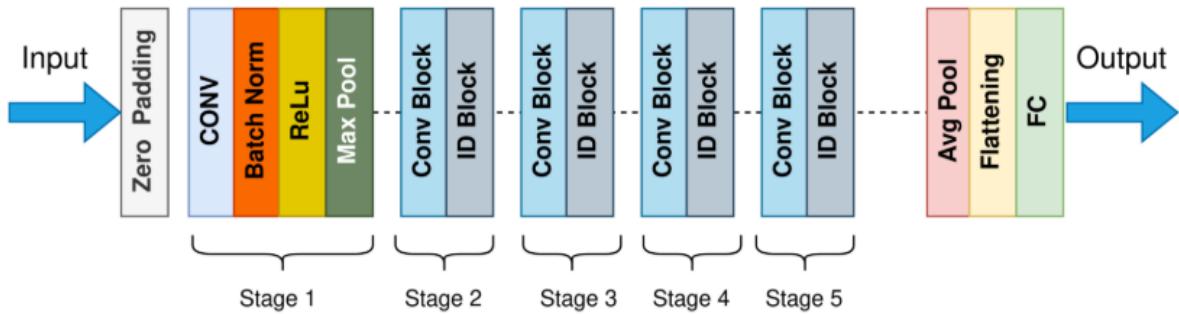


Figura 4.11: Arhitectura modelului ResNet50
Sursă: <https://towardsdatascience.com/the-annotated-resnet-50-a6c536034758/>
(Accesat în 07-05-2025)

ResNet este performant pe seturi mari de date, dar în momentul în care datele sunt dezechilibrate, acesta poate ajunge la supraînvățare [35].

EfficientNetB0 și **EfficientNetB3** sunt arhitecturi dezvoltate de Google. Acestea fac parte din familia EfficientNet. Această familie s-a remarcat prin echilibrul dintre acuratețe și eficiență computațională. Aceste modele s-au remarcat prin performanță ridicată obținută cu un număr mai mic de parametri comparativ cu alte arhitecturi.

Modelele folosesc scalarea compusă, adică ajustează în același timp adâncimea retelei (prin numărul de straturi), lățimea (prin numărul de filtre) și rezoluția datelor de intrare.

EfficientNetB0 este modelul inițial al acestei familii. Acesta utilizează procesul cunoscut sub numele de Neural Architecture Search (NAS). Această tehnică presupune utilizarea unui algoritm care explorează posibile structuri de rețele neuronale și evaluează performanța acestora pe seturi de date, cu scopul de a identifica arhitectura optimă.

Modelul folosește blocuri MBConv (Mobile Inverted Bottleneck Convolution). Acestea sunt o versiune îmbunătățită a strukturilor convoluționale clasice, fiind optimizate pentru eficiență pe dispozitive mobile. MBConv efectuează următoarele operații:

- Crește temporar numărul de canale pentru extragerea de caracteristici complexe (expansiune);
- Aplică filtre pe fiecare canal separat, reducând numărul de operații (convoluție depthwise);
- Comprimă rezultatul pentru a reveni la dimensiunea originală (convoluție pointwise).

EfficientNetB3 reprezintă o versiune îmbunătățită a modelului EfficientNetB0. Acest model acceptă o rezoluție mai mare a imaginii de intrare și prezintă un număr crescut de filtre în fiecare strat. El oferă o precizie mai ridicată în sarcinile de clasificare, fiind potrivit pentru aplicații unde resursele hardware permit rularea modelelor mai

mari. Pentru a fi utilizat într-o aplicație mobilă, acesta necesită tehnici suplimentare de optimizare.

În Tabelul 4.1 se prezintă principalele diferențe dintre EfficientNetB0 și EfficientNetB3.

Tabela 4.1: Comparație între EfficientNetB0 și EfficientNetB3

Caracteristică	EfficientNetB0	EfficientNetB3
Rezoluție imagine	224×224	300×300
Număr de parametri	5,3 milioane	12 milioane
Acuratețe (ImageNet)	77%	81,6%
Timp de inferență	Reduc	Mai ridicat
Potrivire cu aplicații mobile	Excelentă	Necesită optimizare

Structura rețelei EfficientNet este alcătuită dintr-o succesiune de blocuri MBConv, fiecare cu rolul de a extrage și procesa trăsături. Fiecare bloc este definit printr-un tip de convecție, o dimensiune de kernel specifică. În Figura 4.12 este ilustrată arhitectura EfficientNet și sunt evidențiate principalele etape ale procesării.

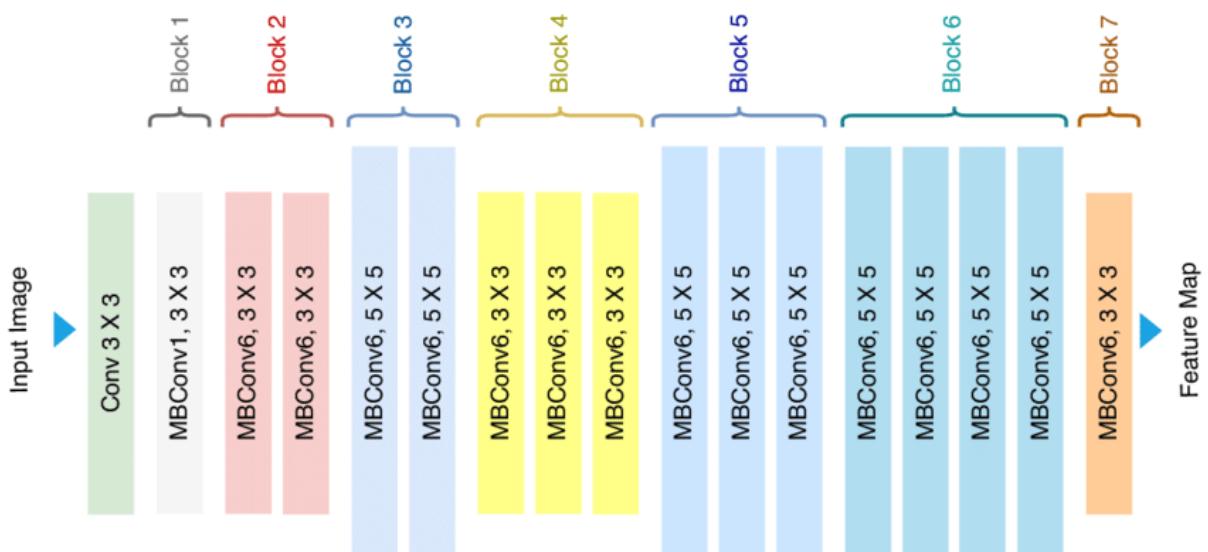


Figura 4.12: Arhitectura modelului EfficientNet

Sursă: <https://iq.opengenus.org/efficientnet/>

(Accesată în 09-05-2025)

Modelele EfficientNetB0 și EfficientNetB3 demonstrează că performanța în clasificarea imaginilor poate fi obținută prin utilizarea blocurilor MBConv și scalarea compusă. EfficientNetB0 se remarcă prin dimensiunea redusă și viteza de inferență, este ideal pentru aplicații mobile. EfficientNetB3 aduce o creștere a acurateței, fiind o variantă optimă în scenarii unde precizia este esențială, iar resursele hardware permit integrarea unui model mai extins [36].

EfficientNetV2B0 face parte din familia EfficientNetV2, care este o versiune mai bună a familiei EfficientNet. Modelele din această familie oferă viteze de antrenare mai mari, performanță superioară pe seturi mici de date și o capacitate mai bună de generalizare. EfficientNetV2B0 este versiunea de bază, potrivită pentru aplicații mobile. Timpul de inferență este scurt, iar numărul de parametrii este redus.

EfficientNetV2B0 aduce îmbunătățiri semnificative prin utilizarea blocurilor Fused-MBConv (combină operațiile de expansiune și convoluție depthwise într-o singură etapă). Această unificare reduce timpul de antrenare și inferență. Modelul beneficiază de tehnici avansate de regularizare și augmentare, cum ar fi progressive learning (creșterea graduală a complexității datelor în timpul antrenării) și stochastic depth (dezactivarea aleatorie a unor straturi în timpul antrenării pentru a preveni supraînvățarea). Aceste strategii contribuie la obținerea unor rezultate mai stabile pe seturi de date mici sau dezechilibrate [37].

MobileNetV2 este un model dezvoltat de Google. Acesta a fost optimizat pentru dispozitive mobile și embedded, unde resursele sunt limitate.

Arhitectura sa folosește blocuri care la început extind informația pentru a extrage caracteristici complexe, apoi o reduc din nou înainte de ieșire. Uneori, aceste blocuri păstrează legătura directă între intrare și ieșire, ceea ce ajută la menținerea informațiilor esențiale și la reducerea pierderii din procesul de învățare [38]. Figura 4.13 ilustrează arhitectura completă a rețelei, dar și structura blocurilor amintite anterior.

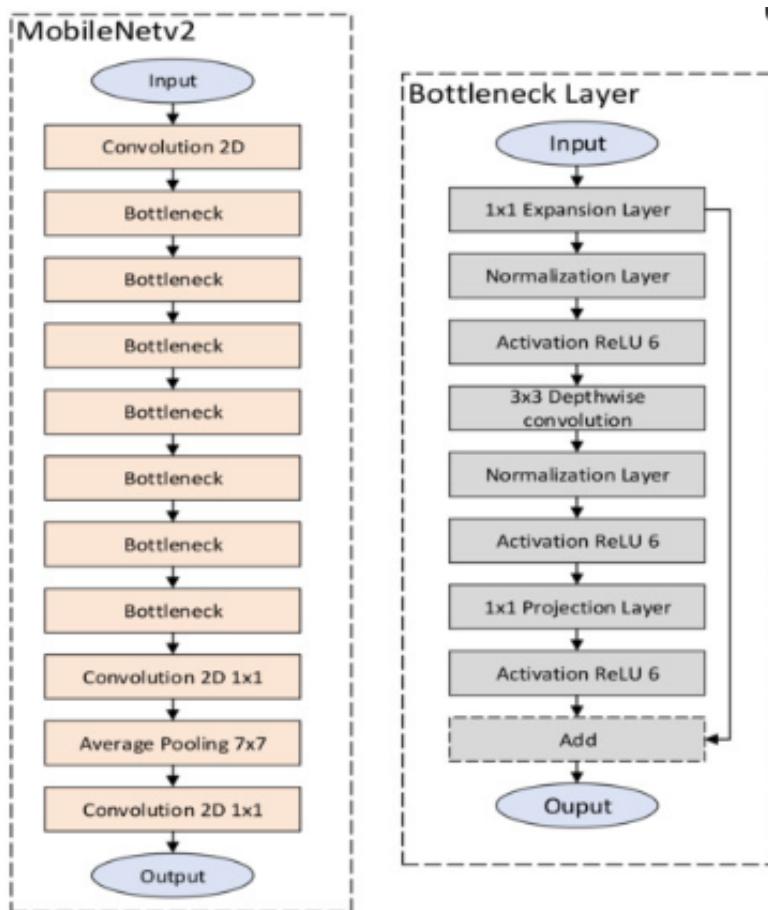


Figura 4.13: Arhitectura modelului MobileNetV2

Sursă: <https://www.mdpi.com/1424-8220/22/12/4318>
(Accesată în 14-05-2025)

DenseNet121 (Densely Connected Convolutional Network) este o arhitectură de rețea neuronală convoluțională care propune un nou mod de conectivitate a straturilor. Dacă la rețelele neuronale convoluționale tradiționale fiecare strat primește ca input doar ieșirea stratului anterior, DenseNet conectează fiecare strat cu toate straturile anterioare.

Această abordare favorizează reutilizarea trăsăturilor și ajută la obținerea unui număr redus de parametri [39].

Arhitectura modelului este ilustrată în Figura 4.14. Structura are la origine o conoluție care este urmată de un strat de pooling. Acest strat de pooling pregătește datele pentru extragerea de trăsături. Arhitectura este organizată în patru blocuri dense (Dense Blocks). Între aceste blocuri se regăsesc straturi de tranziție (Transition Layers) care aplică conoluții și pooling pentru a reduce dimensiunea și pentru a menține eficiența. În final, caracteristicile sunt comprimate printr-un strat de Global Average Pooling, urmat de un strat de clasificare Softmax care generează predicțiile.

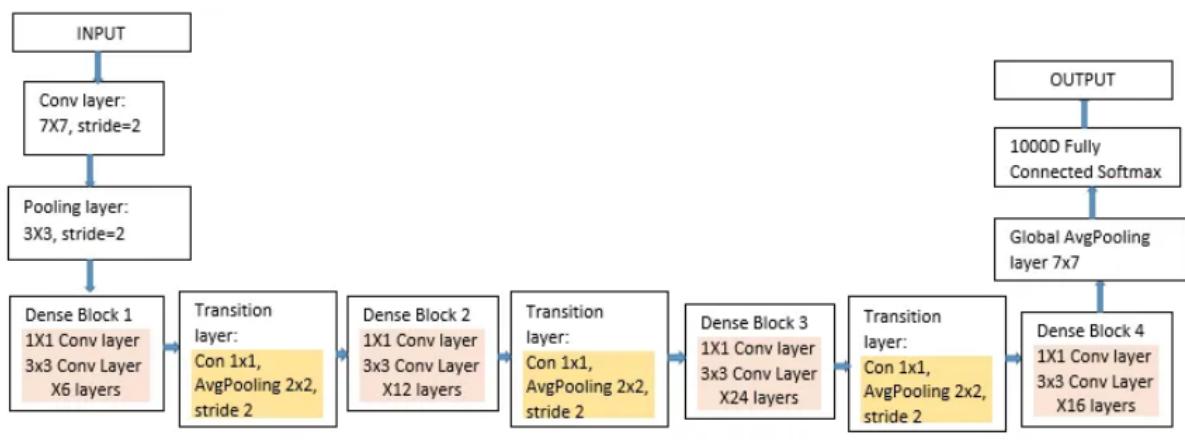


Figura 4.14: Arhitectura modelului DenseNet121

Sursă: <https://salonilokeshdutta.medium.com/densenet-architecture-with-explanation-375ef32a727b>
(Accesată în 21-05-2025)

DenseNet121 oferă performanțe ridicate chiar dacă setul de date este limitat. Connectivitatea propusă în arhitectura modelului face ca în timpul antrenării să fie nevoie de mai multe resurse.

Analiza rețelelor conoluționale prezentate anterior a fost esențială pentru identificarea celui mai potrivit model care să realizeze clasificarea posturală. Performanțele obținute de fiecare model testat vor fi detaliate în capitolul 6, care este dedicat testării și validării.

4.2. Setul de date

Setul de date utilizat în cadrul acestui studiu este construit preponderent din imagini proprii, selectate manual din diverse surse online și seturi de date existente. Imaginile surprind persoane aflate în poziție așezată, în special la birou. Calitatea vizuală a imaginilor este variabilă: unele prezintă rezoluție scăzută, orientări necorespunzătoare, zgomote sau imperfecțiuni. Imaginile sunt supuse variațiilor de culoare și lumină, care au îngreunat procesul de etichetare. Fotografile includ perspective frontale, laterale și posterioare, iar gradul de acoperire al corpului variază. Unele imagini surprind întregul corp, în timp ce altele includ doar o parte din acesta (de exemplu: doar partea superioară a corpului e vizibilă, picioarele sunt absente sau lipsește capul).

Pentru construcția setului, au fost analizate mai multe colecții de date publice, chiar dacă scopul lor inițial nu corespunde obiectivului propus în lucrarea curentă. Imaginile au fost filtrate manual, alegându-se exclusiv cele care oferă context postural relevant. Sursele includ:

- Setul de date *Sitting vs. Standing* [40] conține imagini cu persoane în poziție așezată și în picioare;
- Două seturi de date care conțin imagini cu oameni care efectuează activități precum alergare, stat jos, umblat sau dormit [41, 42];
- Setul de date *Employee Productivity 2.0* [43] este folosit pentru a vedea care este productivitatea angajaților la locul de muncă, în funcție de activitatea pe care o efectuează. Sunt imagini cu oameni care stau la birou, dorm, fumează.

Distribuția imaginilor pe clase prezintă dezechilibru. Clasa slouched_posture are exemple prea puține. Pentru a nu ajunge la efecte negative, în timpul antrenării s-au folosit ponderi calculate în funcție de frecvența fiecărei clase. Aceste ponderi contribuie la penalizarea diferență a greșelilor făcute de model. Distribuția imaginilor în setul de date este următoarea:

- neutral_posture: 277 imagini;
- postural_asymmetry: 494 imagini;
- slouched_posture: 113 imagini;
- stooped_posture: 229 imagini;
- symmetrical_front: 390 imagini.

Setul de date final a fost împărțit în trei subseturi: 80% din imagini au fost folosite pentru antrenarea modelului, 20% pentru validare, iar un set de 10% a fost utilizat pentru testarea finală.

4.2.1. MediaPipe

În cadrul acestei lucrări, pentru a facilita procesul de etichetare a imaginilor în funcție de postura utilizatorilor, s-a utilizat framework-ul MediaPipe, dezvoltat de compania Google. MediaPipe este o platformă open-source care permite dezvoltarea de pipeline-uri care procesează date în timp real. Unul dintre cele mai relevante module pentru această lucrare este Pose Estimation, care permite detectia automată a 33 de puncte cheie anatomici de pe corpul uman. MediaPipe Pose Estimation este antrenat pe mai multe seturi de date și are o latență redusă, fapt care îl face potrivit pentru aplicațiile care oferă feedback în timp real [44]. Punctele pe care modelul le poate detecta sunt înfățișate în Figura 4.15, iar legenda este în Figura 4.16.

Pentru detectarea punctelor, modelul parcurge două etape:

1) Detectia corpului: inițial, se identifică prezența corpului într-o imagine, iar apoi acesta este încadrat într-un bounding box.

2) Aproximarea punctelor: modelul prezice locația celor 33 de puncte 2D și le poziționează în interiorul box-ului.

Modelul returnează pentru fiecare punct anatomic, următoarele informații:

- **x**: poziția punctului pe axa orizontală (normalizată între 0 și 1);
- **y**: poziția punctului pe axa verticală (normalizată între 0 și 1);
- **z**: adâncimea punctului față de cameră;
- **vizibilitatea**: o valoare între 0 și 1 care indică probabilitatea ca acel punct să fie vizibil.

Am ales utilizarea MediaPipe pentru că printre cele 33 de puncte cheie anatomici pe care le detectează se află și cele de care am nevoie pentru etichetarea imaginilor și pentru analiza posturii. Această alegere s-a bazat pe acuratețea estimărilor 3D furnizate de modelul Pose Landmarker și pe latența mică pe care o oferă, fiind optimizat pentru procesare în timp real. Utilizarea MediaPipe are și anumite limitări. În cazul imaginilor slab calitative, cu fundal aglomerat sau care prezintă variații ale luminii, acuratețea scade.

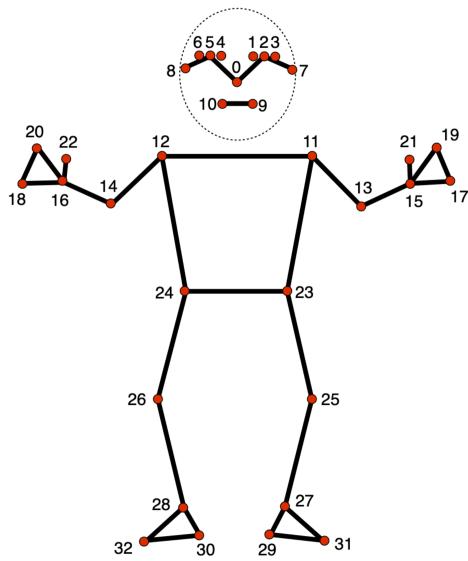


Figura 4.15: Punctele de pe corp detectate de MediaPipe

Sursă: https://ai.google.dev/edge/mediapipe/solutions/vision/pose_landmarker
(Accesată în 15-04-2025)

0 - nose
1 - left eye (inner)
2 - left eye
3 - left eye (outer)
4 - right eye (inner)
5 - right eye
6 - right eye (outer)
7 - left ear
8 - right ear
9 - mouth (left)
10 - mouth (right)
11 - left shoulder
12 - right shoulder
13 - left elbow
14 - right elbow
15 - left wrist
16 - right wrist
17 - left pinky
18 - right pinky
19 - left index
20 - right index
21 - left thumb
22 - right thumb
23 - left hip
24 - right hip
25 - left knee
26 - right knee
27 - left ankle
28 - right ankle
29 - left heel
30 - right heel
31 - left foot index
32 - right foot index

Figura 4.16: Legenda punctelor anatomico-detectate de MediaPipe

Sursă: https://ai.google.dev/edge/mediapipe/solutions/vision/pose_landmarker
(Accesată în 05-04-2025)

4.2.2. Etichetarea setului de date

Identificarea posturii utilizatorului poate fi realizată prin analiza punctelor cheie de pe corp care au fost returnate automat de către MediaPipe. Coordonatele acestor puncte au fost analizate și utilizate pentru a calcula niște unghiuri între segmente ale corpului. În funcție de punctele de pe corp, mai exact în funcție de poziționarea acestora și de unghiiurile determinante, s-a decis dacă persoana are o postură corectă sau nu. Dacă poziția este incorectă, se identifică principala problemă care poate provoca dureri. Întregul procedeu se bazează pe analiza coordonatelor returnate de MediaPipe, în urma detectiei umerilor, urechilor, soldurilor, nasului, genunchilor. Persoana care desfășoară o activitate la birou apare în diferite ipostaze, iar uneori detectia tuturor punctelor anterior menționate nu este posibilă. Astfel, regulile pe baza cărora setul de date a fost etichetat au fost adaptate pentru a fi cât mai generale (să fie potrivit pentru orice fel de imagine), dar în același timp, cât mai clare pentru a identifica în mod corespunzător problema posturală cu care persoana din imagine se confruntă. Evident, analiza se face în raport cu o postură ideală, dar sunt permise mici abateri. Poziția corectă este prezentată în Figura 3.3, din Capitolul 3. Se observă că spatele este drept, trunchiul nu este înclinat, capul nu este aplecat în față. Mușchii sunt fermi, susțin coloana vertebrală fără să ofere impresia că persoana alunecă de pe scaun. Aceste aspecte conturează postura ideală descrisă de fizioterapeuți [45].

Pentru clasificarea imaginilor nu s-a analizat poziția mâinilor și nici picioarele de la genunchi în jos deoarece acestea își schimbă poziția destul de des și atunci nu sunt chiar relevante pentru a decide corectitudinea unei posturi. Inițial, am optat pentru etichetarea pozelor în următoarea manieră, în funcție de descrierea dominantă a posturii: poziție neutră și corectă, asimetrie posturală, capul aplecat înainte, partea superioară/inferioară

a spitelui cu probleme. După etichetare am observat că de cele mai multe ori când capul era prea aplecat spre înainte era dezvoltată și o problemă la partea superioară a spitelui, mai exact se observa o cifoză (zona toracică posterioară era cocoșată). Astfel am decis să găsesc o nouă etichetă care să cuprindă ambele probleme, iar în acest fel am ajuns la eticheta de „postură cocoșată”. Deoarece eticheta poziției neutre conținea extrem de multe imagini comparativ cu restul claselor, am decis să o divizez în funcție de tipul imaginii. Am introdus încă o etichetă care să caracterizeze poziția neutră a imaginilor efectuate de la spate sau ale celor frontale, iar cea anteroară a rămas doar pentru imaginile în care persoana e vizibilă din lateral. S-au folosit șase clase pentru etichetarea imaginilor. Aceste clase sunt prezentate în Tabelul 4.2:

Tabela 4.2: Descrierea etichetelor utilizate pentru clasificarea posturii

Etichetă	Denumire	Descriere
neutral_posture	Postură neutră	Poziția este simetrică, spatele este drept, trunchiul nu este înclinat, nu prezintă deviații ale capului sau membrelor.
symmetrical_front	Postură frontală și simetrică	Persoana e privită frontal, poziția este simetrică, umerii sunt la același nivel, iar picioarele sunt aliniate.
postural_asymmetry	Postură asimetrică	Poziția este asimetrică, sunt diferențe între partea stângă și dreaptă a corpului. Pot apărea diferențe între umeri sau între picioare (de exemplu: picior peste picior).
stooped_posture	Postură cocoșată	Partea superioară a corpului este aplecată în față, spatele este curbat, iar uneori capul este proiectat înainte.
slouched_posture	Postură relaxată (în exces)	Poziția conferă impresia că tonusul muscular este absent, întregul corp pare a nu fi susținut de mușchi.
undetected	Nedetectat	Postura nu poate fi clasificată corect din cauza detectiei incomplete a punctelor esențiale.

Fiecare imagine din setul de date a fost asociată cu o singură etichetă corespunzătoare posturii dominante identificate. Această etichetă a fost atribuită pe baza regulilor care prelucrează punctele anatomici detectate. Câteva imagini au fost clasificate ca „undetected” din cauza unor puncte lipsă. Acestea prezentau o calitate slabă sau erau prea intunecate, iar ulterior au fost revizuite manual și asociate cu o etichetă.

Imaginiile au fost grupate în categorii pe baza distanțelor și a unghiurilor dintre puncte. Pentru pozele în care s-au detectat puncte atât de pe partea stângă a corpului cât și de pe partea dreaptă, s-a ales, în funcție de adâncime(z), prelucrarea punctelor de pe partea mai apropiată de cameră. Punctele cheie folosite pentru întreg progresul de etichetare sunt: umeri(SHOULDER), șolduri(HIP), genunchi(KNEE), urechi(EAR), nas(NOSE). Dacă o zonă a corpului nu este vizibilă, iar indicatorul acesteia este absent, regula nu este aplicată pentru imaginea respectivă. Aplicarea acestor criterii duce la o etichetare automată a imaginilor și asigură coerentă asocierea dintre imagini și posturile identificate.

În figurile următoare, se prezintă o comparație vizuală între posturile anterior menționate, pentru înțelegerea diferențelor dintre acestea. În prima imagine (4.17) sunt prezentate problemele care sunt văzute din lateral, iar în cea de-a doua imagine (4.18), sunt creionate inconvenientele care sunt observabile dintr-o poziție frontală.

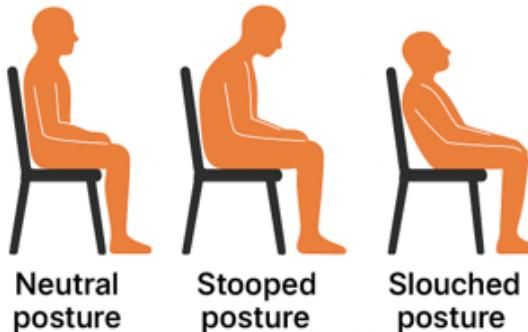


Figura 4.17: Reprezentarea vizuală a posturilor analizate din lateral

Sursă: <https://depositphotos.com/ro/illustrations/people-standing-at-desk.html>
(Accesată în 25-04-2025)

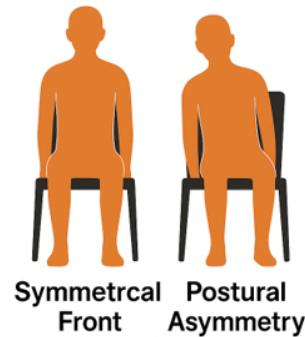


Figura 4.18: Reprezentarea vizuală a posturilor analizate din partea frontală

Sursă: <https://depositphotos.com/ro/illustrations/people-standing-at-desk.html>
(Accesată în 25-04-2025)

În Tabelul 4.3 sunt prezentate, pentru fiecare etichetă, regiunile analizate și punctele implicate. Pragurile au fost stabilite prin testarea repetată și vizualizarea rezultatelor, până când erorile au fost mici, aproape insesizabile.

Tabela 4.3: Descrierea etichetelor utilizate pentru clasificarea posturală

Etichetă	Regiune analizată	Puncte implicate
neutral_posture	Întregul trunchi	NOSE EAR SHOULDER HIP KNEE
symmetrical_front postural_asymmetry	Centura scapulară (umeri) și șolduri	SHOULDERS KNEES
stooped_posture	Zona cervicală și toracală	EAR SHOULDER HIP KNEE
slouched_posture	Zona lombară	SHOULDER HIP KNEE

Criteriile utilizate în momentul clasificării respectă recomandările specialiștilor în ergonomie, dar includ și pragurile de abatere permise, care nu influențează rezultatul.

- **Neutral posture:** Dacă unghiul umăr–șold–genunchi este între 80° și 110° , unghiul ureche–umăr–șold este aproximativ unghi alungit și capul nu este proiectat înainte, atunci poziția este corectă.

- **Symmetrical front, Postural asymmetry:** Dacă diferența de înălțime între umeri este > 0.017 sau cea dintre genunchi este > 0.02 , atunci se sesizează înclinări ale corpului sau picioare încrucișate care duc la o poziție incorectă a coloanei vertebrale.
- **Stooped posture:**
Dacă unghiul ureche-umăr-șold este $< 143^\circ$, atunci spatele este cocoșat, problema fiind la nivel de torace.
Dacă unghiul umăr-șold-genunchi este $< 80^\circ$, atunci persoana are trunchiul prea aproape de genunchi, e aplecată, spatele nu e drept.
Dacă distanța dintre nas și umăr este > 0.135 , atunci capul este proiectat înainte.
- **Slouched posture:** Dacă unghiul umăr-șold-genunchi este $> 110^\circ$, atunci persoana este inclinată pe spate, lasă impresia că alunecă de pe scaun.

4.3. Diagrama cazurilor de utilizare (use-case)

Modul în care utilizatorul interacționează cu aplicația SpinalX este evidențiat prin intermediul unei diagrame use-case. Aceasta ilustrează acțiunile principale pe care utilizatorul le poate realiza în aplicație. Diagrama conținează funcționalitățile aplicației din perspectiva utilizatorului.

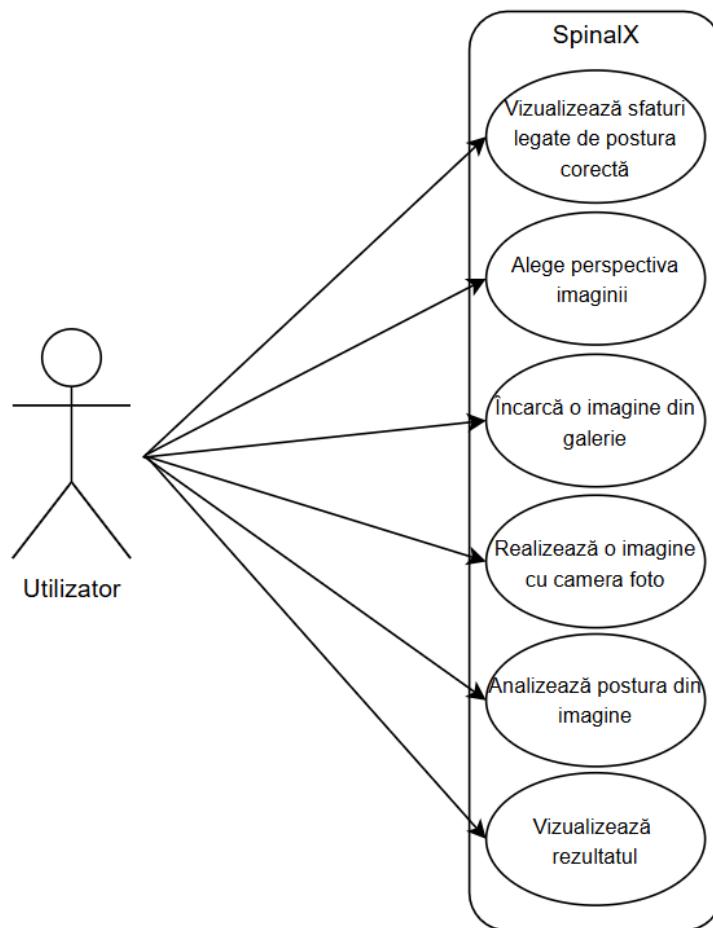


Figura 4.19: Diagrama cazurilor de utilizare

Actor: Utilizator

Actorul reprezintă persoana care folosește aplicația pentru a obține feedback legat de postura corporală. Utilizatorul este singurul actor care interacționează direct cu aplicația.

Cazuri de utilizare:

- Vizualizează sfaturi legate de postura corectă: utilizatorul poate accesa o secțiune ce oferă recomandări privind menținerea unei posturi corecte, bazate pe principii ergonomicice.

- Alege perspectiva imaginii: utilizatorul trebuie să aleagă din ce unghi este realizată imaginea (din față sau lateral). Această alegere este importantă pentru aplicarea regulilor potrivite de clasificare.

- Încarcă o imagine din galerie: utilizatorul poate selecta o imagine deja existentă pe dispozitiv pentru a fi analizată.

- Realizează o imagine cu camera foto: utilizatorul poate captura o imagine nouă, folosind camera dispozitivului.

- Analizează postura din imagine: utilizatorul declanșează procesul de evaluare posturală.

- Vizualizează rezultatul: utilizatorul primește feedback sub formă de etichetă, iar dacă postura este incorectă, primește și o explicație.

4.4. Tehnologii folosite pentru dezvoltarea aplicației mobile

Aplicația mobilă care permite analiza posturii utilizatorului a fost realizată în Android Studio, utilizând limbajul Java. Am optat pentru o arhitectură Android Native, având în vedere avantajele directe oferite de această platformă, precum:

- accesul facil la camera telefonului prin API-uri furnizate de Android

- compatibilitatea nativă cu TensorFlow Lite (framework utilizat pentru rularea locală a modelului de clasificare posturală)

Aplicația este disponibilă exclusiv pentru platforma Android, iar pentru portabilitatea acestuia către alte sisteme de operare (de exemplu: iOS), codul trebuie rescris. În ciuda acestui dezavantaj, alegerea dezvoltării aplicației pentru Android pune accent pe viteză și control. Avantajul oferit de integrarea cu TensorFlow Lite contribuie la reducerea latenței [46].

În cadrul acestui capitol au fost conturate componentele utilizate în realizarea proiectului (tehnologii software, platforme de dezvoltare, metodele de procesare și etichetare, soluții de învățare automată). Fiecare alegere a fost motivată prin prezentarea avantajelor și dezavantajelor care sunt introduse în aplicația de analiză posturală.

Capitolul 5. Proiectare de detaliu și implementare

Acest capitol descrie arhitectura aplicației dezvoltate, modul în care componentele acesteia interacționează, precum și deciziile de implementare. Scopul acestuia este de a facilita înțelegerea structurii aplicației pentru a permite dezvoltarea și întreținerea sa ulterioară. Aplicația mobilă implementată are ca funcționalitate principală analiza posturii utilizatorului pe baza unei imagini preluate de la cameră sau furnizată de utilizator. Clasificarea se realizează local, prin utilizarea modelului antrenat și convertit în format TensorFlow Lite.

5.1. Arhitectura conceptuală a aplicației

Funcționarea aplicației realizate și modul în care componentele sale principale interacționează sunt conturate de diagrama arhitecturii conceptuale (Figura 5.1). Aceasta este structurată în module care colaborează pentru a permite clasificarea posturii utilizatorului.

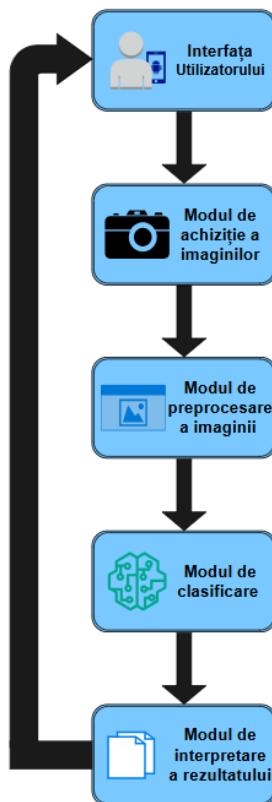


Figura 5.1: Arhitectura conceptuală a aplicației

Pe diagramă se disting următoarele componente principale:

- **Interfața utilizatorului**
Modul în care utilizatorul interacționează cu aplicația.
- **Modulul de achiziție a imaginilor**

Este responsabil de preluarea imaginilor de la utilizator prin intermediul camerei dispozitivului sau din galerie. Acest modul gestionează permisiunile necesare și se asigură că imaginea este salvată local.

- **Modulul de preprocesare a imaginii**

Convertește imaginea într-un format compatibil cu modelul antrenat. Preprocesarea implică redimensionarea, normalizarea și conversia la formatul necesar pentru inferență.

- **Modulul de clasificare**

Aplicația conține două modele de clasificare, unul pentru imagini frontale și unul pentru imagini laterale. Acest modul se ocupă cu returnarea clasei posturale prezisă.

- **Modulul de interpretare a rezultatului**

Pe baza rezultatului returnat de model, aplicația afișează utilizatorului un mesaj care specifică dacă postura este corectă sau ce tip de problemă a fost detectată.

Comunicarea între componente se realizează local, în cadrul aplicației, fără a fi necesară o conexiune la internet. Toate procesările se execută pe dispozitiv. Acest lucru contribuie la reducerea latenței și la protejarea datelor personale. Această arhitectură formată din module permite ca extinderile ulterioare să nu aducă prea multe modificări.

5.2. Modulele aplicației

5.2.1. Interfața utilizatorului

Interfața utilizatorului reprezintă legătura între utilizator și aplicație. Aceasta a fost implementată utilizând framework-ul Android Native (Java). Acest modul include:

- un meniu care permite selectarea perspectivei (frontală sau laterală), în funcție de care va fi ales modelul de clasificare potrivit;
- un buton de achiziție care declanșează solicitarea de permisiune și acces la cameră;
- o zonă unde va fi afișată imaginea analizată;
- un buton care permite selectarea unei imagini din galerie;
- un buton care declanșează analiza imaginii;
- un buton care afișează sfaturi legate de poziția neutră și care prezintă greșeli frecvente;
- mesaje care prezintă rezultatele sau erorile; mesajele de eroare ajută utilizatorii să folosească aplicația.

Designul interfeței a fost realizat respectând recomandările impuse de sistemul Android. S-au respectat următoarele principii de accesibilitate:

- dimensiunea minimă a elementelor interactive (cum ar fi butoanele) a fost păstrată la cel puțin 48x48 dp;
- contrastul dintre text și fundal a fost menținut la un nivel adecvat pentru a asigura vizibilitatea;
- dimensiunea textului nu este prea mică, pentru a nu fi greu de citit;
- elementele au fost poziționate într-o ordine logică, pentru ca aplicația să poată fi ușor de navigat și pentru persoanele care folosesc cititoare de ecran.

5.2.2. Modulul de achiziție a imaginilor

Modulul de achiziție a imaginilor este responsabil de obținerea fotografiei care va fi analizată ulterior de aplicație. Acest modul oferă două scenarii. Utilizatorul poate să facă o poză direct cu camera dispozitivului sau să selecteze o imagine din galeria telefonului.

Pentru utilizarea unei imagini realizate cu camera, se parcurg următoarele etape:

1) Se verifică dacă aplicația are permisiunea de acces la cameră. În caz contrar, utilizatorului i se afișează o solicitare de acordare a acestei permisiuni.

2) Dacă permisiunea este acordată, se creează un fișier temporar în care va fi salvată imaginea.

3) Se deschide aplicația de cameră a telefonului. După ce utilizatorul realizează fotografia, aceasta este salvată și transmisă mai departe în aplicație pentru procesare.

Pentru selectarea unei imagini din galerie, aplicația lansează un picker de fișiere, care permite utilizatorului să aleagă o imagine stocată local. După selectare, imaginea este transmisă spre modulul de procesare.

5.2.3. Modulul de preprocesare a imaginilor

Imaginea furnizată de utilizator ajunge în modulul de preprocesare. Rolul acestui modul este de a pregăti imaginea pentru clasificare, asigurând compatibilitatea cu modelul de inteligență artificială folosit.

În primul rând, imaginile sunt redimensionate (256x256 pixeli), astfel încât să corespundă dimensiunii de intrare a rețelei neuronale DenseNet121.

În al doilea rând, imaginea este transformată într-o matrice de numere care descriu culoarea fiecărui pixel. Această matrice este numită tensor și este forma prin care modelul poate procesa imaginea.

În al treilea rând, valorile pixelilor din imagine sunt normalizate.

În al patrulea rând, ne asigurăm că orientarea imaginii nu este schimbată deoarece o imagine rotită poate genera alte rezultate, făță de imaginea originală.

Preprocesarea este esențială în cadrul acestei aplicații deoarece modelul a fost preantrenat cu imagini modificate în aceeași manieră. O preprocesare diferită a imaginii poate duce la predicții incorecte.

5.2.4. Modulul de clasificare

Modulul de clasificare reprezintă componenta centrală a aplicației. El are rolul de a analiza imaginea preprocesată și de a returna o etichetă care descrie postura detectată. Acest modul este responsabil pentru interacțiunea cu modelul de tip TensorFlow Lite și pentru interpretarea rezultatelor obținute în urma inferenței. Se folosește inferență locală, astfel procesarea este rapidă și nu depinde de conexiunea la internet, dar în acest fel resursele hardware pot influența viteza de răspuns.

În funcție de postura selectată de utilizator (Frontal sau Side), aplicația utilizează modelul corespunzător, alături de fișierul cu etichete asociate. Modelul analizează imaginea și generează un vector de probabilități. Este aleasă eticheta cu scorul cel mai mare, iar probabilitatea este convertită în procent. Au fost antrenate două modele pentru că în cadrul etapei de analiză a erorilor, s-a constatat că unele imagini realizate din profil (lateral) erau clasificate greșit ca fiind posturi corecte frontale. Acest lucru se întâmplă deoarece, în ciuda perspectivei laterale, imaginea conținea suficiente trăsături vizuale (precum umeri aliniati) încât să inducă modelului o clasificare eronată. Pentru a evita aceste confuzii și a obține o performanță mai bună, s-a decis împărțirea setului de date în funcție de perspectivă. Un model este dedicat imaginilor frontale, iar celălalt imaginilor laterale.

5.2.5. Modulul de interpretare a rezultatului

După ce imaginea este clasificată, rezultatul (eticheta și scorul asociat) este prelucrat și afișat într-un mod plăcut pentru utilizator. Modulul are rolul de a transforma datele tehnice într-un mesaj clar și de a furniza utilizatorilor explicații despre postura acestora, dacă este cazul.

Eticheta returnată de model este preluată și procesată pentru a fi afișată într-un format lizibil. În funcție de eticheta obținută, aplicația adaugă o iconă pentru a indica dacă postura este corectă sau incorectă. Prin forma și culoarea acestor iconițe, utilizatorul înțelege direct dacă postura adoptată de el este corectă sau incorectă. Dacă postura este incorectă, aplicația oferă automat un mesaj de avertizare în care precizează ce nu este corect. Etichetele considerate corecte sunt `neutral_posture` și `symmetrical_front`. La apariție, textul rezultatului este animat.

5.3. Diagrama de clase

În Figura 5.2 este prezentată diagrama de clase asociată aplicației mobile dezvoltate. Această diagramă evidențiază structura aplicației din perspectiva programării orientate pe obiect. Sunt evidențiate principalele clase implementate, metodele și atributele relevante, precum și relațiile dintre acestea.

Accentul este pus pe separarea funcționalităților, cum ar fi gestionarea interfeței utilizatorului, clasificarea imaginilor și procesarea acestora. Această organizare facilitează scalabilitatea și întreținerea aplicației.

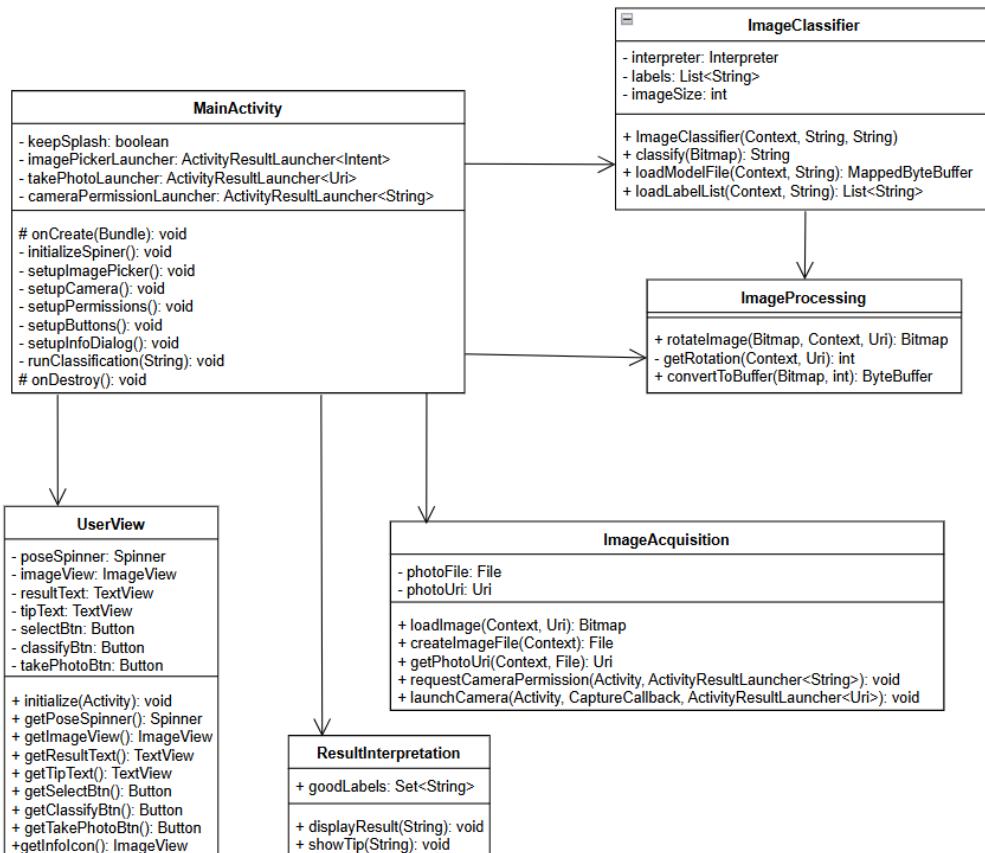


Figura 5.2: Diagrama de clase a aplicației

5.4. Modelul DenseNet121

În aplicație am utilizat o rețea neuronală conoluțională, bazată pe arhitectura DenseNet121, preantrenată pe setul de date ImageNet. Partea superioară a rețelei a fost eliminată pentru a permite adaptarea la sarcina specifică de clasificare a posturii. Prin utilizarea tehnicii de transfer learning, au fost păstrate greutățile straturilor initiale, care au fost preantrenate. Aceste straturi conțin trăsături utile pentru analiza imaginilor, cum ar fi detectarea marginilor, texturilor și formelor. Ulterior, modelul a fost rafinat printr-un proces de fine-tuning care a îmbunătățit performanța.

Pentru a gestiona dezechilibrul existent între clase, s-a analizat distribuția numerică a imaginilor din setul de antrenament. Pe baza acestei distribuții, s-au calculat ponderi folosind funcția *compute_class_weight*. Aceste ponderi reflectă frecvența fiecărei clase și previn modelul din a favoriza clasele dominante în timpul învățării. Pentru clasele cu număr mai mic de exemple, greșelile modelului sunt taxate mai aspru. Aceste ponderi au fost utilizate pentru a contribui la combaterea supraînvățării.

La extragerea trăsăturilor, stratul de clasificare original a fost eliminat. Aceste caracteristici sunt extrase automat din imaginile de intrare prin straturile conoluționale și de pooling. Inițial, straturile modelului DenseNet121 au fost înghețate, ceea ce înseamnă că ponderile acestora nu au fost actualizate în timpul antrenării. Acest lucru a permis păstrarea cunoștințelor. Ulterior, în etapa de fine-tuning, anumite straturi superioare au fost deblocate pentru a permite adaptarea rețelei la setul de date utilizat.

Am utilizat un clasificator personalizat, compus din următoarele straturi:

- GlobalAveragePooling2D: Acest strat a fost utilizat pentru a reduce dimensiunea ieșirii conoluționale și pentru a genera o reprezentare compactă.
- BatchNormalization: Normalizează activările și accelerează antrenamentul.
- GaussianNoise: Adaugă zgomot în faza de antrenare pentru a simula variații naturale.
- Dense(128) + ReLU: Strat dens cu 128 de neuroni și funcție de activare ReLU, utilizat pentru a învăța o reprezentare abstractă a caracteristicilor extrase.
- Regularizare L2 + Dropout: Ambele sunt introduse pentru a preveni supraînvățarea. Dropout dezactivează o parte dintre neuroni în timpul antrenării, forțând modelul să generalizeze.
- Dense + Softmax: Strat final cu un număr de neuroni egal cu numărul de clase, activare softmax, pentru a produce distribuții de probabilitate asupra claselor posibile.

Funcția de pierdere utilizată a fost Categorical Focal Loss, definită manual. Aceasta este o variantă adaptată a funcției Categorical Crossentropy, care nu penalizează uniform toate greșelile de clasificare. Această funcție de pierdere reduce contribuția exemplelor ușor de clasificat și acordă o pondere mai mare exemplelor dificile. Funcția este potrivită pentru situațiile în care datele nu sunt echilibrate, deoarece permite modelului să învețe mai bine clasele minoritare. Pentru optimizare a fost utilizat algoritmul Adam și funcția Cosine Decay.

Procesul de antrenare a fost structurat în două etape. Inițial, modelul a fost antrenat cu baza DenseNet complet înghețată, pentru a păstra caracteristicile învățate de pe ImageNet. Ulterior, în etapa de fine-tuning, s-au deblocat straturile DenseNet de la nivelul 313 în sus (acolo începe ultimul bloc Dense, care are cea mai mare capacitate de a învăța caracteristici specifice problemei abordate). Funcția *EarlyStopping* a fost configurată să monitorizeze evoluția pierderii. Dacă timp de 5 epoci consecutive, nu se

observă o îmbunătățire, antrenamentul se oprea. *ModelCheckpoint* a fost activat pentru a salva greutățile corespunzătoare celei mai bune performanțe, asigurând astfel că se păstrează cea mai bună variantă a modelului. Rata de învățare a fost ajustată dinamic prin intermediul *ReduceLROnPlateau*.

Arhitectura modelelor utilizate în cadrul aplicației este prezentată în Figura 5.3. Aceasta evidențiază atât componenta de extragere a caracteristicilor, cât și clasificatorul personalizat adăugat ulterior. Se observă o separare între partea convoluțională (care generează reprezentări abstrakte ale imaginilor) și clasificatorul format din straturile anterior menționate.

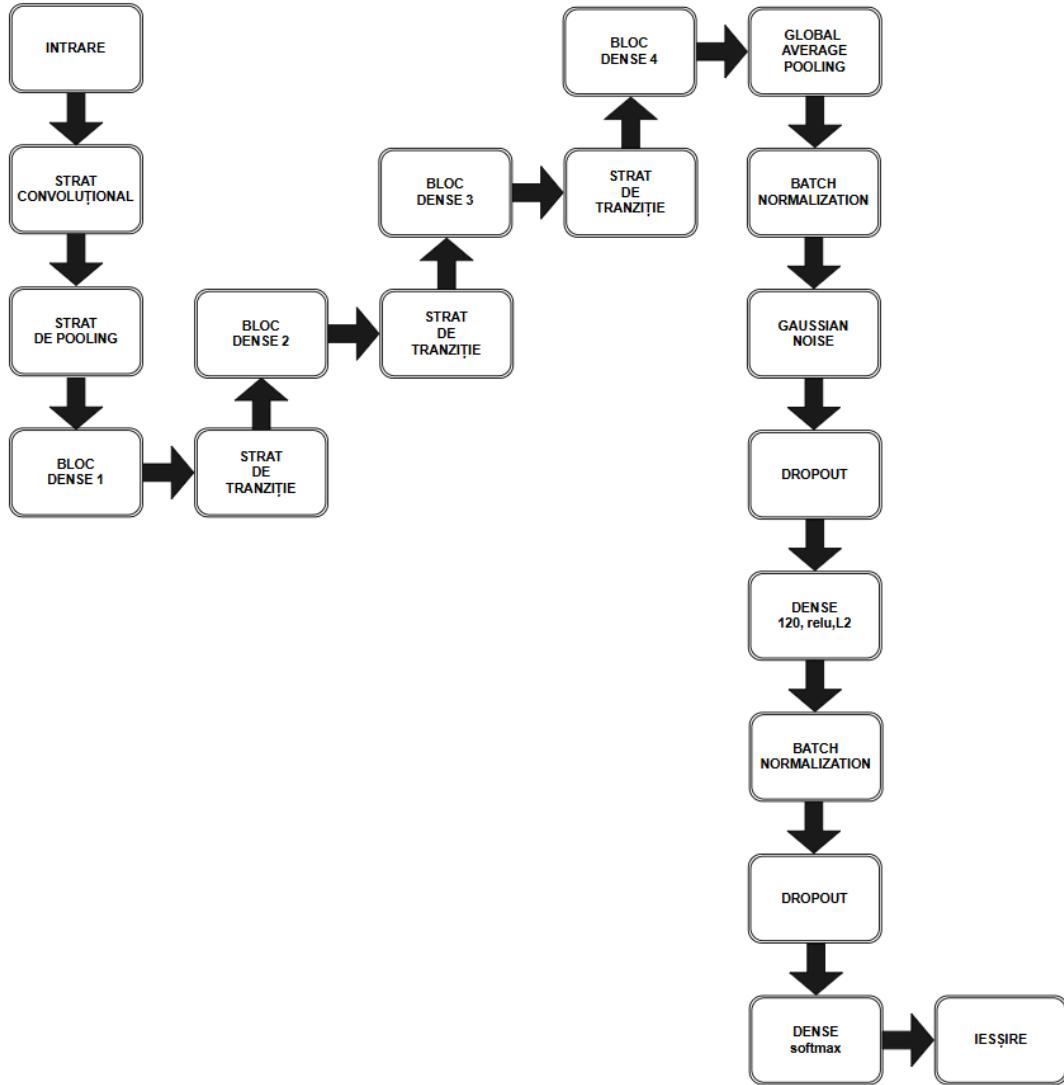


Figura 5.3: Arhitectura modelului

Integrarea modelului în aplicație a fost posibilă după conversia acestuia în formatul optimizat .tflite, utilizând TensorFlow Lite Converter. Deoarece modelul era salvat sub formă de ponderi, a fost nevoie să reconstruim manual arhitectura sa (aceeași arhitectură ca și în timpul antrenării). După reconstrucție, modelul a fost convertit și salvat în formatul compatibil cu TensorFlow Lite, permitând astfel rularea directă pe dispozitive mobile, cu latență redusă, independent de alte infrastructuri (nu este necesară nici măcar conexiunea la internet).

5.5. Diagrama de secvență

Pentru a evidenția modul în care utilizatorul interacționează cu aplicația, am realizat o diagramă de secvență ce ilustrează fluxul principal de date și comenzi (happy flow). Figura 5.4 surprinde pașii parcursi în momentul în care un utilizator dorește să identifice postura dintr-o poză efectuată pe moment. Interacțiunile sunt reprezentate între actor (utilizatorul) și componentele principale ale aplicației.

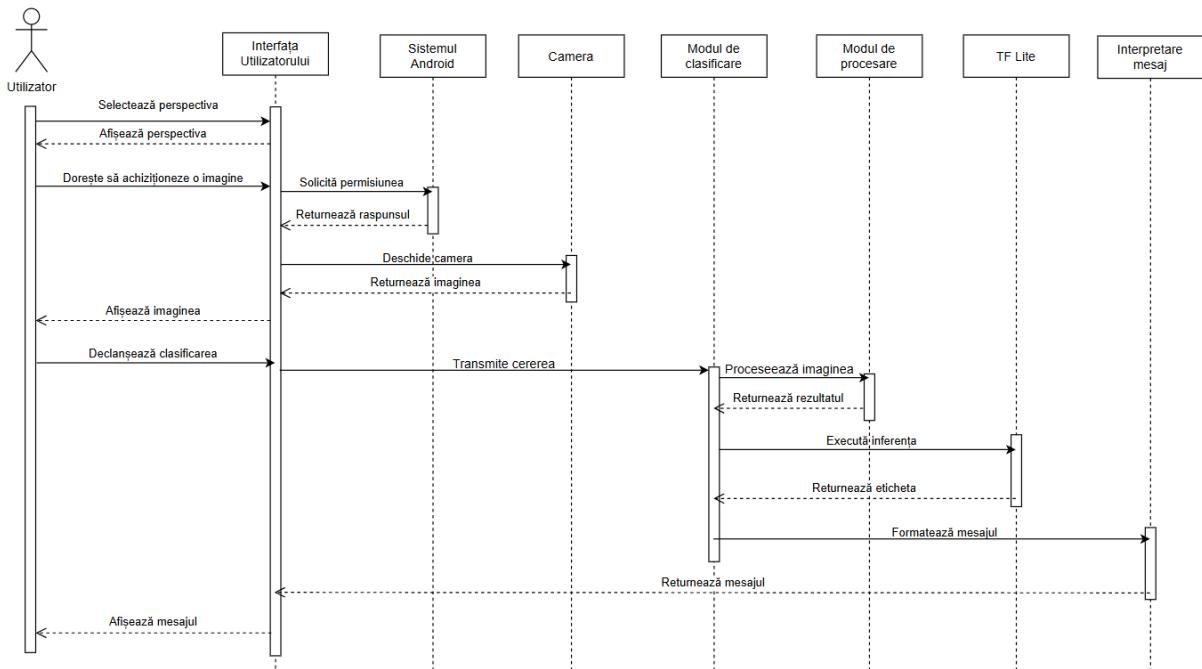


Figura 5.4: Diagrama de secvență

Diagrama conturează următoarele etape:

- 1) Utilizatorul selectează perspectiva dorită (frontală sau laterală);
- 2) Sistemul solicită permisiunea de acces la cameră, dacă este cazul;
- 3) Imaginea este achiziționată prin intermediul camerei;
- 4) Imaginea este transmisă către modulul de clasificare;
- 5) Imaginea este preprocesată și convertită în tensor, ulterior este transmisă către model;
- 6) Eticheta este transmisă spre modulul de interpretare, care construiește un mesaj pentru utilizator;
- 7) Mesajul este afișat pe ecran.

5.6. Diagrama de flux

Diagrama de flux conturează scenariul complet al utilizării aplicației. În aceasta se prezintă modul în care utilizatorul interacționează cu sistemul, conturându-se atât scenariul ideal, cât și situațiile neprevăzute care pot apărea.

Diagrama (Figura 5.5) parcurge următorii pași:

- 1) Selectarea perspectivei (frontală/laterală);
- 2) Achiziționarea imaginii (din galerie sau prin cameră);
- 3) Gestionarea permisiunilor pentru cameră;
- 4) Verificarea furnizării tuturor informațiilor necesare;

- 5) Clasificarea imaginii;
- 6) Vizualizarea rezultatului final sub formă de mesaj.

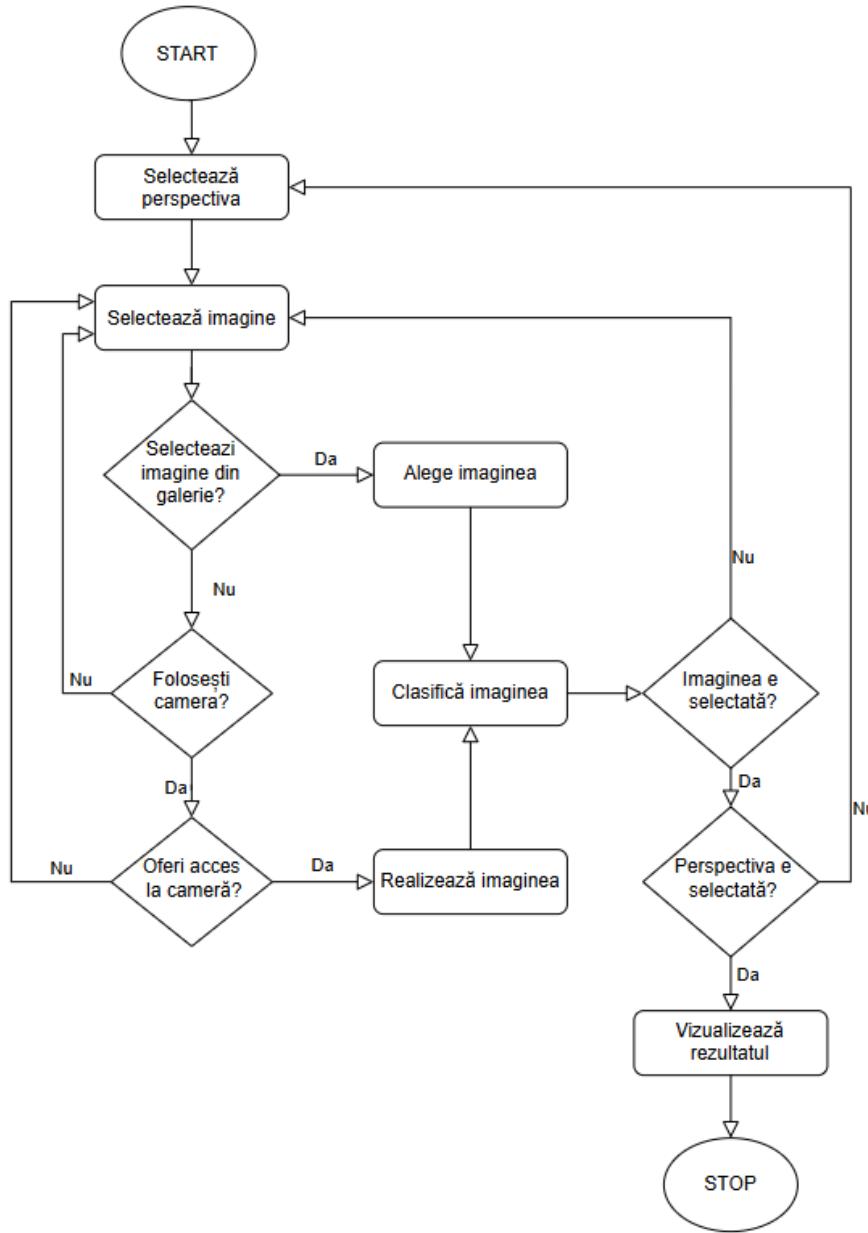


Figura 5.5: Diagrama de flux

Prin arhitectura propusă, aplicația reușește să îndeplinească scopul propus: evaluarea automată a posturii utilizatorului. Modelul de clasificare folosit oferă un echilibru bun între precizie și rapiditate. Decizia de a integra modelul în aplicație oferă un mediu sigur, care permite analiza posturii fără a fi nevoie de conexiune la internet.

5.7. Preprocesarea datelor

În procesul inițial de antrenare, s-a observat că modelul supraînvață deoarece acuratețea era foarte ridicată pe datele de antrenament, dar performanța pe setul de test era scăzută. După analizarea manuală a setului de date, s-a constatat că o parte

importantă a imaginilor erau duplicate sau imaginile erau extrem de similare vizual, ceea ce a dus la o generalizare deficitară.

Acest comportament a fost evidențiat prin analiza curbelor de învățare, care reflectă evoluția funcțiilor de acuratețe și pierdere. În Figura 5.6, se observă o creștere constantă a performanței pe setul de antrenament, iar pe setul de validare, după un anumit număr de epoci, performanța se degradează și stagniază. Acuratețea pe setul de antrenament continuă să crească (ajunge la 98%), ceea ce sugerează că modelul memorizează datele din antrenament. În ceea ce privește funcția de pierdere (loss), aceasta scade constant pentru setul de antrenament, dar crește ușor pentru validare, indicând că modelul începe să piardă din capacitatea de a face predicții corecte pe exemple noi.

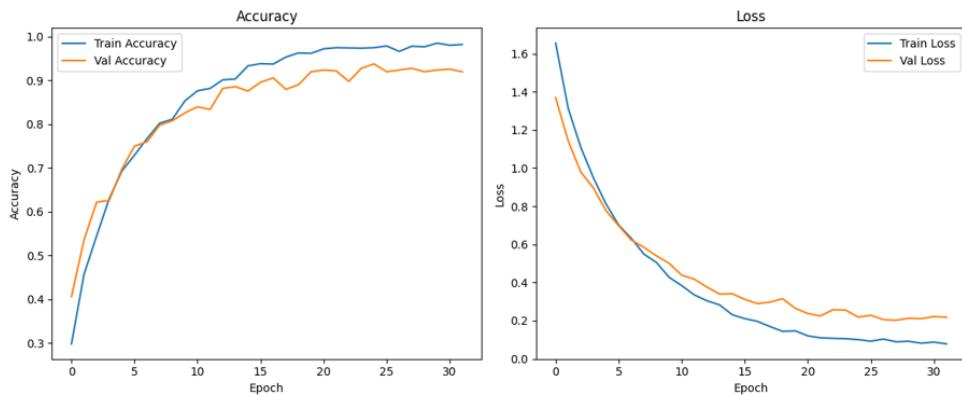


Figura 5.6: Evoluția acurateței și a funcției de pierdere înainte de curățarea setului de date

Suspiciunea supraînvățării a fost confirmată și prin matricea de confuzie obținută în urma evaluării pe setul de test, care indică faptul că modelul clasifică majoritatea imaginilor într-o singură clasă, indiferent de eticheta reală (Figura 5.7). Restul claselor sunt ignorate, deci modelul nu generalizează. Aceasta a învățat să recunoască doar tiparul cel mai frecvent întâlnit în setul de date. Acest comportament a condus la o acuratețe pe setul de test de doar 20%, cu o pierdere ridicată. Duplicatele și imaginile similare au condus la apariția unui model supraantrenat.

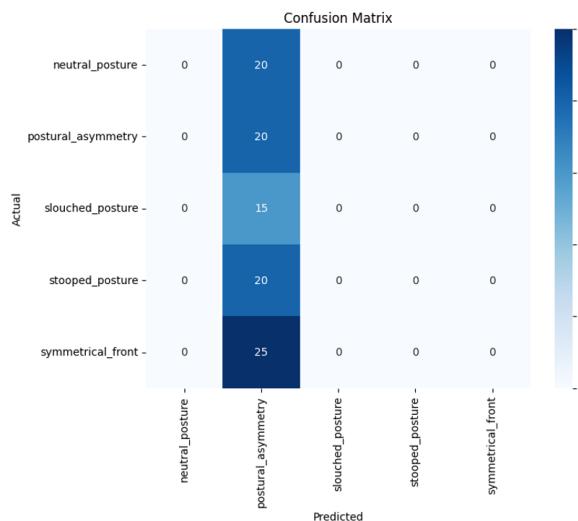


Figura 5.7: Performanța modelului pe setul de test înainte de curățarea setului de date

Pentru a identifica imaginile complet identice din setul de date, s-a aplicat o metodă de comparare a conținutului digital al fișierelor. Fiecare imagine a fost analizată la nivel binar, iar fișierele care aveau conținutul la fel au fost grupate. Din fiecare grup de duplicate, s-a păstrat o singură imagine, iar celelalte au fost eliminate. Acest proces a permis înălțarea imaginilor care fuseseră salvate de mai multe ori. Apariția acestor duplicate a fost datorată combinării seturilor de date prezentate anterior. Unele imagini făceau parte din mai multe seturi. Eliminarea acestor duplicate a contribuit la echilibrarea datelor utilizate pentru antrenarea modelului.

Imagini aproape identice din punct de vedere vizual erau ușor de confundat cu duplicatele exacte. În Figura 5.8, observăm două imagini care au rămas în setul de date, după ce duplicatele au fost eliminate. Astfel, s-a realizat că e nevoie și de o metodă care să elimine imaginile care sunt similare.



Figura 5.8: Exemple de imagini aproape identice, păstrate în setul de date după ce duplicatele au fost eliminate

După eliminarea dupliilor exacte, s-a aplicat o metodă pentru identificarea imaginilor aproape identice din punct de vedere vizual. Această metodă, cunoscută sub denumirea de perceptual hashing, constă în generarea unei reprezentări numerice care surprinde aspecte generale ale imaginii (forme, lumină, distribuția culorilor). Dacă două imagini aveau reprezentări foarte apropiate, acestea erau considerate similare, iar una dintre ele era eliminată. Această abordare este utilă în cazul imaginilor care conțin aceeași informații vizuale.

Întreaga etapă de curățare a dus la eliminarea a 1400 de imagini din setul de date și a fost esențială în procesul de preprocesare, contribuind la combaterea supraînvățării și la creșterea capacitatei de generalizare a modelului.

Imaginiile rămase au fost pregătite pentru antrenarea modelului prin un proces de preprocesare specific rețelei DenseNet121. Fiecare imagine a fost redimensionată la 256x256 pixeli, iar valorile pixelilor au fost normalizate utilizând funcția *preprocess_input*. Această funcție transformă imaginea într-un format compatibil cu rețelele DenseNet preantrenate.

Pentru a reduce riscul de supraînvățare, imaginile din setul de antrenament au fost augmentate în mod aleatoriu prin aplicarea unor transformări precum: translatare pe orizontală și verticală, ajustări de luminozitate, efecte de forfecare, zoom și oglindire orizontală. Toate operațiile menționate anterior au fost aplicate moderat. Nu s-au aplicat transformări care ar putea distorsiona postura. Aceste augmentări au permis modelului să învețe trăsături mai generale. Pentru seturile de validare și testare, nu s-au aplicat augmentări, ci doar redimensionarea și normalizarea prin funcția menționată anterior.

Capitolul 6. Testare și validare

Testarea și validarea sunt etapele care asigură o bună funcționare a aplicației de clasificare posturală. Procesul a urmărit etichetarea corectă a imaginilor din setul de date, validarea predicțiilor generate de model și comportamentul aplicației în scenarii reale de utilizare.

6.1. Validarea etichetării setului de date

În prima etapă a dezvoltării acestui proiect s-a realizat pregătirea datelor. Imaginilor din setul de date li s-a asociat un identificator printr-un mecanism automat de etichetare. Validarea acestor etichete a urmărit compararea rezultatelor generate automat cu etichetele stabilite manual.

Pe parcursul testării, au fost ajustate pragurile de abatere de la postura ideală folosite în regulile de clasificare, pentru a obține o separare cât mai clară între clase. În plus, au fost analizate manual imagini pentru care sistemul nu a detectat corect punctele anatomicice (fie erau lipsă, fie erau prea aproape de margini) și au fost etichetate automat ca *undetected*. Există doar câteva astfel de exemple izolate cărora nu li s-a putut atribui automat o etichetă.

În majoritatea cazurilor, testarea a demonstrat că etichetele erau corect atribuite, însă au existat și situații ambigue. De exemplu, unele imagini laterale în care erau vizibile și trăsături frontale duceau la confuzii deoarece li se potriveau două etichete. Acestea puteau fi încadrate atât la postură neutră privită din lateral, cât și la postură corectă privită frontal. Aceste constatări au contribuit la îmbunătățirea procesului de etichetare și la justificarea antrenării unor modele separate pentru imagini frontale și cele laterale.

Pentru a susține validarea etichetării, în continuare sunt prezentate câteva exemple vizuale reprezentative pentru fiecare clasă. Acestea reflectă faptul că procesul de etichetare s-a realizat cu succes. Imaginile fac parte din seturile de date de care am vorbit în capitolele anterioare, iar sursele acestora au fost deja precizate [41, 42, 40, 43].



Figura 6.1: Imagine reprezentativă pentru eticheta slouched_posture



Figura 6.2: Imagine reprezentativă pentru eticheta stooped_posture



Figura 6.3: Imagine reprezentativă pentru eticheta neutral_posture



Figura 6.4: Imagine reprezentativă pentru eticheta postural_asymmetry



Figura 6.5: Imagine reprezentativă pentru eticheta symmetrical_front

6.2. Evaluarea performanței modelului

Evaluarea performanței modelului de analiză a posturii s-a bazat pe utilizarea seturilor de date de test. Aceste seturi sunt alcătuite din imagini noi, pe care modelul nu le-a întâlnit în timpul antrenării. Evaluarea nu a urmărit doar precizia globală a modelului, ci și capacitatea acestuia de a distinge corect între etichetele definite.

Pentru inferență, s-a aplicat și o tehnică de tip Test-Time Augmentation (TTA), în care fiecare imagine de test a fost supusă la câteva transformări minore, iar predicția finală a fost obținută prin agregarea celorlalte predicții.

Evaluarea s-a realizat utilizând următoarele metri: acuratețe, precizie, recall, scor F1. În continuare, este prezentat un tabel comparativ care evidențiază rezultatele obținute prin antrenarea mai multor modele, cu scopul de a identifica arhitectura cea mai potrivită pentru sarcina de clasificare a posturii. Fiecare model a fost antrenat și evaluat pe același set de date, utilizând aceleași condiții de antrenare și aceleași metri de evaluare. Tabelul 6.1 prezintă rezultatele obținute.

Se poate observa că DenseNet121 a obținut cele mai bune rezultate pe toate metriile analizate, inclusiv o acuratețe de 82.1% și cel mai mic scor al funcției de pierdere (0.18), ceea ce indică o capacitate bună de generalizare. De asemenea, valorile ridicate ale preciziei și scorului F1 (ambele 83%) sugerează o performanță echilibrată între clase. Prin comparație, celelalte modele au înregistrat scoruri vizibil mai scăzute, ceea ce indică o dificultate în identificarea corectă a claselor. Având în vedere aceste rezultate, DenseNet121 a fost utilizat în aplicația mobilă.

Rezultatele corespund evaluării modelelor de clasificare posturală aplicate pe imagini cu perspectiva laterală. Un proces similar a fost realizat pentru imaginile cu perspectiva frontală, folosind aceleași arhitecturi și același set de date. Si în acest caz, modelul DenseNet121 a demonstrat cele mai bune performanțe (acuratețe 80% și pierdere 0.14).

Tabela 6.1: Compararea performanței modelelor de clasificare posturală

Model	Acuratețe	Pierdere	Precizie (macro)	Recall (macro)	F1 (macro)
DenseNet121	82.1%	0.18	83%	85%	83%
EfficientNetV2B0	60.5%	1.09	59%	59%	59%
EfficientNet3	64.2%	1.10	66%	67%	66%
EfficientNetB0	59.5%	1.12	60%	58%	58%
MobileNetV2	78.9%	0.59	80%	75%	77%
ResNet50	61.9 %	1.08	62%	55%	57%

Graficele din Figura 6.6 prezintă performanța modelului DenseNet121 antrenat pe setul de date frontal, evaluat pe parcursul a 40 de epoci. Sunt analizate două aspecte esențiale: acuratețea și pierderea, atât pentru setul de antrenament (train), cât și pentru setul de validare (val).

Acuratețea și pierderea sunt complementare. Acuratețea măsoară proporția de predicții corecte, dar nu oferă detalii despre nivelul de încredere al modelului în predicții greșite. Pierderea penalizează mai puternic predicțiile greșite făcute cu mare încredere. Astfel, dacă acuratețea crește și pierderea scade, înseamnă că modelul face predicții tot mai corecte și cu o certitudine mai mare.

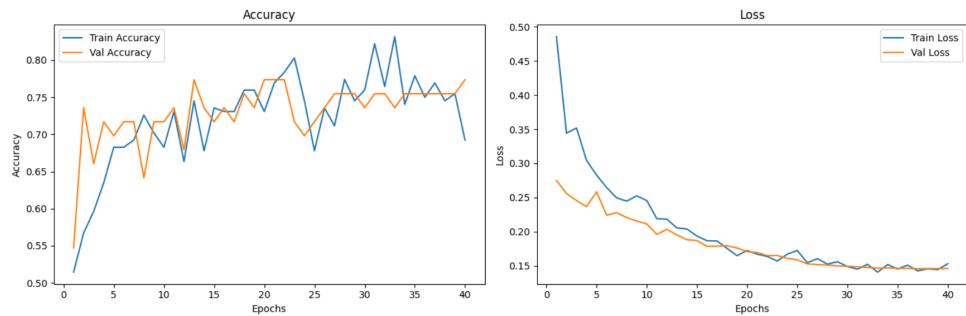


Figura 6.6: Evoluția acurateței și pierderii în timpul antrenării modelului DenseNet121 (perspectivă frontală)

Interpretarea curbei de acuratețe:

- Curba albastră („Train Accuracy”) arată o creștere constantă a acurateței în timpul antrenării, pornind de la aproximativ 52% și ajungând la un maxim de peste 80%.
- Curba portocalie („Val Accuracy”) indică o performanță stabilă pe setul de validare, cu valori cuprinse între 65% și 77%, fără scăderi semnificative.
- Acest comportament sugerează că modelul reușește să generalizeze bine și nu este afectat de supraînvățare.

Interpretarea curbei de pierdere:

- Pierderea în antrenare („Train Loss”) scade de la aproximativ 0.48 la sub 0.15, arătând că modelul a învățat din datele de antrenament.
- Pierderea pe validare („Val Loss”) scade constant și se stabilizează în jurul valorii de 0.15, fără variații brusă.
- Aceste valori scăzute confirmă că modelul este stabil.

Pentru a evalua performanța modelului DenseNet121 antrenat pe perspectiva laterală, a fost generată o matrice de confuzie, prezentată în Figura 6.7. Aceasta evidențiază distribuția predicțiilor în raport cu etichetele reale și oferă o imagine clară asupra cazurilor corect și incorrect clasificate.

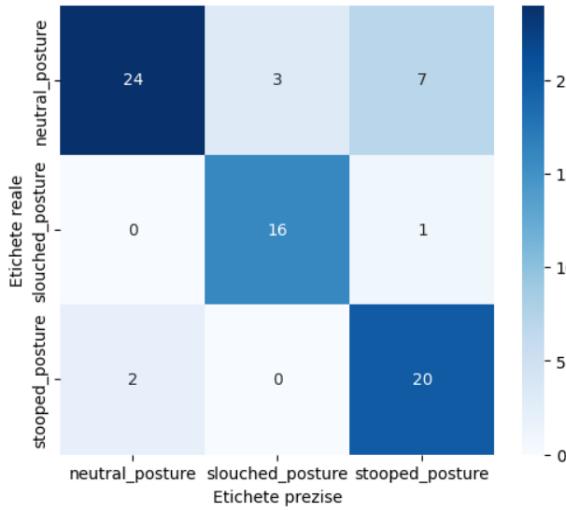


Figura 6.7: Matricea de confuzie (perspectiva frontală)

Observăm că majoritatea exemplelor din clasa neutral_posture au fost clasificate corect, însă modelul a confundat 7 imagini cu stooped_posture și 3 cu slouched_posture. Acest lucru poate indica faptul că în anumite poziții, postura neutră poate semăna vizual cu o ușoară înclinare a trunchiului sau cu o curbare a spotelui. Pentru slouched_posture, modelul a obținut rezultate excelente, clasificând corect 16 din cele 17 imagini, ceea ce reflectă o delimitare bună a trăsăturilor acestei clase. În cazul clasei stooped_posture, 20 din 22 de imagini au fost corect identificate. Această analiză poate să ne trimită spre ideea că posturile analizate sunt apropiate vizual, iar din acest motiv apar mici confuzii.

Modelul DenseNet121 a oferit cele mai bune rezultate, iar din acest motiv a fost ales pentru integrarea în aplicație. Evoluția constantă a acurateței și pierderii indică o antrenare stabilă, iar matricea de confuzie confirmă o bună capacitate de diferențiere între etichetele posturale. Rezultatele obținute susțin alegerea modelului ca soluție eficientă pentru analiza automată a posturii.

6.3. Testarea aplicației mobile

Evaluarea comportamentului aplicației mobile în condiții reale de utilizare s-a realizat printr-o serie de etape de testare, care au vizat atât verificarea componentelor individuale, cât și coerența fluxului general de utilizare.

6.3.1. Testare pe emulatoare

Într-o primă fază, aplicația a fost testată pe emulatoarele Android disponibile în Android Studio. Aceste emulatoare imită comportamentul unor dispozitive reale care au diverse configurații hardware și versiuni de sistem de operare. În Figura 6.8 este ilustrată interfața unui emulator. În partea din dreapta, jos, se observă aplicația dezvoltată, SpinalX.

Testarea pe emulator a avut multiple avantaje, precum:

- Rapiditatea: Codul sursă putea fi modificat și rulat imediat, fără a necesita transferul aplicației pe un dispozitiv fizic.
- Diversitate de dispozitive: Am avut oportunitatea de a testa aplicația pe mai multe tipuri de telefoane, cu dimensiuni de ecran și versiuni Android diferite.

Funcționalitățile principale testate pe emulator au inclus:

- afișarea corectă a interfeței grafice pe diferite ecrane;
- tratarea erorilor (de exemplu, fără imagine selectată);
- rularea fluxului de preprocesare și clasificare a imaginilor capturează;
- afișarea corectă a rezultatelor.



Figura 6.8: Interfața unui emulator

Această etapă a permis identificarea unor aspecte legate de dimensiunea butoanelor sau afișarea mesajelor în cazul ecranelor mici. După ajustări, interfața a fost optimizată pentru a respecta ghidurile de design Android, inclusiv reguli legate de dimensiunea minimă a zonelor tactile, contrast și vizibilitate.

Totuși, emulatorul nu oferă suport pentru accesul la camera fizică, ceea ce limitează testarea aplicației propuse.

6.3.2. Testare pe dispozitive reale

Pentru a evalua complet funcționalitatea aplicației, aceasta a fost instalată pe un telefon fizic, Samsung Galaxy S23 FE cu sistem de operare Android 14.

Testele au vizat atât stabilitatea aplicației, cât și corectitudinea funcționalităților implementate. Aplicația a funcționat stabil, fără erori, în toate scenariile testate. Au fost verificate următoarele:

- Deschiderea camerei pentru achiziția imaginii în timp real, fără blocaje;
- Selectarea imaginilor din galerie și afișarea corectă a acestora în interfață;
- Clasificarea posturii în funcție de perspectiva selectată;
- Afișarea feedbackului printr-un mesaj vizibil și ușor de înțeles de către utilizator.

În cadrul testării pe dispozitive reale, s-a verificat dacă procesul de clasificare a posturii are loc aproape instantaneu, fără întârzieri pe care utilizatorul să le percepă. Acest aspect este esențial în contextul utilizatorilor moderni, care se așteaptă la un răspuns imediat din partea aplicațiilor. Conform cercetărilor în domeniul interacțiunii om-calculator, o reacție percepă ca fiind instantanee trebuie să aibă loc în sub 0.1 secunde. Întârzierile mai mari de 1 secundă pot afecta experiența utilizatorului [47].

6.3.3. Testare în condiții reale de utilizare

Validarea experienței de utilizare și a funcționalității aplicației în condiții reale a fost realizată prin testarea acesteia de către un grup de 11 utilizatori (4 femei și 7 bărbați),

cu vârste cuprinse între 18 și 30 de ani. Testarea s-a desfășurat pe dispozitivele personale ale participanților.

Participanții au fost rugați să își analizeze postura atât folosind o imagine din galerie, cât și efectuând o poză cu ajutorul camerei. Niciun utilizator nu a întâmpinat dificultăți sau blocaje. Una dintre cele 11 persoane consideră că adoptă o postură mai corectă decât prezice aplicația și crede că postura corectă este prea strictă. Utilizatorii au menționat următoarele:

- Aplicația este ușor de folosit, chiar și fără explicații suplimentare;
- Clasificarea se realizează aproape instant, nu e nevoie să aștepți;
- Rezultatul afișat este ușor de înțeles;
- Sfaturile legate de cum să adoptă o poziție corectă și ce să încerci să eviți sunt clare și bine ilustrate în exemplul din aplicație.

În Figura 6.9 este ilustrat un exemplu de utilizare completă a aplicației, în care un utilizator parcurge cu succes toți pașii (de la selectarea perspectivei și încărcarea imaginii, până la obținerea rezultatului final).

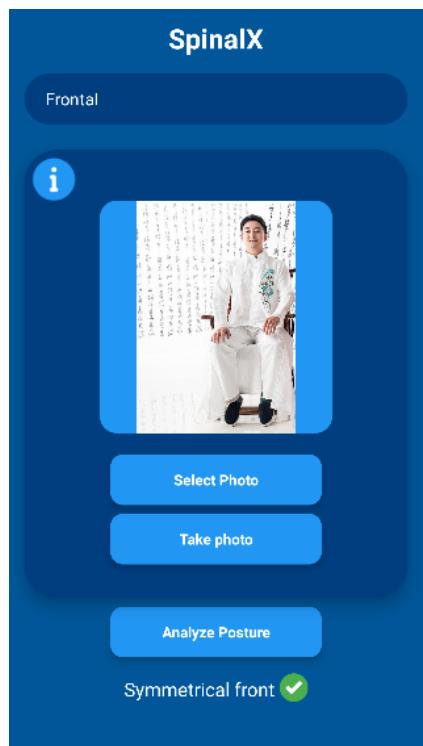


Figura 6.9: Exemplu de utilizare a aplicației

Testarea în condiții reale a evidențiat stabilitatea aplicației și nivelul ridicat de accesibilitate. Viteza cu care este afișat rezultatul a fost apreciată de utilizatori care tind să renunțe rapid dacă o aplicație nu oferă răspunsuri imediate. Astfel, obiectivul de a păstra o latență scăzută a fost realizat.

Prin urmare, testarea și validarea aplicației au confirmat atât faptul că sistemul funcționează corect, cât și performanța modelului de clasificare. Etichetarea automată a imaginilor a fost verificată în detaliu. Evaluările realizate pe emulatoare, dispozitive reale și prin interacțiunea cu utilizatori reali au evidențiat o experiență de utilizare fără erori, cu timpi de răspuns foarte mici și mesaje clare. Aplicația oferă astfel o soluție fiabilă pentru analiza posturii.

Capitolul 7. Manual de instalare și utilizare

Acest capitol oferă ghidul complet pentru instalarea și utilizarea aplicației mobile destinață analizării posturii. Informațiile sunt prezentate pentru utilizatorul final, care nu necesită cunoștințe tehnice. Scopul capitolului este de a se asigura că orice persoană poate să instaleze aplicația pe propriul dispozitiv Android și să o utilizeze fără să întâmpine dificultăți.

Instrucțiunile sunt structurate în două părți:

- 1) Pași de instalare și cerințe minime;
- 2) Explicații legate de utilizarea aplicației.

Folosind acest manual, orice utilizator va putea să își analizeze propria postură urmărind doar câțiva pași simpli.

7.1. Instalarea aplicației pe un dispozitiv mobil

Utilizarea aplicației de analiză a posturii este posibilă prin instalarea acesteia pe un telefon mobil cu sistem de operare Android. Procesul de instalare este rapid, simplu.

Cerințe minime pe care dispozitivul trebuie să le îndeplinească:

- Sistem de operare: Android 8.0 (Oreo) sau versiune mai nouă;
- Memorie RAM: minim 2 GB;
- Minim 150 MB spațiu liber pentru instalare și rulare fluentă;
- Cameră foto (optional);
- Conexiune la internet (doar pentru descărcare).

Aplicația poate fi instalată direct pe dispozitivul mobil Android prin intermediul unui fișier. Aplicația nu colectează date personale, iar fișierul .apk provine dintr-o sursă verificată și sigură. Procesul de instalare este local și nu presupune conectarea la servere externe, astfel încât utilizatorul nu trebuie să-și facă griji în privința securității aplicației. Pașii necesari pentru instalare sunt:

- Transferați fișierul .apk pe telefon (cu ajutorul cablului USB, prin email sau alte modalități de transfer);
- Mergeți în aplicația Fișiere/File Manager și localizați fișierul descărcat. Apăsați pe acesta pentru a începe instalarea.
- Dacă este prima dată când instalați aplicații din surse necunoscute, va fi afișat un mesaj care solicită accordarea permisiunii pentru această acțiune. Activăți opțiunea: *Permite instalarea din această sursă*.
- După instalare, aplicația va apărea pe ecranul principal sau în meniul cu aplicații.

Dacă instalarea nu reușește, este posibil ca dispozitivul să nu permită instalarea aplicațiilor din surse externe. În acest caz, utilizatorul trebuie să acceseze Setări -> Securitate și să activeze opțiunea *Permite instalarea din surse necunoscute*. De asemenea, se recomandă verificarea compatibilității versiunii Android și asigurarea unui spațiu liber de cel puțin 150 MB. După ce fișierul a fost instalat cu succes, utilizatorul poate lansa aplicația prin simpla apăsare pe pictogramă. Pentru utilizarea aplicației, utilizatorii trebuie să respecte pașii din subcapitolul următor.

7.2. Utilizarea aplicației

Acest subcapitol descrie pașii necesari pentru utilizarea corectă a aplicației mobile de analiză posturală. Interfața a fost proiectată punând accent pe simplitate, astfel încât aplicația să poată fi utilizată de toate categoriile de utilizatori.

7.2.1. Lansarea aplicației

După instalare, aplicația poate fi accesată direct din meniul principal al dispozitivului Android. La prima rulare, utilizatorul va fi informat cu privire la permisiunile necesare (de exemplu: accesul la cameră), pe care trebuie să le accepte pentru funcționarea corectă. În Figura 7.1 se prezintă modul în care este cerut accesul la utilizarea camerei foto. Accesul la cameră nu este obligatoriu, este la latitudinea utilizatorului dacă dorește să utilizeze camera sau să folosească doar poze din galerie.

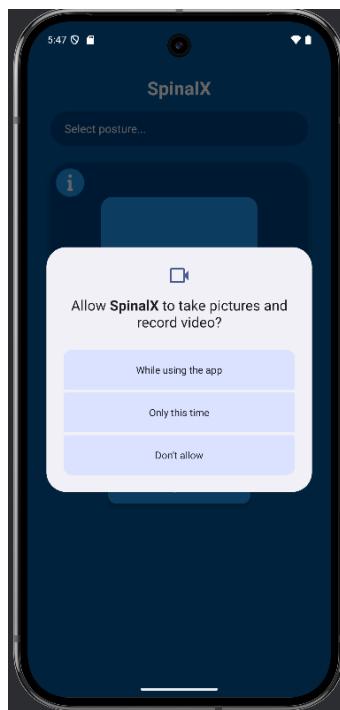


Figura 7.1: Notificarea prin care se cere accesul la cameră

În cazul în care mesajul privind permisiunile nu apare sau utilizatorul refuză accesul din greșeală, este necesar ca acestea să fie activate manual din setările dispozitivului. Se poate face astfel:

- 1) Accesați Setări -> Aplicații -> SpinalX -> Permișuni;
- 2) Activăți accesul la componentele necesare (Cameră, Galerie).

7.2.2. Interfața aplicației

La deschiderea aplicației, utilizatorul vede logo-ul aplicației, iar apoi se face o tranziție către meniul principal. În Figura 7.2 se prezintă meniul principal, care are următoarele elemente:

- Titlul aplicației: afișat în partea de sus;
- Selectorul de perspectivă : un meniu de tip dropdown pentru alegerea între perspectivă frontală (frontal) și perspectivă laterală (side);

- Butonul *Informatii* (i): deschide sfaturi cu privire la o postură corectă;
- Butonul *Selectează imaginea* (Select photo): permite alegerea unei imagini din galerie;
- Butonul *Realizează o fotografie* (Take photo): deschide camera pentru a achiziționa o imagine nouă;
- Butonul *Analyzează postura* (Analyze Posture): conduce la generarea rezultatului.

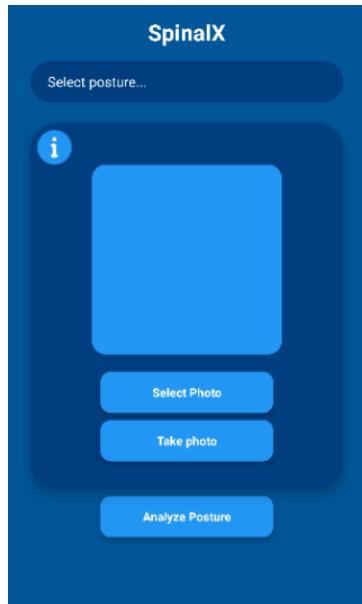


Figura 7.2: Meniul principal al aplicației

7.2.3. Fluxul aplicației

Utilizatorii care doresc să își analizeze postura trebuie să parcurgă următorii pași:

1) Utilizatorul trebuie să aleagă perspectiva din care vrea să își analizeze postura;
2.1) Utilizatorul apasă pe butonul *Selectează imaginea* și ulterior alege o poză din galerie.

2.2) Utilizatorul apasă pe butonul *Realizează o fotografie*, ulterior, se va deschide camera și imaginea va fi captată.

Această imagine va fi afișată pe ecran, în spațiul dedicat (deasupra butoanelor). Dacă achiziționarea imaginii ridică probleme din cauza permisiunilor, trebuie să se consulte subsecțiunea 7.2.1.

3) Utilizatorul apasă pe butonul *Analizare postură* și primește instant un feedback. Dacă postura este greșită, primește și o explicație.

În Figura 7.3 se ilustrează modul în care aplicația trebuie să reacționeze dacă pașii anteriori au fost respectați. Dacă utilizatorul dorește analizarea unei alte imagini, se reia procesul descris mai sus. Dacă nu sunt furnizate toate informațiile necesare, aplicația afișează mesaje de avertismență, precum cele din Figura 7.4.

După analiză, aplicația afișează un mesaj însotit de o pictogramă sugestivă, care ajută utilizatorul să înțeleagă rapid dacă postura sa este corectă sau nu. Dacă postura este corectă, se afișează o bifă verde. În caz contrar, aplicația indică ce problemă a fost detectată și afișează un semn roșu de avertismență. Acest sistem facilitează interpretarea rapidă a mesajului.

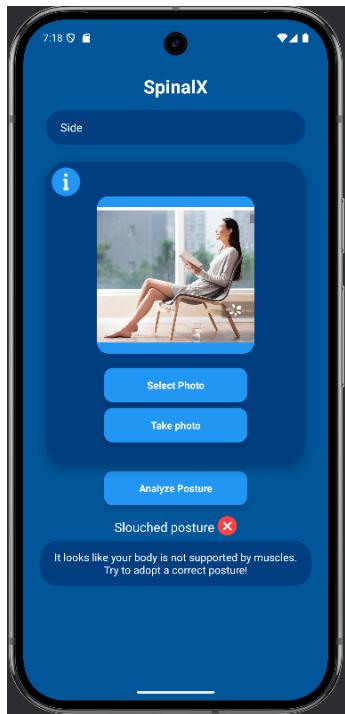


Figura 7.3: Exemplu de utilizare

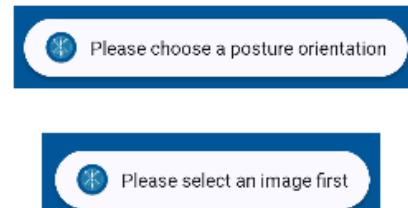


Figura 7.4: Exemplu de mesaje primite de către utilizator

În Figura 7.5 este ilustrată fereastra informativă care se afișează atunci când utilizatorul apasă pe butonul *Informații*. Aceasta oferă o listă în care se indică ce trebuie respectat și ce trebuie evitat pentru a avea o postură corectă. Scopul acestui ghid este de a ajuta utilizatorii să vizualizeze și să înțeleagă cum ar trebui să stea la birou.

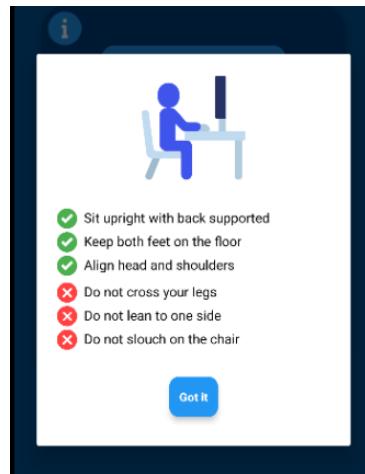


Figura 7.5: Ghid pentru menținerea unei posturi corecte

Acest capitol conturează o prezentare detaliată a modului în care aplicația poate fi instalată și utilizată de către orice utilizator, fără a fi necesare cunoștințe tehnice avansate. Instrucțiunile pas cu pas și capturile de ecran ghidează utilizatorul pe tot parcursul interacțiunii sale cu aplicația.

Capitolul 8. Concluzii

Proiectul a avut ca scop dezvoltarea unei aplicații mobile capabile să analizeze postura utilizatorilor în timp real, utilizând tehnici de viziune artificială. Scopul general a fost atins prin parcurgerea unui set de obiective secundare bine definite, care au ghidat pas cu pas implementarea.

Obiectivul principal al lucrării, dezvoltarea unei aplicații care analizează postura utilizatorului fără echipamente externe sau conexiune la internet, a fost realizat cu succes, fapt dovedit de aplicația SpinalX. Aceasta integrează un model de clasificare antrenat pe imagini reale și oferă feedback instantaneu. Acest obiectiv a fost atins prin îndeplinirea tuturor obiectivelor secundare.

Obiectivele secundare s-au concretizat astfel:

- Obiectivul 1 - Crearea unui set de date: s-a realizat prin colectarea de imagini cu persoane care stăteau la birou în diverse poziții (corecte și incorecte) și prin procesarea acestora, proces descris în Secțiunea 4.2 și în Secțiunea 5.7.
- Obiectivul 2 - Etichetarea setului de date: imaginile au fost etichetate utilizând un algoritm care analizează puncte anatomici de pe corp, detectate cu MediaPipe. Procesul este descris în Subsecțiunea 4.2.2.
- Obiectivul 3 - Antrenarea unui model: s-au antrenat mai multe modele în încercarea de a maximiza performanța, acestea sunt prezentate în Subsecțiunea 4.1.7, iar modelul integrat în aplicație este descris în Secțiunea 5.4.
- Obiectivul 4 - Evaluarea performanței modelului: s-au folosit metrii de evaluare (de exemplu: acuratețe, pierdere, precizie), toate discutate în Secțiunea 6.2.
- Obiectivul 5 - Implementarea aplicației mobile: aplicația este formată din module, arhitectura acesteia cât și alte detalii de implementare sunt prezentate în Capitolul 5 (Secțiunile 5.1, 5.2, 5.3).
- Obiectivul 6 - Integrarea modelului în aplicație: modelul a fost convertit într-un format compatibil cu Android și integrat local în aplicație, fără a fi necesară conexiune la internet, aspect prezentat în Subsecțiunea 5.4.
- Obiectivul 7 - Testarea în condiții reale – aplicația a fost testată pe dispozitive reale și validată de utilizatori, întreg procesul este descris în Subsecțiunea 6.3.

Rezultatele obținute demonstrează că modelul DenseNet121 antrenat pe setul de date creat pentru acest proiect a reușit să generalizeze informațiile învățate și să atingă o acuratețe de peste 80%. Aplicația oferă rezultatele aproape instant și funcționează fără să aibă nevoie de o conexiune la internet. Aceste caracteristici o transformă într-o soluție practică și accesibilă pentru toți utilizatorii. Prin feedbackul oferit, aplicația contribuie la îmbunătățirea sănătății utilizatorilor deoarece îi determină să conștientizeze dacă au o postură incorecta. Acest proiect a demonstrat că viziunea artificială, integrată într-o aplicație mobilă, poate fi utilizată pentru monitorizarea posturii utilizatorilor.

Pentru a îmbunătăți funcționalitatea aplicației pot fi luate în considerare următoarele dezvoltări ulterioare:

- extinderea setului de date;
- adăugarea unui istoric unde utilizatorul poate să își urmărească progresul;
- detectia posturii din secvențe video.

Bibliografie

- [1] S. Stuart, A. Godfrey, and M. Mancini, “Staying upright in parkinson’s disease: A pilot study of a novel wearable postural intervention,” *Gait & Posture*, vol. 91, pp. 86–93, 2022.
- [2] B. Millington, “‘quantify the invisible’: Notes toward a future of posture,” *Critical Public Health*, vol. 26, no. 4, pp. 405–417, 2016.
- [3] B. C. Hopkins, “Validity of posturescreen mobile in the measurement of standing posture,” Master’s thesis, Brigham Young University, Provo, UT, USA, 2014, m.S. Thesis. [Online]. Available: <https://scholarsarchive.byu.edu/etd/4119>
- [4] E. Didyk, L. K. Lewis, and B. Lange, “The efficacy of the smartphone app for the self-management of low back pain: A systematic review,” *International Journal of Environmental Research and Public Health*, vol. 21, no. 1, p. 61, 2024. [Online]. Available: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC11204566/>
- [5] R. Moreira, R. Fialho, A. S. Teles, V. Bordalo, S. S. Vasconcelos, G. P. de Moraes Gouveia, V. H. Bastos, and S. Teixeira, “A computer vision-based mobile tool for assessing human posture: A validation study,” *Computer methods and programs in biomedicine*, vol. 214, p. 106565, 2022.
- [6] S. E. Mathiassen, “Diversity and variation in biomechanical exposure: what is it, and why would we like to know?” *Applied ergonomics*, vol. 37, no. 4, pp. 419–427, 2006.
- [7] A. L. Cohen, *Elements of Ergonomics Programs: A Primer Based on Workplace Evaluations of Musculoskeletal Disorders*. DIANE Publishing, 1997.
- [8] M. O'Reilly, B. Finder, and M. K. Werrell, *An Ergonomics Guide to Computer Workstations*. AIHA, 2007.
- [9] Occupational Safety and Health Administration (OSHA), “Computer workstations etool,” <https://www.osha.gov/etools/computer-workstations>, 2008, accesată: 2025-02-22.
- [10] B. Kristyanto, B. B. Nugraha, A. K. Pamosoaji, and K. A. Nugroho, “Head and neck movement: simulation and kinematics analysis,” *Procedia Manufacturing*, vol. 4, pp. 359–372, 2015.
- [11] W. Tapanya, M. S. Neubert, R. Puntumetakul, and R. Boucaut, “The effects of shoulder posture on neck and shoulder musculoskeletal loading and discomfort during smartphone usage,” *International Journal of Industrial Ergonomics*, vol. 85, p. 103175, 2021.
- [12] B. Jo and S. Kim, “Comparative analysis of openpose, posenet, and movenet models for pose estimation in mobile devices,” *Traitement du Signal*, vol. 39, no. 1, p. 119, 2022.

- [13] S. Negi, M. Garg, H. Maindola, V. Kansal, U. Jain, and S. Bhatla, “Real-time human pose estimation: A mediapipe and python approach for 3d detection and classification,” in *2023 3rd International Conference on Technological Advancements in Computational Sciences (ICTACS)*. IEEE, 2023, pp. 128–133.
- [14] Z. Min, “Human body pose intelligent estimation based on blazepose,” in *2022 IEEE International Conference on Electrical Engineering, Big Data and Algorithms (EEBDA)*, 2022, pp. 150–153.
- [15] W. Kim, J. Sung, D. Saakes, C. Huang, and S. Xiong, “Ergonomic postural assessment using a new open-source human pose estimation technology (openpose),” *International Journal of Industrial Ergonomics*, vol. 84, p. 103164, 2021.
- [16] H. Coskun, “Human sitting postures classification based on angular features with fuzzy-logic labeling,” in *Proceedings of the 5th International Conference on Applied Engineering and Natural Sciences*, 2023, pp. 611–619.
- [17] P.-C. Lin, Y.-J. Chen, W.-S. Chen, and Y.-J. Lee, “Automatic real-time occupational posture evaluation and select corresponding ergonomic assessments,” *Scientific Reports*, vol. 12, no. 1, p. 2139, 2022.
- [18] M. Gómez-Galán, J. Pérez-Alonso, Á.-J. Callejón-Ferre, and J. López-Martínez, “Musculoskeletal disorders: Owas review,” *Industrial health*, vol. 55, no. 4, pp. 314–337, 2017.
- [19] S. Hignett and L. McAtamney, “Rapid entire body assessment (reba),” *Applied ergonomics*, vol. 31, no. 2, pp. 201–205, 2000.
- [20] L. McAtamney and E. N. Corlett, “Rula: a survey method for the investigation of work-related upper limb disorders,” *Applied ergonomics*, vol. 24, no. 2, pp. 91–99, 1993.
- [21] A. Kulikajevas, R. Maskeliunas, and R. Damaševičius, “Detection of sitting posture using hierarchical image composition and deep learning,” *PeerJ Computer Science*, vol. 7, p. e442, 2021.
- [22] T. P. Trappenberg, *Fundamentals of machine learning*. Oxford University Press, 2019.
- [23] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*. MIT press Cambridge, 2016, vol. 1, no. 2.
- [24] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*. Springer, 2006, vol. 4, no. 4.
- [25] A. Singh, N. Thakur, and A. Sharma, “A review of supervised machine learning algorithms,” in *2016 3rd international conference on computing for sustainable global development (INDIACoM)*. Ieee, 2016, pp. 1310–1315.
- [26] Z. Ghahramani, “Unsupervised learning,” in *Summer school on machine learning*. Springer, 2003, pp. 72–112.

- [27] N. N. Pise and P. Kulkarni, “A survey of semi-supervised learning methods,” in *2008 International conference on computational intelligence and security*, vol. 2. IEEE, 2008, pp. 30–34.
- [28] R. S. Sutton, A. G. Barto *et al.*, *Reinforcement learning: An introduction*. MIT press Cambridge, 1998, vol. 1, no. 1.
- [29] K. O’shea and R. Nash, “An introduction to convolutional neural networks,” *arXiv preprint arXiv:1511.08458*, 2015.
- [30] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola, “Padding and strides — dive into deep learning,” https://classic.d2l.ai/chapter_convolutional-neural-networks/padding-and-strides.html, accesat în 15 martie 2025.
- [31] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, “A survey of convolutional neural networks: analysis, applications, and prospects,” *IEEE transactions on neural networks and learning systems*, vol. 33, no. 12, pp. 6999–7019, 2021.
- [32] R. Shyam, “Convolutional neural network and its architectures,” *Journal of Computer Technology & Applications*, vol. 12, no. 2, pp. 6–14, 2021.
- [33] H. Zhang, L. Zhang, and Y. Jiang, “Overfitting and underfitting analysis for deep learning based end-to-end communication systems,” in *2019 11th international conference on wireless communications and signal processing (WCSP)*. IEEE, 2019, pp. 1–6.
- [34] X. Ying, “An overview of overfitting and its solutions,” in *Journal of physics: Conference series*, vol. 1168. IOP Publishing, 2019, p. 022022.
- [35] S. Mascarenhas and M. Agarwal, “A comparison between vgg16, vgg19 and resnet50 architecture frameworks for image classification,” in *2021 International conference on disruptive technologies for multi-disciplinary research and applications (CENTCON)*, vol. 1. IEEE, 2021, pp. 96–99.
- [36] T. Ibtekar, M. R. Haque, and A. Y. Srizon, “Comparative study between resnet and efficientnet family for classification of leukemia,” in *2023 26th International Conference on Computer and Information Technology (ICCIT)*. IEEE, 2023, pp. 1–6.
- [37] M. Tan and Q. Le, “Efficientnetv2: Smaller models and faster training,” in *International conference on machine learning*. PMLR, 2021, pp. 10 096–10 106.
- [38] K. Dong, C. Zhou, Y. Ruan, and Y. Li, “Mobilenetv2 model for image classification,” in *2020 2nd International Conference on Information Technology and Computer Application (ITCA)*. IEEE, 2020, pp. 476–480.
- [39] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.
- [40] Roboflow, “Posture detection dataset and project,” <https://universe.roboflow.com/netraininitial/posture-detection>, accesat la 14 martie 2025.

- [41] Roboflow, “Human activity detection (hrscm),” <https://universe.roboflow.com/ss-wq1cw/human-activity-detection-hrscm>, accesat la 14 martie 2025.
- [42] Roboflow, “Human activity recognition (har),” <https://universe.roboflow.com/uni-project-vsqqp/har-no7r2>, accesat 14 martie 2025.
- [43] Roboflow, “Employee productivity 2.0,” <https://universe.roboflow.com/bahadur-pelqz/employee-productivity-2.0-whbwv>, accesat la 14 martie 2025.
- [44] Google AI, “Mediapipe pose landmarker,” https://ai.google.dev/edge/mediapipe/solutions/vision/pose_landmarker, accesat în 04 aprilie 2025.
- [45] K. O’Sullivan, P. O’Sullivan, L. O’Sullivan, and W. Dankaerts, “What do physiotherapists consider to be the best sitting spinal posture?” *Manual Therapy*, vol. 17, no. 5, pp. 432–437, 2012. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1356689X12000938>
- [46] J. Steczko, “Analysis of companies’ experience with cross-platform development compared to native development for mobile devices,” 2016.
- [47] J. Nielsen, “Response times: The 3 important limits,” <https://www.nngroup.com/articles/response-times-3-important-limits>, 1993, accesat în 29-05-2025.